Concept Formation During Interactive Theory Revision

STEFAN WROBEL

WROBEL@GMD.DE

German National Research Center for Computer Science (GMD), FIT.KI, P.O. Box 1316,
53757 Sankt Augustin, Fed. Rep. Germany

Abstract. This article examines the problem of concept formation in machine learning, and focuses in particular on the problem of aggregation, i.e., the decision of which objects are to be grouped together into a new concept. While existing concept formation approaches have mainly concentrated on aggregation constraints that rely on structural or correlational properties of the concepts themselves, we argue that in an integrated learning system, other learning activities can provide an additional context that focuses concept formation before structural criteria are applied. In particular, we present the concept formation method realized by the KRT and CLT components of the integrated learning system MOBAL. In MOBAL, a concept formation attempt is triggered whenever no existing concept can adequately capture the rule instance and exception sets as they arise from the theory revision activities of the system. We describe how the so-proposed aggregate is characterized by a set of (function-free) first-order Horn clauses and how these are evaluated according to structural criteria to decide about the introduction of the concept into the representation. We show how a structural criterion can be used to ensure that any new concept improves the structure of the knowledge base, and we empirically evaluate how the introduction of new concepts according to different criteria affects the classification accuracy of learned rules.

Keywords. Concept formation, theory revision, representation change, new-term problem, MOBAL, KRT, CLT

1. Introduction

One of the central problems studied in machine learning is the task of acquiring concepts in a learning system. Depending on the assumptions made about the available data, this task comes in two variants. In the first and simpler case, it is assumed that the learner is given sets of positive and negative examples of the target concept for which it is to induce an intensional definition. This task is usually referred to as *concept learning from examples*. In the second and more complex case, we assume that the learner is given only a set of objects that is not partitioned into positive and negative examples of a concept. Here, the learner must first decide which concepts to form and which objects are to belong (or not to belong) to these concepts. This task is usually referred to as *concept formation*.

Borrowing terminology from Easterlin and Langley (1985), we can more precisely define the concept formation task as follows:

- *Given* a set of object (instance, event, case) descriptions (usually presented incrementally),
- *Find* sets of objects that can usefully (according to the goals of the learner) be grouped together (*aggregation*), and
- *Find* intensional definitions for these sets of objects (*characterization*).¹

The nonincremental variant of this task is usually referred to as *conceptual clustering*. As evident from the above definition, the characterization step of concept formation corresponds to the task of concept learning from examples. If the new concepts are to be introduced into the system's representation permanently, an additional requirement is

- *Define* a new name (predicate) for the new concept, and introduce it into the representation so that it can be used elsewhere, for instance, in the definition of further concepts, or for the description of future input objects.

In hierarchical concept formation systems (Michalski & Stepp, 1983; Lebowitz, 1987; Fisher, 1987; Gennari et al., 1989), this latter step is usually not included, i.e., these systems aggregate objects and characterize them, but do not give them names that are available elsewhere.

In this article, we are primarily concerned with the aggregation part of the concept formation task, i.e., the question of how to find useful sets of objects to group together, where usefulness, of course, must be defined with respect to the goals of the learner or user—for instance, classification accuracy or structural simplicity of the theory. Following results from psychology, we will regard aggregation as a process of grouping together a set of objects that have played a similar role in a given reasoning or problem-solving context. As a particular instantiation of this general aggregation strategy, we will present the concept formation method used in the learning and knowledge acquisition system MOBAL (Morik et al., 1993). In the MOBAL system, the aggregation context is provided by the *theory revision* activities performed by KRT, the system's knowledge revision tool. Whenever KRT cannot correctly specialize an incorrect rule, the rule's instances and exceptions are used as an aggregate for a proposed new concept. This aggregate is then passed to CLT, MOBAL's concept learning tool, which induces an intensional definition (a set of first-order clauses), evaluates the quality of the proposed new concept based on structural criteria, and uses the new concept to restructure the knowledge base. The concept formation task is thus distributed between KRT (aggregation) and CLT (characterization, evaluation, and utilization).

This method of concept formation is based on the original idea of exploiting exceptions in METAXA (Emde et al., 1983) and our previous work on concept formation in BLIP, respectively, the MODELER (Wrobel, 1988; Wrobel, 1989), but uses a much improved concept characterization step, is capable of forming relational (n-ary) concepts, employs a theoretically more sound restructuring step based on resolution and θ -subsumption, and monitors concept quality over time. This article also includes theoretical and empirical evaluations of concept quality not present in our earlier work.

The rest of the article is organized as follows. In section 2, we will give an overview of the aggregation constraints used in previous concept formation methods and give the rationale for the context-based constraint that was used in this work. In section 3, we will introduce the MOBAL system, its components, and its knowledge representation. In section 4, we will then detail the components of the concept formation process, i.e., aggregation as performed in the context of KRT (section 4.1), and characterization, evaluation, and utilization in CLT (sections 4.2–4.4). Section 5 contains both theoretical and empirical evaluations of our method, followed by a review of related work (section 6) and our conclusions (section 7).

2. Constraints on aggregation

In concept learning from examples, the learning system is given a set of positive examples known to be members of the target concept and a set of negative examples known to be outside of the target concept. In concept formation, the learner is given only a set of objects or observations and must first decide which of these objects to collect into an aggregate and then use as the set of positive examples for the new concept. Since there are exponentially many aggregates (subsets) in a given object set, concept formation requires strong constraints on aggregation. Existing concept formation systems have mainly used three types of constraints on aggregation: similarity based, correlation based, and structure based.

Similarity-based constraints rely on a context-free numerical measure of similarity between two objects, such as Euclidean distance. Such measures are most heavily used in numerical taxonomy systems (Everitt, 1980), but have also been used in UNIMEM (Lebowitz, 1987) to decide when to combine two instances into a more general concept. Similarity-based constraints are simple, but the quality of the concepts that are formed depends very much on a proper definition of the similarity measure. For example, it has been noted that the effect of UNIMEM's parameters on its results remains unclear (Gennari et al., 1989). Moreover, psychological studies of human similarity judgments show that perceived similarity depends heavily on both context and task (Tversky, 1977), and these are not incorporated into context-free numerical similarity measures.

Feature correlation constraints are based on the assumption that the best conceptual systems are those that maximize intraconcept correlations and minimize interconcept correlations. This *correlated feature* hypothesis was first formulated in psychological work by Rosch and colleagues (Rosch et al., 1976; Rosch, 1978). Correlation measures are not based on distances between individual objects, but can only be applied to proposed *sets* of concepts. They have been used in many existing concept formation approaches. The CLUSTER system (Michalski & Stepp, 1983) uses cluster distance, discrimination index, and dimensionality reduction as indirect measures of feature correlation in its lexicographic evaluation function (LEF); UNIMEM (Lebowitz, 1987) uses integer counters of predictability and predictiveness to decide when to fix a feature as part of a concept definition; COBWEB (Fisher, 1987) and its successor CLASSIT (Gennari et al., 1989) both use probabilistic correlation measures based on feature correlations (*category utility*) to compare proposed clusterings; WITT (Hanson & Bauer, 1989) uses another correlation-based measure (*cohesion*) for bottom-up clustering.

Structural constraints refer to the syntactical structure of individual concept descriptions, or to the structure of the entire conceptual system. In CLUSTER (Michalski & Stepp, 1983), only aggregates that could be characterized by conjunctive expressions were considered, and those with shorter definitions (total number of "selector" conjuncts used in the entire clustering) were preferred. In CLUSTER/G (Stepp & Michalski, 1986), the available descriptors had associated weights derived from a goal dependency network, and the system preferred clusterings that used descriptors with high weights. In KLUSTER (Kietz & Morik, this issue), aggregation is restricted to subsets of the set of instances of a common super-concept, which are required to be disjoint. For CIGOL, a structural criterion based on information compression has been employed (Muggleton, 1988).

Context-based constraints. The above types of constraints on concept formation have in common that they all refer only to the properties of concepts and the concept formation process proper. The *context* in which concept formation is embedded is not taken into account. Nonetheless, psychological evidence as put forth, for instance, in Nelson's (1983) work on concept formation from event contexts and in Barsalou's (1983) research on ad-hoc concepts strongly suggests that humans rely on contextual constraints quite heavily: people choose to aggregate sets of objects that play a similar role in the context of a reasoning or problem-solving activity of interest to them and name these sets with a new concept because they need to be able to refer to the aggregate independent of the original context. So-called ad-hoc categories are created for and during the solution of one particular task, and generally have the form "things instrumental to achieving goal X." While extensional at first, frequently used ad-hoc categories (like "things to sell at a garage sale," which may be useful every Saturday) may become fully accessible concepts by an additional characterization step, and then exhibit empirical properties similar to standard object concepts (Barsalou, 1983).

Correspondingly, we can technically define the use of contextual constraints in a learning system as *the introduction of a concept precisely for a set of objects that have turned out to be instrumental for a particular task in the learning system*; this task, in turn, is the context of concept formation. Since the introduction of a new concept responds to a need for a concept in the system's processing, we also refer to this strategy as *demand-driven* concept formation. As pointed out in the introduction, for the concept formation approach in the MOBAL system, we have exploited the theory revision activities of KRT as one particular context for concept formation. As we will see in section 4.1, it naturally defines the set of successful instances of a rule as an interesting aggregate for concept formation. We should also emphasize that context-based constraints are in some sense "orthogonal" to the other constraints named above, since after triggering aggregation based on context, it is very possible (and done in MOBAL) to judge the quality of the concept's characterization with structural criteria. Before describing the details of this process in section 4, we will now briefly introduce MOBAL and its knowledge representation.

3. The MOBAL system

The MOBAL system (Morik, 1993; Morik et al., 1993) is an interactive knowledge acquisition system that integrates machine learning techniques according to the *balanced cooperative modeling* paradigm of K, Morik (1991). This paradigm regards knowledge acquisitions as a cyclic interactive process of building a model of a domain in which both user and (learning) system contribute to the development of the model in a balanced fashion. Central requirements of the paradigm are system support for the inspection of a knowledge base, for its inductive extension, and for both monotonic and nonmonotonic interactive revisions of the evolving model.

MOBAL consists of a number of interacting modules that address the above requirements, namely, a powerful graphical interface, an inference engine (Emde, 1989), sort taxonomy (Kietz, 1988), and predicate structuring tools (Klingspor, 1991), the first-order learning algorithm RDT (Kietz & Wrobel, 1992), the knowledge revision tool KRT, and the tools

for concept learning and rule set restructuring, CLT and RRT.² A *domain model* in MOBAL consists of a variety of different knowledge sources. For the purposes of this article, we are most interested in the facts and rules that are stored in the inference engine, and we will be referring to them as the knowledge base or the theory of the system, ignoring the other knowledge sources that exist in the system (e.g., metapredicates, rule models, sort taxonomy, and predicate topology).

With this restriction in mind, we can say that MOBAL uses the function-free subset of Horn-clause logic as its knowledge representation formalism, extended by computed predicates and negated literals in the body. Following the terminology from Lloyd (1987), we can define this representation more precisely as follows.³ A *predicate* is denoted by a predicate symbol (usually $p, q, r \dots$) and has a fixed arity a . A *term* is a constant (denoted by c_1, c_2 , or other lower-case letters) or a variable (denoted by X, Y , or other upper-case letters). An *atom* is a predicate applied to terms, i.e., of the form $p(t_1, \dots, t_a)$. A *positive literal* is an atom; a *negative literal* is the negation of an atom, i.e., $\neg p(t_1, \dots, t_a)$. A *clause* is a finite disjunction of literals where all variables are implicitly universally quantified. It is usually written as an implication:

$$L_1 \& \dots \& L_n \rightarrow L_c$$

where L_c , the *conclusion*, or *head*, and the L_i , the *premises*, or the *body* of the clause, are positive or negative literals. If $n = 0$, the clause is called a *fact*, and usually written without the $\rightarrow$. All facts are required to be *ground*, i.e., contain no variables. If $n \geq 1$, such a clause is called a *rule*, and we let $V(R)$ denote the (ordered) set of the rule's variables. In contrast to Prolog, negative literals as premises of a rule are not treated as negation by failure, but as proper negation, i.e., negative premises require explicitly negated facts in the knowledge base to be satisfied.

As an example, table 1 shows several facts and rules from a MOBAL knowledge base about German traffic law. The knowledge base consists of 15 cases of various traffic violations; the facts shown in table 1 are what is represented about the first violation case. For this example, which we will use throughout the article for illustration, the classification goals is to decide who is responsible for a traffic violation. The rule at the bottom of the table shows a classification rule learned by RDT from the first 12 cases in the knowledge base. As we will see below, this rule is overly general, and needs to be revised.

Table 1. An example from the "traffic law" domain.

```
involved_vehicle(event1,b_au_6773)
sedan(b_au_6773)
owner(sw,b_au_6773)
car_parked(event1,place1)
bus_lane(place1)
parking_violation(event1)
involved_vehicle(X,Y) & owner(Z,Y) →
responsible(Z,X)
```

Each rule R (with variables $V(R) := \{X_1, \dots, X_n\}$) has an attached *support set* (Emde et al., 1983; Wrobel, 1989) $S(R)$, which is an expression of the form

$$(p_1 \setminus LE_1) \times \dots \times (p_n \setminus LE_n) \setminus GE.$$

The p_i are predicate symbols or the special symbol all , the LE_i are sets of constant symbols and called *local exception sets*, and GE , the set of *global exceptions*, has the form

$$\{(c_{1,1}, \dots, c_{1,n}), \dots, (c_{m,1}, \dots, c_{m,n})\}.$$

where the $c_{j,i}$ are constant symbols, and $m \geq 0$. The meaning of a support set can be defined by the following set of rule premises:

$$p_1(X_1) \& \dots \& p_n(X_n) \& X_1 \notin LE_1 \& \dots \& X_n \notin LE_n \& (X_1, \dots, X_n) \notin GE.$$

The meaning of the special symbol all is defined by

$$\text{all}(X) \equiv \text{true}.$$

A support set is thus a shorthand notation that permits convenient restriction of a rule's applicability by specifying exceptions sets; this facility is used by KRR to perform minimal specializations on rules (cf. section 4.1). For the example rule from table 1, the unrestricted default support set would be

$$\text{all} \times \text{all} \times \text{all}$$

MOBAL's inference engine [M-2 (Emde, 1989; Morik et al., 1993)] is capable of storing a knowledge base of facts and rules (including support sets), and of performing forward and backward inferences with it. Derivation traces are maintained, so that reason maintenance can take place when part of the knowledge base is changed. A number of computed or auto-epistemic predicates are allowed as premises and are properly handled.

Finally, within this representation, by a *concept* we mean both a name and the set of rules that define the name. The name is a unary or n-ary predicate; n-ary predicates are also referred to as *relations*. The set $\text{ext}(c)$, where c is the predicate that names the concept, is called the *extension* of the concept in a knowledge base Γ and is defined as $\text{ext}(c) := \{(a_1, \dots, a_n) \mid c(a_1, \dots, a_n) \in C_n(\Gamma)\}$, where $C_n(\Gamma)$ denotes the inferential closure of Γ . We correspondingly define $\text{ext}(\neg c) := \{(a_1, \dots, a_n) \mid \neg c(a_1, \dots, a_n) \in C_n(\Gamma)\}$. The set of rules of a concept consists of all rules in which the concept predicate occurs and is also referred to as the concept's *intension*. The rules that contain a concept predicate in their conclusion are the *sufficient conditions* $SC_\Gamma(c)$, since they infer concept membership from other information. The rules that have the concept as their only premise state *necessary conditions* $NC_\Gamma(c)$, since the absence of the conclusion implies nonmembership in the concept. Finally, all other rules mentioning a concept are referred to as the *uses* $U_\Gamma(c)$ of the concept.⁴

Note that this definition of concepts permits the representation of polymorphic concepts (Hanson & Bauer, 1989), i.e., concepts the members of which do not share a common

set of features that sufficiently characterizes the concept; the set of necessary conditions is not jointly sufficient, and the set of sufficient conditions is disjunctive. This representation thus addresses many of the concerns that have been put forth by psychologists about “classical” models of concepts (Smith & Medin, 1981) (see Wrobel (1991a, 1991b) for discussions of related issues).

4. Concept formation during theory revision

4.1. Aggregation in KRT

In a typical usage scenario of MOBAL, a user inputs a set of cases, additional background knowledge rules, and then lets the system discover rules about the target predicates of the domain. Since RDT, the inductive component of MOBAL, learns the most general rules that are instances of its rule schemata and that meet its acceptance criterion (Kietz & Wrobel, 1992), the resulting knowledge base is sometimes overly general, and thus incorrect. Similarly, user input rules often address the main cases only, being incorrect elsewhere. In this situation, the knowledge revision tool KRT can be used to specialize the knowledge base interactively and incrementally.

As an example of KRT’s task, consider the traffic law domain introduced in table 1. The rule given at the bottom of this table,

r5: `involved_vehicle(X,Y) & owner(Z,Y) → responsible(Z,X)`

states that the owner of a vehicle is always responsible, which is overly general—actually, he is responsible only for minor violations; otherwise, the driver is responsible. Consequently, when applied to three cases of major violations, the learned rule produced the false positives (shown with the corresponding antecedents):

`involved_vehicle(cab_event,cab1) owner(ace_cab.co, cab1)
responsible(ace_cab.co,cab_event)
involved_vehicle(loan_event,b_xs_400) owner(sw, b_xs_400)
responsible(sw, loan_event)
involved_vehicle(stolen_event,b_dx_986) owner(dx,b_dx_986)
responsible(dx, stolen_event)`

The user thus calls KRT to change the knowledge base so that these incorrect facts can no longer be derived.

KRT’s task consists of two parts. Given an incorrect theory Γ , the first step is to select a set of clauses (facts or rules) in Γ for specialization (blame assignment). This step is performed by computing, from the derivation trace information in the inference engine, the set of *minimal removal sets*, each of which is a minimal set of clauses the removal of which would be sufficient to prevent the incorrect derivation (Wrobel, 1993b). To choose among these sets, KRT computes a confidence value between 0 and 1 for each rule in the knowledge base based on the ratio of known applications and known exceptions (as stated

in the rule's support set), whereas the confidence of input facts can be assigned by the user.⁵ KRT then proposes to the user the removal set that involves the minimal loss of confidence (see Wrobel (1993a) for more detail).

A removal set can be more precisely defined as follows. A pair (R, V) of a rule R and a tuple of values V is called a *rule application* iff there is a substitution σ and facts $g, g_1, \dots, g_n \in Cn(\Gamma)$ such that $V = V(R)\sigma$, and R resolves with $g_1, \dots, g_n$ to produce g with substitution σ . We let Σ_Γ denote the set of all applications of R in $Cn(\Gamma)$. A removal set contains facts and rule applications to be removed. In our example, let us assume that KRT (or the user) has chosen the rule `r5` shown above as the culprit for all three incorrect derivations. In this simple case, the chosen minimal removal set, call it M , would be

$$\{ \begin{array}{l} (\text{r5}, (\text{cab_event}, \text{cab1}, \text{ace_cab_co}, \text{cab1})) \\ (\text{r5}, (\text{loan_event}, \text{b_xs_400}, \text{sw})) \\ (\text{r5}, (\text{stolen_event}, \text{b_dx_986}, \text{dx})) \end{array} \}$$

KRT then begins the second part of its task. Once a removal set is selected, its members must be removed or specialized so they cannot be used in the incorrect derivation anymore. Since (ground) facts cannot be specialized further, they are deleted. For rules, however, further specialization can be performed. This specialization step is the context for concept formation, so we will describe it in some detail. Based on the derivation trace information from the inference engine, KRT determines the bindings of all variables in the chosen rule for both successful and unsuccessful applications. This can be more precisely defined as follows. For any rule $R \in \Gamma$ with a support set $S(R) = D_1 \times \dots \times D_n \setminus GE$, let

$$\begin{aligned} I_{\Gamma, M}(R) &:= \Sigma_\Gamma(R) \setminus \{V \mid (R, V) \in M\} \\ E_{\Gamma, M}(R) &:= \{V \mid (R, V) \in M\} \cup GE. \end{aligned}$$

In other words, $I(R)$ (we will drop the subscripts of I and E in the following) is the set of rule *instances*, i.e., applications that must be preserved, whereas $E(R)$ is the set of rule *exceptions*, i.e., cases where the rule must not apply. The latter set contains the previously acquired exceptions as stored in the support set plus the new exceptions arising from the current revision.⁶ In the example, we would find

$$\begin{aligned} I(R) = \{ & (\text{event1}, \text{b_au_6773}, \text{sw}), & (\text{event2}, \text{b_dx_1385}, \text{dj}), \\ & (\text{event3}, \text{hh_mo_195}, \text{mo}), & (\text{event4}, \text{b_md_4321}, \text{md}), \\ & (\text{event5}, \text{b_st_888}, \text{st}), & (\text{event6}, \text{b_ab_89}, \text{ab}), \\ & (\text{event7}, \text{b_bc_90}, \text{bc}), & (\text{event8}, \text{b_cd_01}, \text{cd}), \\ & (\text{event9}, \text{b_de_12}, \text{de}), & (\text{event10}, \text{b_ef_23}, \text{ef}), \\ & (\text{event11}, \text{b_fg_34}, \text{fg}), & (\text{event12}, \text{b_gh_45}, \text{ef}) \\ E(R) = \{ & (\text{cab_event}, \text{cab1}, \text{ace_cab_co}), & (\text{loan_event}, \text{b_xs_400}, \text{sw}), \\ & (\text{stolen_event}, \text{b_dx_986}, \text{dx}) & \}. \end{aligned}$$

We can now define the task of KRT's second step, the specialization task, precisely:

Given: A rule R in a knowledge base Γ , and instance and exception sets $I(R)$ and $E(R)$,

Find: A new rule R' such that $\Sigma(R) \supseteq \Sigma(R') \supseteq I(R)$, and $\Sigma(R') \cap E(R) = \emptyset$.

This task must be performed by KRT for all rules mentioned in M .

To perform such a specialization, KRT possesses a number of specialization operators (also called *reformulation* operators in MOBAL); they are shown in table 2. In this table, $\pi_{i_1, \dots, i_m}(S)$ of a set S of n -tuples ($m \leq n$) denotes the projection of S on positions $i_1, \dots, i_m$. In the rule representation of MOBAL, these operators produce specializations of R that do not introduce additional existential variables and do not remove existing premises. The first operator, *Minimal specialization*, produces the maximally general correct specialization of the given rule by simply adding the new exceptions to the support set's global exception list. The *Localization* operator specializes further, trying to find restrictions on single variables sufficient to exclude all exceptions. Operator (2) should be familiar; it adds a new premise to the rule. The goal of KRT in this step is to find a *perfect* specialization, i.e., a rule R' with empty sets GE and LE_i , since rules with explicit lists of exceptions are regarded as less plausible than rules with intensionally defined domains of applicability.⁷ To this end, the operators in table 2 are tried in order, the next being tried only if the preceding did not find a perfect specialization.

In the example, the minimal specialization operator produces the support set

$$\text{all} \times \text{all} \times \text{all} \setminus \{(\text{cab_event}, \text{cab1}, \text{ace_cab_co}), \\ (\text{loan_event}, \text{b_xs_400}, \text{sw}), (\text{stolen_event}, \text{b_dx_986}, \text{dx})\}.$$

The *Localize* operator would be applied only for $i \in \{1, 2\}$ (since $\pi_3(I(R)) \cap \pi_3(E(R)) = \{\text{sw}\}$) and produces

$$(\text{all} \setminus \{\text{cab_event}, \text{loan_event}, \text{stolen_event}\}) \text{all} \times \text{all} \\ \text{all} \times (\text{all} \setminus \{\text{cab1}, \text{b_xs_400}, \text{b_dx_986}\}) \times \text{all}$$

all of which require explicit exception lists to be correct. Since no existing predicate covers all instances of the rule (after all, there is no predicate corresponding to "minor violation" yet), operator (2) is not applicable.

Table 2. KRT's reformulation operators.

For R with $V(R) = X_1, \dots, X_n$, $S(R) = D_1 \times \dots \times D_n \setminus GE$ and $I(R)$, $E(R)$ as defined in the text:

(0) **Minimal specialization.** Set $S(R) := D_1 \times \dots \times D_n \setminus E(R)$.

(1) **Localization.** For i such that $\pi_i(I(R)) \cap \pi_i(E(R)) = \emptyset$, replace $D_i = \pi_i \setminus LE_i$ with $\pi_i \setminus (LE_i \cup \pi_i(E(R)))$ and GE by $\emptyset$ in $S(R)$.

(2) **Add existing predicate.** Select a subset $V' = \{X_{i_1}, \dots, X_{i_m}\} \subseteq V(R)$, and an existing predicate p of arity $|V'|$, let (a) $P = p$ or (b) $P = \neg p$. If $\text{ext}(P) \supseteq \pi_{i_1, \dots, i_m}(I(R))$, and $\pi_{i_1, \dots, i_m}(I(R)) \cap \pi_{i_1, \dots, i_m}(E(R)) = \emptyset$, replace $S(R)$ by $D_1 \times \dots \times D_n \setminus \{t \in E(R) \mid \pi_{i_1, \dots, i_m}(t) \in \text{ext}(P)\}$ and add $P(V')$ to the body of R . [If $m = 1$, use $\pi_i(E(R)) \cup LE_i$ instead of $\pi_i(E(R))$.]

We can now see how the specialization task of KRT for a rule R constitutes an aggregation context for concept formation. This task precisely defines the two sets $I(R)$ and $E(R)$, which, corresponding to the general rationale laid out in section 2, are the sets of objects that can (respectively, cannot) be used successfully in this rule. Furthermore, the specialization context also identifies when it is necessary to introduce a new concept for these sets: whenever the system cannot find a perfect specialization using operators (0), (1), and (2) alone, as was the case in the above example. We can thus define one further specialization operator for KRT, as shown in table 3. This operator triggers a concept formation attempt using (a projection of) $I(R)$ as the instances and (a projection of) $E(R)$ as the exceptions of the proposed new concept. The default in KRT is to attempt concept formation only on single-dimension projections of $I(R)$ and $E(R)$, i.e., to introduce unary concepts. In our example, KRT would thus call CLT with the π_1 and π_2 projections (the π_3 projection has sw as intersection). For the π_1 projection, the aggregate supplied to CLT is

$$\begin{aligned} I(R) &= \{ \text{event1, event2, event3, event4, event5, event6, event7,} \\ &\quad \text{event8, event9, event10, event 11, event12} \} \\ E(R) &= \{ \text{cab_event, loan_event, stolen_event} \}. \end{aligned}$$

Note that since CLT applies its own quality criteria (see below) to each proposed concept, not all concept formation attempts are successful. Among all the specializations found by operators (0) through (3), KRT selects the one with smallest number of explicitly remaining global exceptions, breaking ties in favor of specializations with existing concepts. If no specialization was found, the rule is deleted. Table 4 summarizes the control structure of KRT's specialization step.

Table 3. KRT's reformulation operator for introducing a new concept.

For R with $V(R) = X_1, \dots, X_n$, $S(R) = D_1 \times \dots \times D_n \setminus GE$ and $I(R)$, $E(R)$ as defined in the text:

- (3) **Aggregation.** Select a subset $V' = \{X_{i_1}, \dots, X_{i_m}\} \subseteq V(R)$. If $\pi_{i_1, \dots, i_m}(I(R)) \cap \pi_{i_1, \dots, i_m}(E(R)) = \emptyset$, supply $\pi_{i_1, \dots, i_m}(I(R))$ and $\pi_{i_1, \dots, i_m}(E(R))$ as members and non-members of a new concept to CLT. If CLT successfully forms a new concept c from this aggregate such that $\text{ext}(c) \supseteq \pi_{i_1, \dots, i_m}(I(R))$ replace $S(R)$ by $D_1 \times \dots \times D_n \setminus \{t \in E(R) \mid \pi_{i_1, \dots, i_m}(t) \in \text{ext}(c)\}$ and add $c(V')$ to the body of R . [If $m = 1$, use $\pi_{i_1}(E(R)) \cup LE_{i_1}$ instead of $\pi_{i_1}(E(R))$.]

Table 4. Control structure of KRT's specialization step.

1. Apply the *Minimal specialization operator*. If the minimal specialization meets the user-set plausibility criterion, return it as result. Otherwise, go to the next step.
2. Apply operators (1) and (2). If they produce "perfect" specializations (i.e., not requiring explicit exception lists), return one of these as the result. Otherwise, go to the next step.
3. Apply the aggregation operator (3) for new concepts up to arity A (parameter, default is 1). Go to the next step.
4. From the result of all operators, return the specialization with the fewest number of explicit exceptions as result. If there are several, prefer results of operator (1) over operator (2) over operator (3). If no specialization was found, query the user whether rule should be deleted.

4.2. Characterization

After KRT has already performed the aggregation step of concept formation, the task of CLT is to characterize, evaluate, and if successful use the new concept in the knowledge base. As pointed out above, given instances and noninstances of the proposed concept by KRT, the characterization step performed in CLT merely involves the invention of a new name c for the concept and the introduction of positive and negative facts about c into the knowledge base according to the supplied aggregate, and then reduces to a learning-from-examples problem in MOBAL's function-free Horn clause representation. Since we are interested not only in sufficient condition rules about c , however, the learning algorithm that is used must also be capable of searching for rules with c in the body. In our example, the knowledge base in which the characterization step is performed consists of the original knowledge base including all the cases plus background knowledge, plus the facts that extensionally define the new concept, which CLT has named $c1$:

$c1(event1).$	$c1(event2).$	$c1(event3).$
$c1(event4).$	$c1(event5).$	$c1(event6).$
$c1(event7).$	$c1(event8).$	$c1(event9).$
$c1(event10).$	$c1(event11).$	$c1(event12).$
$\neg c1(cab_event).$	$\neg c1(loan_event).$	$\neg c1(stolen_event).$

In MOBAL, we are using the learning module RDT (Kietz & Wrobel, 1992) for the purpose of characterizing a new concept. RDT is a model-driven most-general learner, i.e., it finds all most general rules that are instances of its set of second-order rule schemata and that meet its (user-settable) acceptance criterion. In our example, RDT has used the rule schema

$$P(X, Y) \ \& \ Q(Z, Y) \rightarrow R(Z, X)$$

to learn the rule numbered $r5$ with the second-order substitution

$$\{P / involved_vehicle, Q / owner, R / responsible\}.$$

This use of rule schemata allows the hypothesis space to be adapted ahead of time so that only rules with a new concept c in the premises are found. In the example, we would replace the original rule schema by the two more special rule schemata

$$c(X, Y) \ \& \ Q(Z, Y) \rightarrow R(Z, X) \ . \ P(X, Y) \ \& \ c(Z, Y) \rightarrow R(Z, X) \ .$$

The two rule schemata above would then become part of a special set of rule schemata to be used when we are trying to characterize binary new concepts, and need not be considered for unary new concepts.⁸

To perform the characterization, RDT is called several times with different arguments. In the first call, the new concept is specified as conclusion and the original rule schema set is used. Further calls are then used to find rules in which the new concept occurs as a premise. Each of these calls specifies one of the remaining predicates in the knowledge

base as conclusion and uses the special rule schema set for characterization (of the appropriate arity). Such a call is performed for all of the remaining predicates in the knowledge base. In the same fashion, it is of course also possible to use a data-driven learner such as FOIL (Quinlan, 1990) for characterization (as was done in the experiments in section 5), albeit without the possibility of reducing the search space ahead of time—we must let it search for rules about all predicates in the knowledge base and then select the ones that contain the new concept.

In our example, RDR found the following rules about *c1*:

$$\begin{array}{lcl}
 \text{parking_violation}(X) \rightarrow c1(X) & & \\
 \text{involved_vehicle}(X,Y) \ \& \ \neg \text{buckled_up}(Z,Y) \rightarrow c1(X) & & \\
 \text{involved_vehicle}(X,Y) \ \& \ \text{lights_necessary}(X) & & \\
 \quad \& \ \neg \text{headlights_on}(X,Y) \rightarrow c1(X) & \left. \vphantom{\begin{array}{l} \text{involved_vehicle}(X,Y) \ \& \ \text{lights_necessary}(X) \\ \& \ \neg \text{headlights_on}(X,Y) \end{array}} \right\} SC(c1) \\
 \text{unsafe_vehicle_violation}(X) \rightarrow c1(X) & & \\
 c1(X) \rightarrow \neg \text{tvr_points_p}(X) & \} & NC(c1) \\
 c1(X) \ \& \ \text{appeals}(Y,X) \rightarrow \text{court_citation}(Y) & \} & U(c1)
 \end{array}$$

where *SC*, *NC* and *U* denote sufficient conditions, necessary conditions, and uses of *c1*, respectively, as defined in section 3. This new concept *c1* actually corresponds to the notion of a “minor violation”: it includes parking violations and other small offenses that are not penalized very strongly. In a telecommunication security domain developed in cooperation with Alcatel Alsthom Recherche, Paris, CLT was used to introduce the concept of a “senior operator” that was needed to fix an overly general access control rule; in a satellite diagnosis domain that was developed in cooperation with British Aerospace, Bristol, CLT found a new concept that distinguished grounded from nongrounded circuit measurement points.

4.3. Evaluating the new concept

Since KRT and CLT are designed for interactive use, any new concept introduced by CLT is presented to the user for approval. The user can decide whether to entirely remove the new concept, to keep only its extensional definition (the membership facts), or to keep facts and learned rules. Keeping the facts only can be useful if the concept itself is meaningful in the domain but the found characterization is incorrect, for instance, because the knowledge base is still incomplete. In any case, the user can replace the system-chosen “gensym” name with a more appropriate domain-specific term. To help the user make these choices, the system presents the current list of members and nonmembers of the concept.

Furthermore, CLT uses an additional internal structural evaluation criterion. In interactive use, this criterion is used to provide an initial recommendation that the user can override; in noninteractive use, this criterion is relied upon completely as an additional structural filter on the concepts proposed by KRT. We have theoretically and empirically examined three different, and progressively weaker, evaluation criteria defined as follows:⁹

Strict criterion. Accept c iff $|SC(c)| \geq 2$, and $(|NC(c)| \geq 2$ or $|U(c)| \geq 1)$.

Default criterion. Accept c iff $|SC(c)| \geq 1$, and $(|SC(c)| + |NC(c)| + |U(c)| \geq 2)$.

Null criterion. Accept c in any case.

The first two criteria are motivated by general desirable properties of useful concepts, namely, that they be *recognizable*, i.e., have sufficient conditions, and that they be *predictive*, i.e., have necessary conditions or uses. The strict criterion, as we will see in section 5, is very strong and results in the introduction of very few concepts, but has interesting theoretical properties with respect to knowledge-base structure. The default criterion is less strict, but still requires sufficient and necessary conditions. In our experiment, it turned out to be still strong enough to guarantee that new concepts improve the accuracy of the learning system. The final criterion was proposed by Muggleton and Bain (1992), who introduce a new concept for every specialization. We employ it as the “control condition” in the empirical experiment of section 5.

Note that whereas the aggregation constraint used by CLT is derived from the theory revision context, the above evaluation criteria are structural constraints in the classification given at the beginning of this article. Indeed, the use of contextual constraints to trigger concept formation is orthogonal to the use of other constraints that can be used to judge the concepts so proposed. Also, since an additional inductive step is involved to find the rules upon which the criteria are based, these criteria indeed are an additional strong constraint. In the example given above, we see that

$$|SC(c_1)| = 4, |NC(c_1)| = 1, |U(c_1)| = 1,$$

so all three are met.

Since the knowledge base may change after the introduction of a concept, it is not guaranteed that the above evaluation criteria will continue to hold (e.g., a rule in which a concept was used could be deleted). To ensure that concepts that have turned out to be useless are removed from the knowledge base, CLT verifies after each rule modification or deletion whether the evaluation criteria for a CLT-introduced concept are negatively affected and removes the concept if this is the case. This process is also referred to as *concept garbage collection*.

4.4. Using the new concept to restructure the rule base

Whenever a new concept has been introduced and is approved by the internal evaluation criterion or by the user, CLT examines whether the knowledge base can be simplified by using the new concept in existing rules.¹⁰ This restructuring task consists of deriving via resolution all rules implied by the concept rules found during characterization and then removing all existing rules that are θ -subsumed by any of those resolvents (i.e., it is a simple redundancy check).

More precisely, let $SC(c)$, $NC(c)$, and $U(c)$ be defined as above. If

$$res(c, \Gamma) := \{R | \exists R_1 \in SC(c), R_2 \in NC(c) \cup U(c) : R = \text{resolvent}(R_1, R_2)\},$$

then RRT can be defined as follows: given a concept c in a knowledge base Γ , RRT produces the knowledge base

$$\Gamma' := \Gamma \setminus \{R \mid \exists \theta, R' \in \text{res}(c, \Gamma) \wedge R'\theta \subseteq R\}.$$

In our example, the original knowledge base had also contained the learned rules

```
involved_vehicle(X,Y) & ¬ buckled_up(Z,Y) → ¬ tvr_points_p(X)
involved_vehicle(X,Y) & lights_necessary(X) & ¬ headlights_on(X,Y)
    → ¬ tvr_points_p(X)
parking_violation(X) → ¬ tvr_points_p(X)
parking_violation(X) & appeals(Y,X) → court_citation(Y)
unsafe_vehicle_violation(X) & appeals(Y,X) → court_citation(Y)
```

which have all become redundant after the introduction of the new concept, and will be removed.

5. Evaluating MOBAL's concept formation method

As part of the MOBAL system, CLT is well suited for interactive use during knowledge revision. In practice, users find it relatively easy to judge a new concept, since the system shows the user the existing examples of the concept. Muggleton (1987) has reported similar experience with his DUCS system. In the interactive process of revising an incorrect knowledge base, CLT proposes new concepts that are *qua definitionem* likely to be relevant to the problem at hand. How much can be said, however, about the quality of CLT concepts independent of an interactive evaluation by the user?

A newly introduced concept influences the learning result along several dimensions: the learnability of the desired target concept(s), the classification accuracy of the rules that are learned, the speed of inference of the problem solver, and the structure and understandability of the resulting knowledge base. In the following, we will theoretically evaluate the three evaluation criteria defined in section 4.3 with respect to their effect on knowledge-base structure (section 5.1), and then report the results of empirical tests of their effect on classification accuracy (section 5.2).

5.1. Effects on knowledge-base structure

As for the effects of new CLT concepts on the structure of a knowledge base, Fu and Buchanan (1985) have already pointed out that a knowledge base consisting of compiled, single-step rules is not optimal in terms of understandability and robustness, and that the introduction of intermediate concepts, as is achieved by KRT/CLT, is preferable. Since we are also interested in the size of a knowledge base, however, we will now propose a measure of concept quality that balances knowledge-base size against the advantages of small chunk size.

Previous research (Muggleton, 1987; Muggleton & Buntine, 1988) has used the total size of the knowledge base as approximated by the number of premise and conclusion literals as a quality measure (call it *TS*); this, however, produces an unwanted effect in our context, as the following simple knowledge bases show:

$$\Gamma_1 := \left\{ \begin{array}{l} p(X) \& c(X) \rightarrow r(X) \\ a(X) \rightarrow c(X) \\ b(X) \rightarrow c(X) \end{array} \right\} \quad \Gamma_2 := \left\{ \begin{array}{l} p(X) \& a(X) \rightarrow r(X) \\ p(X) \& b(X) \rightarrow r(X) \end{array} \right\}$$

According to TS , Γ_2 is the preferred knowledge base, since $TS(\Gamma_2) = 6$, whereas $TS(\Gamma_1) = 7$. To avoid penalizing the introduction of new concepts in this fashion, while still incorporating a global measure of knowledge-base size, we therefore use a measure S that counts only *premise* literals. According to S , the two theories are equally good, since $S(\Gamma_1) = S(\Gamma_2) = 4$.

Definition 1. We define a quality ordering $>_q$ on knowledge bases such that

$$\Gamma_1 >_q \Gamma_2 \text{ iff } S(\Gamma_1) < S(\Gamma_2) \text{ or } (S(\Gamma_1) = S(\Gamma_2) \text{ and } C(\Gamma_1) > C(\Gamma_2)),$$

where $C(\Gamma)$ denotes the number of predicates in Γ .

In our example, we thus find $\Gamma_1 >_q \Gamma_2$. Note that the second and third evaluation criteria proposed above (“default” and “null”) do not ensure that knowledge-base quality improves along the above measure. Concepts that meet the strict criterion, however, always produce an improvement in $>_q$, as can easily be shown.¹¹

Let Γ_1 be a knowledge base into which CLT has introduced (perhaps at an earlier time) a new concept c ; let $\Gamma_1^H = SC(c)$, the set of all rules with head c . Similarly, let $\Gamma_1^B = NC(c) \cup U(c)$, the set of all rules with c in the body, and $\Gamma_1' := \Gamma_1^B \cup \Gamma_1^H$. Then define

$$\Gamma_2' := R(c, \Gamma_1) = \{R \mid \exists R_1 \in \Gamma_1^H, R_2 \in \Gamma_1^B : R = \text{resolvent}(R_1, R_2)\},$$

and $\Gamma_2 := \Gamma_1 \setminus \Gamma_1' \cup \Gamma_2'$. In other words, Γ_2 is the knowledge base that would result if c were removed. The following holds:

Theorem 1. For any Γ_1 and Γ_2 as defined above, if c meets the strict evaluation criterion of section 4.3, then $\Gamma_1 >_q \Gamma_2$.

Proof. The above can easily be verified by computing S for the new knowledge base with and without the new concept. In producing Γ_2 from Γ_1 , all rules in Γ_1^H are simply removed, and in each rule in Γ_1^B , C is replaced by its definitions, resulting in $|\Gamma_1^H|$ new rules each. It is thus

$$S(\Gamma_2') = |\Gamma_1^H| \cdot (S(\Gamma_1^B) - |\Gamma_1^B|) + |\Gamma_1^B| \cdot S(\Gamma_1^H).$$

Consequently, the S -difference between the two knowledge bases is

$$\begin{aligned} S(\Gamma_2) - S(\Gamma_1) &= S(\Gamma_2') - S(\Gamma_1') \\ &= |\Gamma_1^H| \cdot (S(\Gamma_1^B) - |\Gamma_1^B|) + |\Gamma_1^B| \cdot S(\Gamma_1^H) - S(\Gamma_1^B) - S(\Gamma_1^H) \\ &= |\Gamma_1^H| \cdot (S(\Gamma_1^B) - |\Gamma_1^B|) + |\Gamma_1^B| \cdot S(\Gamma_1^H) - (S(\Gamma_1^B) - |\Gamma_1^B|) - |\Gamma_1^B| - S(\Gamma_1^H) \\ &= (|\Gamma_1^H| - 1) \cdot (S(\Gamma_1^B) - |\Gamma_1^B|) + (|\Gamma_1^B| - 1) \cdot S(\Gamma_1^H) - (|\Gamma_1^B| - 1) - 1 \\ &= (|\Gamma_1^H| - 1) \cdot (S(\Gamma_1^B) - |\Gamma_1^B|) + (|\Gamma_1^B| - 1) \cdot (S(\Gamma_1^H) - 1) - 1 \end{aligned}$$

From the evaluation conditions imposed by CLT, we know that C has at least two sufficient conditions, and thus $|\Gamma_1^H| \geq 2$ and $S(\Gamma_1^H) \geq 2$. Also, since it is required that either the concept have at least two necessary conditions or that it be used in another rule (with other premises—for instance, in the rule that triggered the knowledge revision in the first place), we know that either (case a) $|\Gamma_1^B| \geq 2$ and $S(\Gamma_1^B) \geq |\Gamma_1^B|$ (if there are at least two necessary conditions for C), or (case b) $|\Gamma_1^B| \geq 1$ and $S(\Gamma_1^B) - |\Gamma_1^B| \geq 1$ (if C is used in another, longer rule). In case (a), we find that the difference in size between Γ_1 (including C) and Γ_2 (excluding C) is

$$\geq (2 - 1) \cdot 0 + (2 - 1) \cdot (2 - 1) - 1 = 0.$$

Similarly, in case (b), we find that the difference is

$$\geq (2 - 1) \cdot 1 + (1 - 1) \cdot (2 - 1) - 1 = 0.$$

The knowledge base including the new concept is thus guaranteed to be smaller or of the same size as the equivalent knowledge base without the concept, and since in any case, $C(\Gamma_1) > C(\Gamma_2)$, it is guaranteed that $\Gamma_1 >_q \Gamma_2$. $\square$

5.2. Empirical evaluation of effects on accuracy

To test the effect of the different evaluation criteria given above on the classification accuracy of the underlying learning system, we have performed a number of empirical tests. To test the generality of our technique for concept formation during theory revision, these tests were performed using Quinlan's FOIL¹² (Quinlan, 1990) as the underlying learning system. FOIL was used both to derive an initial set of rules and to perform the characterization step inside CLT. This way, unintended effects that could have been caused by different choices of RDT's model knowledge were avoided.

The experiments were performed in the kinship domain originally developed by Hinton (1986) and already used by Quinlan (1990). In the factual representation appropriate for MOBAL and FOIL, this domain consists of 112 positive and 6800 negative facts representing the examples. For each experiment, we randomly selected a different subset of these examples for training that included 90% of the positive examples and 5% of the negative examples. FOIL was then run on the training set, and the accuracy of learned rules was tested on the full set of examples for *aunt*. Whenever the FOIL-induced rules produced contradictions, i.e., false positives, we applied KRT/CLT in noninteractive mode, searching for unary and binary concepts if necessary according to the search strategy in table 4. Whenever a new concept was proposed, FOIL was called again to characterize the proposed concepts, as described in section 4.2. The characterization produced by FOIL was then evaluated according to one of the three criteria defined above, and if successful, the concept was introduced into the knowledge base. We then removed the rules produced in the revision process, keeping only the rules defining the new CLT concepts. In order to determine the effect of the newly formed concepts on learning accuracy independent of the information introduced during revision, FOIL was then run one more time on the original data, augmented only by the rules defining the new CLT concepts, and accuracy was measured on the examples for *aunt* again.

Table 5 summarizes the results of repeating this experiments 30 times for each of the criteria. The comparison was always made between FOIL on the original knowledge base vs. FOIL on the original knowledge base plus concept definition rules. Surprisingly, these results show that the strict evaluation criterion is much too strict in this domain and introduced only one concept. For the default criterion, CLT introduced one or more new concepts in 16 out of 30 runs. Among these, almost half (44% or seven runs) led to an improvement in accuracy; in another three runs (19%), the rules learned with CLT concepts had a better fit (fewer false positives), but reduced coverage, resulting in identical accuracy. In three runs, there was no difference, and in only 3 out of 16 runs, CLT concepts led to a decrease in accuracy when the default criterion was used. Following our expectations, the results with the control criterion were not as good, since the proliferation of new concepts leads FOIL to select many irrelevant rules. Without the additional structural check on new concepts, accuracy increased as often as it decreased. Figure 1 graphically depicts the results for the default criterion.

Table 5. Summary of experimental results.

Evaluation Criterion	Strict	Default	Null
Total number of runs	30	30	30
Concepts formed in	1	16	24
Among these:			
Better accuracy using CLT concepts:	0	7 (44%)	7 (29%)
Better fit, less coverage using CLT concepts:	0	3 (19%)	1 (4%)
Same accuracy using CLT concepts:	0	3 (19%)	9 (38%)
Worse accuracy using CLT concepts:	1	3 (19%)	7 (29%)

Figure 1. Detailed results obtained using the default criterion.

6. Related work

As mentioned above, Muggleton and Bain (1992) have also proposed the introduction of new concepts for expressing rule instance and exception sets as they arise during specialization. In contrast to our method, they extensionally define a new predicate containing a rule's exceptions whenever a rule needs to be specialized, that is, without an additional structural evaluation. To avoid overspecialization, Muggleton and Bain need to add the new concept to the rule as a nonmonotonically interpreted negated premise. This means the rule only may be applied whenever the bindings of its variables are not yet known to be members of the exception concept.¹³ Their method thus shared with ours the use of rule instances and exceptions for focusing aggregation, but does not try to characterize the newly introduced concepts, and thus does not perform any additional quality checking on the proposed concepts. As we have seen in our experiment, this can lead to a proliferation of new concepts without a corresponding increase in classification accuracy.

In a propositional representation, the usefulness of exploiting contextual constraints for introducing new terms has also been demonstrated in Utgoff's STABB system (Utgoff, 1986), an extension of LEX (Mitchell, 1982) capable of introducing new descriptors into LEX's context-free grammar representation language. Whereas in our method, the context consists of instances and exceptions arising during the specialization of general inference rules, in STABB the positive and negative instances of LEX's operator applications (as determined by LEX's critic module) are the basis for the introduction of new propositional terms. These are constructed as least disjuncts of existing descriptors, or by collecting all necessary conditions for the successful solution path (regression). The new term is added to the system's grammar vocabulary, but there is no separate characterization step that would search for a concept definition independent of the current operator context. Accordingly, an additional quality check on proposed new terms is not performed. There is also no means for removing a new term once it is no longer needed in the operator condition for which it was constructed.

Another way of looking at the aggregation constraint employed in MOBAL is that it responds to the need to *refer* to a certain set of objects for which no name is yet available in order to complete a particular task (in our case, the theory revision task). This is why we have often referred to this type of strategy as *demand-driven* concept formation (Wrobel, 1988, 1989). Such an aggregation constraint has also been applied in the concept formation system KLUSTER (Kietz & Morik, this issue), which forms concepts in a term subsumption representation language similar to KL-ONE. In KLUSTER, objects are first aggregated based on their common occurrence in the same argument place of a predicate; a similar but simpler aggregation rule has also been used in GLAUBER (Langley et al., 1986). The system then tries to differentiate and characterize these extensional aggregates, and if it needs to refer to a particular set of role fillers for which no existing concept is adequate, a new concept is introduced. KLUSTER thus does not use a task such as theory revision that is external to concept formation, but introduces new concepts as an integral part of its own concept formation activities.

The concept formation method presented here can also be contrasted with work on constructive induction. In constructive induction (see Rendell and Seshu (1990) for an overview), the learning system uses general or domain-dependent operators that examine the

existing set of descriptors or predicates and combine some of them into new descriptors. In contrast, a concept-formation approach such as ours examines the objects of the domain, decides which ones of them to aggregate, and then inductively characterizes them. In this separate inductive step, a definition for the new concept's predicate is found, that is, only those combinations of descriptors are acceptable that can be verified inductively in the knowledge base. As an example of a constructive operator that also works in a first-order representation, we can look at the intraconstruction operator from inverse resolution work (cf., for instance, Muggleton & Buntine, 1988). This operator introduces new predicates in order to group together varying parts of different clauses in a theory that otherwise share the same structure. A new predicate is formed and extensionally defined to cover the varying parts, while in the original clauses, the varying parts are replaced with the new predicate. There is no separate inductive step, however, to relate the new predicate to other knowledge in the system's theory. Thus, other measures, such as those proposed by Muggleton (1988), are necessary to prevent the introduction of uninteresting new predicates.

Finally, the idea of introducing new concepts in a theory revision system has also been pursued in the theory revision system *EITHER* by Ourston and Mooney (1990), albeit in a propositional representation only, and in a different fashion than in *MOBAL*. In *EITHER*, the system first performs specialization on a theory by using essentially the operators that were defined in table 2, augmented by an inductive operator that uses the instances and exceptions of the rule that is to be specialized as an extensional target set for concept learning. Here, however, *EITHER* only goes halfway and does not introduce a new, independently named concept—the learned condition is only incorporated into the rule locally. Thus, the theory revision constraint is not exploited for concept formation. Instead, after revision activity has been completed, *EITHER* uses inverse resolution as described in the preceding paragraph to simplify its theory, which may then lead to predicate introductions. There is also a first-order successor of *EITHER* called *FORTE* (Richard & Mooney, 1991); this system, however, apparently does not introduce new concepts.

7. Conclusion

In this article, we have regarded concept formation as a process of forming extensional aggregates of objects and then characterizing these aggregates with an intensional definition. If the so-formed concept is given a name and made available in the representation, concept formation is an answer to the new-term problem of machine learning. Our work has put the focus on the constraints that can be used to control the aggregation process and, in particular, on the use of context-based constraints arising from other parts of an integrated learning system.

The results reported above indicate that the theory revision activities of a learning system can constitute a powerful contextual constraint on the aggregation step of concept formation. The objects for which the application of a rule is valid, and those for which it is not, can serve as the extension of a proposed new concept. Especially in an interactive context, which is the usual case in *MOBAL*, this constraint ensures that new concepts are proposed in relation to the revision task currently being worked out. This helps users decide whether or not to keep a concept introduced by the system.

Nonetheless, in the experiments reported in section 5, we have also seen that in noninteractive mode, the contextual constraint alone can lead to a proliferation of new concepts that are not beneficial for classification accuracy. An important point of the method in MOBAL is therefore the use of an additional, orthogonal criterion based on structural properties of the proposed concept's intensional definition. This means that whenever a new concept is proposed from within theory revision, the learner first performs an additional inductive characterization step where it tries to induce additional rules about the members of the proposed new concept, and then judges the structural quality of the induced concept definition to decide whether to keep the concept. Without this step, the introduction of a new concept would be no more than the syntactic construction of a new term. With the characterization step, on the other hand, we ensure that a new concept has meaning beyond the theory revision context in which it was originally introduced. In the experiment with MOBAL's default criterion, this has led to corresponding improvements in classification accuracy.

The disadvantage of using additional structural criteria is mainly in their computational cost, since all potential rules about a new concept must be searched for, which can be expensive. Even though we have shown that our structural criteria have desirable effects on theory structure and accuracy, it is an interesting topic of future research to see whether other structural measures—for instance those based on information compression (minimal encoding length)—could be employed while maintaining the general principle of using a context-based constraint followed by a structural constraint. It is also clear that our method is only beginning to exploit the potential of using contextual constraints for concept formation in integrated learning systems. With a constraint based on theory revision, we can be sure only to find all those concepts that are needed in theory revision. An interesting topic for future research is thus the identification of other subtasks in MOBAL or other integrated learning systems that could serve as additional contexts for triggering concept formation.

Acknowledgments

This work was partially supported by the European Community ESPRIT program under contract number P2154 "Machine Learning Toolbox" (MLT), and under contract 6020, "Inductive Logic Programming" (ILP). MOBAL is or has been developed at GMD by Werner Emde, Roman Englert, Christian Haider, Jörg-Uwe Kietz, Volker Klingspor, Katharina Morik, Edgar Sommer, and the author. I want to thank all my colleagues, but especially Katharina Morik, for interesting discussions, important advice, and invaluable guidance. The very constructive comments of the editors and the anonymous reviewers have also greatly helped to improve the article. Parts of this article are taken from the forthcoming book *Knowledge Acquisition and Machine Learning* by K. Morik, S. Wrobel, J.U. Kietz, and Werner Emde, with kind permission of the publisher, Academic Press, London.

Notes

1. Our use of the term *concept formation* is thus more general than the definition used by Gennari et al., (1989), who included the additional requirement of forming a hierarchical organization, and also restricted the term to divisive, hill-climbing methods.
2. MOBAL also contains an interface to the CKRL language of the MLT system (Morik et al., 1991).
3. In this article, MOBAL's knowledge representation is described using the standard logical notation to improve readability; the system actually uses a syntactically different format.
4. In the following, we will usually omit the Γ subscript.
5. By default, the confidence in input facts is assumed to be 1.
6. For the use of local exception sets, see the definitions of operators (2) and (3) in tables 2 and 3.
7. Actually, there is a user-settable *plausibility criterion* that determines how many or what percentage of explicitly listed exceptions KRT will tolerate; as long as the result of operator (0) meets this criterion, no other operators will be applied.
8. In the implementation, c is a Prolog variable, so these special sets of rule schemata can be computed once ahead of time, using Prolog unification to replace c when they are used.
9. Before these criteria are applied, CLT also checks whether the new concept has turned out to be a synonym of an existing concept.
10. This operation is actually performed by a set of routines in the so-called ruleset restructuring tool RRT.
11. Only non-recursive new concepts are considered, since recursive concepts cannot be replaced by their definitions anyway. As correctly pointed out by an anonymous reviewer, a sequence of new concept introductions may lead to a local maximum along $>_q$, since the evaluation of a new concept will take previously introduced concepts into account. It is thus possible that the removal of *all* new concepts ever introduced will result in a better knowledge base.
12. We are thankful to R. Quinlan for making FOIL available to other researchers.
13. In effect, contrary to what is stated by Muggleton and Bain (1992), this is a syntactical variant of our support set representation with explicit exception lists (cf. also Wrobel, 1988; Wrobel, 1989).

References

- Barsalou, L.W. (1983). Ad hoc categories. *Memory & Cognition*, 11, 211-227.
- Easterlin, J. Daniel, & Langley, Pat. (1985). A framework for concept formation. *Seventh Annual Conference of the Cognitive Science Society* (pp. 267-271). Irvine, CA.
- Emde, Werner, Habel, Christopher U., & Rollinger, Claus-Rainer. (1983). The discovery of the equator or concept driven learning. *IJCAI-83* pp. 455-458. San Mateo, CA: Morgan Kaufmann.
- Emde, Werner. (1989). An inference engine for representing multiple theories. In Katharina Morik (Ed.), *Knowledge representation and organization in machine learning*. Berlin, New York: Springer Verlag, (pp. 148-176.)
- Everitt, Brian. (1980). *Cluster analysis*. Halsted Press: New York.
- Fisher, Douglas H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2, 139-172.
- Fu, Li-Min, & Buchanan, Bruce. (1985). Learning intermediate concepts in constructing a hierarchical knowledge base. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence* (pp. 659-666). San Mateo, CA: Morgan Kaufmann.
- Gennari, John H., Langley, Pat, & Fisher, Doug. (1989). Models of incremental concept formation. *Artificial Intelligence*, 40, 11-61.
- Hanson, Stephen J., & Bauer, Malcolm. (1989). Conceptual clustering, categorization, and polymorphy. *Machine Learning*, 3(4), 343-372.
- Hinton, G.E. (1986). Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Lawrence Earlbaum Associates.
- Kietz, Jörg-Uwe & Morik, Katharina. (1993). A polynomial approach to the constructive induction of structural knowledge. *Machine Learning* (this issue).

- Kietz, Jörg-Uwe, & Wrobel, Stefan. (1992). Controlling the complexity of learning in logic through syntactic and task-oriented models. In Stephen Muggleton (Ed.), *Inductive Logic Programming*, chapter 16. London, New York: Academic Press.
- Kietz, Jörg-Uwe. (1988). Incremental and reversible acquisition of taxonomies. *Proceedings of EKAW-88* (pp. 24.1–24.11). Also as KIT-Report 66, Technical University Berlin.
- Klingspor, Volker. (1991). Mlt deliverable 4.3.2/g: Mobal's predicate structuring tool. GMD, St. Augustine, FL.
- Langley, Pat, Zytkow, Jan M., Simon, Herbert A., & Bradshaw, Gary L. (1986). The search for regularity: Four aspects of scientific discovery. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine Learning* (Vol. II). Los Altos, CA: Morgan Kaufmann.
- Lebowitz, Michael. (1987). Experiments with incremental concept formation: UNIMEM. *Machine Learning*, 2, 103–138.
- Lloyd, J.W. (1987). *Foundations of logic programming*, 2nd edition. Berlin, New York: Springer-Verlag.
- Michalski, Ryszard S., & Stepp, Robert E. (1983). Learning from observation: Conceptual clustering. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), Palo Alto, CA: Tioga, pp. 331–363.
- Mitchell, Tom M. (1982). Generalization as search. *Artificial Intelligence*, 18(2), 203–226.
- Morik, K., Causse, K., & Boswell, R. (1991). A common knowledge representation integrating learning tools. In Ryszard S. Michalski & Gheorghe Tecuci (Eds.), *Proceedings of the First International Workshop on Multistrategy Learning (MSL-91)* (pp. 81–96). Fairfax, VA: George Mason University.
- Morik, Katharina, Wrobel, Stefan, Kietz, J.-U. & Emde, Werner, (1993). *Knowledge acquisition and machine learning: Theory methods and applications*. London, New York: Academic Press.
- Morik, K. (1991). Underlying assumptions of knowledge acquisition and machine learning. *Knowledge Acquisition Journal*, 3.
- Morik, Katharina. (1991). Balanced cooperative modeling. *Proceedings of the First International Workshop on Multistrategy Learning (MSL-91)* (pp. 65–80). Fairfax, VA: George Mason University. Also to appear in *Machine Learning*, 10.
- Muggleton, Stephen, & Bain, Michael. (1992). Non-monotonic learning. In Stephen Muggleton (Ed.), *Inductive logic programming*. London, New York: Academic Press.
- Muggleton, Stephen, & Buntine, Wray. (1988). Machine invention of first-order predicates by inverting resolution. *Proceedings of the Fifth International Conference on Machine Learning*. San Mateo, CA: Morgan Kaufmann.
- Muggleton, Stephen. (1987). Structuring knowledge by asking questions. In Ivan Bratko & Nada Lavrač (Eds.), *Progress in Machine Learning—Proceedings of the Second European Working Session on Learning (EWSL)* (pp. 218–229). Wilmslow, UK: Sigma Press.
- Muggleton, Stephen. (1988). A strategy for constructing new predicates in first order logic. In Derek Sleeman (Ed.), *Proceedings of the 3rd European Working Session on Learning* (pp. 123–130). London: Pitman.
- Nelson, Katherine. (1983). The derivation of concepts and categories from event representations. In Ellen Kosky Scholnick (Ed.), *New trends in conceptual representation: Challenges to Piaget's theory?* Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 129–149.
- Ourston, D., & Mooney, R.J. (1990). Changing the rules: A comprehensive approach to theory refinement. *Proceedings of the 9th National Conference on Artificial Intelligence* (pp. 815–820). San Mateo, CA: Morgan Kaufmann.
- Quinlan, J.R. (1990). Learning logical definitions from relations. *Machine Learning*, 5(3), 239–266.
- Rendell, Larry, & Seshu, Raj. (1990). Learning hard concepts through constructive induction: framework and rationale. *Computational Intelligence*, 6, 247–270.
- Richards, Bradley L., & Mooney, Raymond J. (1991). First-order theory revision. *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 447–451). San Mateo, CA: Morgan Kaufmann.
- Rosch, E., Mervis, C.B., Gray, W.D., Johnson, D.M., & Boyes-Braem, P. (1976). Basic objects in natural categories. *Cognitive Psychology*, 8, 382–439.
- Rosch, E., (1978). Principles of categorization. In E. Rosch & B.B. Lloyd (Eds.), *Cognition and categorization*. Hillsdale, NJ: Erlbaum, pp. 27–48.
- Smith, Edward E., & Medin, Douglas L. (1981). *Categories and concepts*. London: Harvard University Press.
- Stepp, Robert E., & Michalski, Ryszard S. (1986). Conceptual clustering of structured objects: A goal-oriented approach. *Artificial Intelligence*, 28, 43–69.
- Tversky, A. (1977). Features of similarity. *Psychological Review*, 84, 327–352.

- Utgoff, Paul E. (1986). Shift of bias for inductive concept learning. In R.S. Michalski, J.G. Carbonell, & T.M. Mitchell (Eds.), *Machine learning—An artificial intelligence approach* (Vol. II). San Mateo, CA: Morgan Kaufmann, pp. 107–148.
- Wrobel, Stefan. (1988). Automatic representation adjustment in an observational discovery system. In D. Sleeman (Ed.), *Proceedings of the 3rd European Working Session on Learning* (pp. 253–262). London: Pitman.
- Wrobel, Stefan. (1989). Demand-driven concept formation. In Katharina Morik (Ed.), *Knowledge representation and organization in machine learning*. Berlin, New York: Springer-Verlag, pp. 289–319.
- Wrobel, Stefan. (1991a). Concepts formation in man and machine: Fundamental issues. Arbeitspapiere der GMD 560, GMD (German National Research Center for Computer Science), P.O. Box 1240, W-5205 St. Augustin 1, Germany, 1991. Presented at the *Workshop on Concept Formation in Man and Machine*, May 1991.
- Wrobel, Stefan. (1991b). Towards a model of grounded concept formation. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence* (pp. 712–717). San Mateo, CA: Morgan Kaufmann.
- Wrobel, Stefan. (1993a). *Concept formation and knowledge revision—a demand-driven approach to representation change*. Doctoral dissertation, Universität Dortmund.
- Wrobel, Stefan. (1993b). On the proper definition of minimality in specialization and theory revision. *Proceedings of the Sixth European Conference on Machine Learning (ECML-93)*, pp. 65–82). Also as Arbeitspapiere der GMD No. 730.

Received February 2, 1992

Accepted March 17, 1993

Final Manuscript April 21, 1993