Finding New Rules for Incomplete Theories: Induction with Explicit Biases in Varying Contexts

Thesis Proposal

Andrea Pohoreckyj Danyluk
Department of Computer Science
Columbia University
New York, NY 10027

andrea@CS.COLUMBIA.EDU

30 March 89

CUCS-466-89

Table of Contents

List of Figures

Abstract

Many AI problem solvers possess explicitly encoded knowledge - a domain theory - that they use to solve problems. If these problem solvers are to be autonomous, they must be able to detect and to fill gaps in their own knowledge. The field of machine learning addresses this issue. Recently two disparate machine learning approaches have emerged as predominant in the field: explanation-based learning (EBL) and similarity-based learning (SBL).

EBL and SBL have been applied to problems in a variety of domains. Both methods have clear problems, however. EBL assumes that a system is given an explicit theory of the domain that is complete, correct, and tractable. These assumptions are clearly unrealistic for most complex, real-world problems. SBL suffers because of its lack of an explicit theory of the domain. The simplicity of the method requires that human intervention play a large role in tailoring input examples and the features describing them in such a way as to allow a system to choose an appropriate set of features to define a concept. Biasing a system in this way may result in its being unable to discover all concepts in even a single domain. Less tailoring of the examples leaves a system open to the possibility of not converging on the best definition for a concept, or any at all, due to the computational complexity.

The research described in this proposal addresses a number of the problems found in explanation-based and similarity-based learning. The major focus of the research is the elimination of the assumption that the domain theory of an EBL system is complete. In particular, it considers the problem of working with an incomplete theory by suggesting a method by which gaps in an EBL system's knowledge can be detected and filled. We suggest that when EBL cannot derive a complete explanation, the partial explanation forms a context in which learning takes place. Information extracted from partial explanations, as well as from complete explanations, can be exploited by SBL to do better induction of the missing domain knowledge. The extracted information constitutes an explicit bias for similarity-based learning. A second problem to be addressed is that of making the biases of SBL explicit. Finally, all testing of the claims made in this proposal is to be done in the Gemini learning system. The development of the system addresses the goal of constructing an integrated learning architecture utilizing both EBL and SBL.

1 Introduction

Many AI problem solvers possess explicitly encoded knowledge - a domain theory - that they use to solve problems. If these problem solvers are to be autonomous, they must be able to detect and to fill gaps in their own knowledge. The field of machine learning addresses this issue. Recently two disparate machine learning approaches have emerged as predominant in the field: explanation-based and similarity-based learning.

Explanation-based learning (EBL) is a deductive machine learning approach in which a definition of a concept is learned, usually after observing only a single example of that concept. The basic goal of an EBL system is to more efficiently recognize concepts that it is already capable (at least in theory) of recognizing. The learning process involves a knowledge-intensive analysis of an environment-provided example of a concept in order to extract its characteristic features. The analysis is in the form of a proof explaining why the particular instance is a member of the concept to be learned. A general concept definition is formed by generalizing the input example in a manner consistent with the proof. The proof thus provides a justification for the generalization.

Similarity-based learning (SBL) is an empirical technique that involves the comparison of a large number of input examples. These are compared in order to find shared features that are assumed to define a concept. The basic goal of SBL is to acquire descriptions that will allow a system to recognize concepts it does not yet know.

EBL and SBL have been applied to a variety of domains. Both methods have clear problems, however. EBL assumes that a system is given an explicit theory of the domain that is complete, correct, and tractable. These assumptions are clearly unrealistic for most complex, real-world problems. SBL suffers because of its lack of an explicit theory of the domain. The simplicity of the method requires that human intervention play a large role in tailoring input examples and the features describing them in such a way as to allow a system to choose an appropriate set of features to define a concept. Biasing a system in this way may result in its being unable to discover all concepts in even a single domain. Less tailoring of the examples leaves a system open to the possibility of not converging on the best definition for a concept, or any at all, due to the computational complexity.

The research proposed here will address a number of the problems found in explanation-based and similarity-based learning. The major focus of the research is the elimination of the assumption that the domain theory of an EBL system is complete. In particular, it considers the problem of working with an incomplete theory by suggesting a method by which gaps in an EBL system's knowledge can be detected and filled. We suggest that when EBL cannot derive a complete explanation, the partial explanation forms a context in which learning takes place. Information extracted from partial explanations, as well as from complete explanations, can be exploited by SBL to do better induction of the missing domain knowledge. The extracted information constitutes an explicit bias for similarity-based learning. A second problem to be addressed is that of making the biases of SBL explicit. All testing of the claims made in this proposal is to be done in the Gemini learning system. The development of the system addresses the goal of constructing an integrated learning architecture utilizing both deductive and inductive methods.

Section 2 introduces background information about explanation-based and similarity-based learning, their limitations, and previous efforts to address the limitations. Section 3 discusses the problems to be addressed by the research, and the hypotheses that led us to our proposed solution to the problems. Section 4 describes our solution to the problems as well as a proposed method for verification of the solution. Section 5 lists the contributions of the research. We conclude in Section 6 with a discussion of the work completed to date and a schedule for completion of the research.

2 Background

2.1 An Overview of Explanation-Based and Similarity-Based Learning

2.1.1 Explanation-Based Learning

Explanation-based learning (EBL)[1] (e.g., [DeJong 81; DeJong 83; DeJong and Mooney 86; Mitchell et al. 86; Silver 86; Winston et al. 83]) is a deductive machine learning approach in which a definition of a concept is derived, usually after observing only a single example of that concept. To understand this method on an intuitive level, consider a robot operating in a room containing household objects. In order to manipulate the objects in its world, the robot must have a mechanism for recognizing them. For example, it might require the ability to recognize cups.

Consider a robot that has been provided with a knowledge base of rules describing objects in its world and their characteristics. The rule base would be the domain theory as it constitutes the robot's knowledge of its domain of application. Some rules from this domain theory might be:

$$\mathrm{LIGHT(X)} \wedge \mathrm{PART\text{-}OF(X,Y)} \wedge \mathrm{ISA(Y,HANDLE)} \Rightarrow \mathrm{LIFTABLE(X)}$$

that states that if an object is light and has a handle then it is liftable, and:

$$\mathrm{ISA(X,OPEN\text{-}VESSEL)} \wedge \mathrm{STABLE(X)} \wedge \mathrm{LIFTABLE(X)} \Rightarrow \mathrm{ISA(X,CUP)}$$

[1] EBL is sometimes referred to in the machine learning literature as explanation-based generalization (EBG).

that states that if an object is an open vessel that is liftable and stable then it is a cup.[2] Now assume that the objects in the world are represented by primitive predicates that describe basic structural features of the objects, such as FLAT and LIGHT. Because the domain theory relates these primitive predicates to higher level predicates such as CUP, it can be used by the robot to deduce that a particular object is or is not a cup. The deductive process, generally implemented by a theorem-prover, is computationally expensive, however. A major goal of explanation-based learning is to improve the performance of such systems.

Input to EBL is the name of the concept to be learned, an instance of it, and a domain theory to be used to prove that the input instance is an example of the given concept. In the robot domain described above, the concept to be learned might be ISA(X,CUP). The example of a cup might be the representation of a particular blue ceramic cup: a conjunction of primitives such as COLOR(obj1,BLUE), MADE-OF(obj1,CERAMIC), etc. The domain theory would include rules such as those given above.

The goal of EBL is to create a general concept description based upon the predicates describing the particular example, such that the general description is justified by the proof derived to show that the example was a member of the concept to be learned. The first step of the learning process is to use the domain theory to generate a proof. The proof is sometimes called an explanation. The explanation for the input described in the preceding paragraph is shown in Figure 1.[3] Note that not all predicates used to describe the input example are actually found in the explanation. It is assumed that these, for instance the color of the cup, are not relevant to the definition of cup.

The next step of EBL is to maximally generalize the description of the input example with the constraint that the derived proof structure still holds. Many constants appearing in the proof can be made variables as shown in Figure 2. In some cases the generalization must be constrained. For example, OPEN-VESSEL was not generalized.

The general description of a cup, deductively validated by the proof derived, is shown in Figure 3. It can now be used by our robot to efficiently recognize cups, because the robot can now simply apply a single rule rather than having to construct a proof. (It has more recently been shown by [Minton 88] that there are cases in which the application of a single rule is less efficient than proof construction due to the complexity of determining that the rule applies.) In domains such as that of solving algebraic equations, where proofs can resemble traces of

[2] These examples are modified from Mitchell *et al.*'s paper on a unifying framework for the method of explanation-based learning [Mitchell et al. 86]. All examples from this domain presented here are based upon examples from their paper.

[3] Modified from [Mitchell et al. 86], page 59.

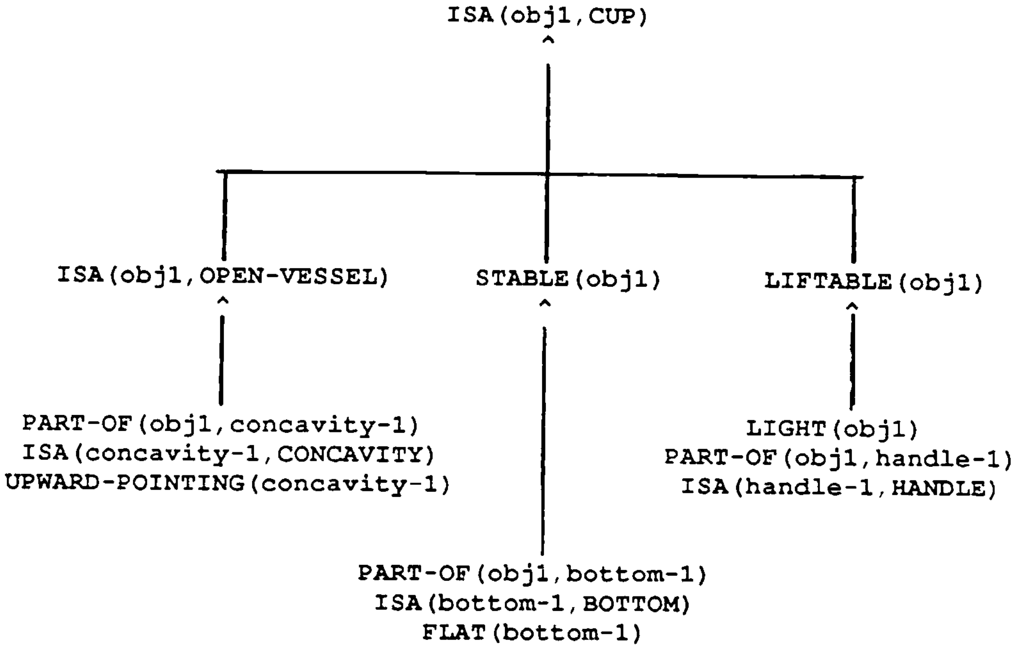


Figure 1: Explanation of Cup

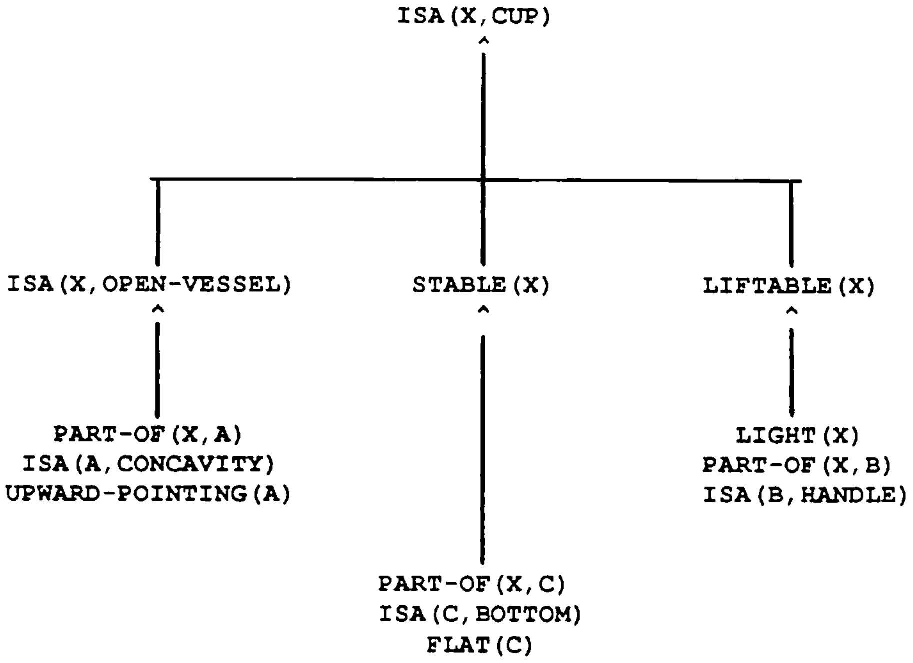


Figure 2: Generalized Explanation of Cup

problem solving steps, the generalized proofs may be stored to be applied automatically to analogous examples later. The result of explanation-based learning as described here is not to have learned a concept that could not have been recognized, but to learn a more efficient definition of such a concept.

```
PART-OF(X,A) ∧ ISA(A,CONCAVITY) ∧ UPWARD-POINTING(A)

∧

PART-OF(X,C) ∧ ISA(C,BOTTOM) ∧ FLAT(C)

∧

LIGHT(X) ∧ PART-OF(X,B) ∧ ISA(B,HANDLE)

⇒ ISA(X,CUP).
```

Figure 3: General Definition of Cup

The discussion above is intended to give an intuitive view of explanation-based learning. A more formal discussion is beyond the scope of this paper. (But see [DeJong and Mooney 86; Mitchell et al. 86]; a survey of EBL systems is presented in [Ellman 89].) The key ideas are: Explanation-based learning is a method in which a single example is analyzed and then generalized by a system. In order to perform the analysis, a system must possess a domain theory that is complete and correct, and the process of analysis must be tractable. The analysis, or derivation of an explanation, will often be performed by a deductive inference mechanism such as a theorem-prover. Depending upon the domain, an explanation may be a deduction of the flavor in the example above, or, more generally, any transformation of an initial problem state to a goal state. Generalized explanations, in addition to learned general definitions, may be saved so that they need not be re-derived. A summary of EBL is presented in Figure 4.

```
INPUT:    name of a concept to be learned;
          specific example of the concept;
          domain theory.

OUTPUT:   general (efficient) definition of the concept;
          generalized explanation.

METHOD:   1.  derive a proof that the example is an instance of
              the concept.
          2.  maximally generalize the proof while maintaining
              its correctness.
          3.  extract a general definition from the generalized
              explanation.
```

Figure 4: Summary of Explanation-Based Learning

2.1.2 Similarity-Based Learning

Similarity-based learning (SBL) (e.g., [Fisher 87; Lebowitz 87; Michalski and Stepp 83; Mitchell 78; Quinlan 86; Winston 72]) is an empirical technique that involves comparisons among large numbers of input examples. The input examples are compared in order to find similarities and differences among them. Similarities are generally assumed to define a useful concept.

To get a flavor of the method, consider the robot world described in the preceding section. We have a robot operating in a room containing household objects. Again the robot must be able to recognize objects in this room, such as cups. Unlike in the EBL setting, our robot does not have a domain theory relating predicates describing basic structural properties of the objects to higher level concepts. It must learn the descriptions of the objects "from scratch".

Input to the learning algorithm is the name of the concept to be learned, for instance CUP, and examples of that concept. Additional input to the algorithm might be examples of objects that are not members of the concept. In Figure 5 we show sample input examples to an SBL system that is to learn the concept CUP. Here we use structural properties that are the same as those used in the preceding section to describe the inputs. The goal of SBL is to create a general description of the concept that, so far as is possible, includes all the positive examples and excludes all the negative examples. The most obvious algorithm for doing so is to compare the input and find the largest set of descriptive features shared by all of the positive examples but not by any of the negative examples. A general description for the CUP concept based upon the input shown in Figure 5 is given in Figure 6.

FEATURE:	CUP-1	CUP-2	NON-CUP-1
COLOR	blue	red	blue
OWNER	Andrea	Anne	Harry
MADE-OF	ceramic	ceramic	ceramic
FLAT-BOTTOM?	yes	yes	yes
CONCAVITY?	yes	yes	yes
HANDLE?	yes	yes	no

Figure 5: Sample Input to SBL

The intuition behind the method is that the features essential for defining "CUP-ness" must appear in all examples of cups. The features that are not relevant to defining the concept, such as color, will vary among the examples and thus not be included in the general description. The version of SBL presented here is a simple-minded one. Often SBL systems deal with more complex representations, disjunctive definitions, and other generalizations of

	CUP
MADE-OF	ceramic
FLAT-BOTTOM?	yes
CONCAVITY?	yes
HANDLE?	yes

Figure 6: General Definition of Cup Found by SBL

that presented above.

Though they do not possess the explicit domain theories of EBL systems, SBL programs do have an implicit bias built into them that allows them to arrive at particular general concept definitions [Mitchell 80; Utgoff 86]. This bias includes, among other things:

- the language in which input examples and the learned concepts are represented.
- concept hierarchies that define relationships among the values that can be taken on by features.

Bias built into SBL systems is called inductive bias, because the generalization performed by such systems is analogous to that of inductive reasoning, or the reasoning from particular instances to a general conclusion.

Our example of similarity-based learning belongs to just one variety of methods that are collectively called SBL. In it a set containing both positive and negative examples was input to the system. There exist methods that take as input only positive examples. There also exist methods that do not receive all examples simultaneously but that incrementally build a description by considering one example at a time. These methods are collectively called **learning from examples** and include the work of [Mitchell 78; Winston 72] among others. Another variation of SBL does not take a concept name as input. Instead, many examples of a number of concepts are presented to the system simultaneously. The system creates sets from the input examples such that the elements of each set share a number of descriptive features not shared by the members of any other set. Work on this method of **conceptual clustering** includes that of [Fisher 87; Lebowitz 87; Michalski and Stepp 83]. Although these methods vary, they share the common intuition that the discovery of commonalities among members of a class may lead to a description for that class that will have predictive power in analyzing future examples.

As with the overview of explanation-based learning above, we will not attempt to present a more formal treatment of similarity-based learning here. A summary of SBL is presented in Figure 7.

```
INPUT:    name of the concept to be learned;
          specific examples (positive and/or negative)
            of the concept.

OUTPUT:   general definition of the concept.

METHOD:   compare examples to find similarities among positive
          examples and differences of positive examples from
          negative.
```

Figure 7: Summary of Similarity-Based Learning

2.2 Limitations of Explanation-Based and Similarity-Based Learning

Explanation-based and similarity-based learning methods are approaches that can be placed on opposite ends of a spectrum describing purely deductive to purely inductive techniques [Mitchell 84]. In explanation-based learning, analysis of a single example guides the generalization of knowledge possessed by a system in order to make itself more efficient. Its power derives from having extensive domain knowledge. The dependence upon the domain theory is also the weakness of the method. The theory is assumed to be complete and correct and the explanation derivation process to be tractable. These assumptions are clearly unrealistic in complex, real-world domains. Similarity-based learning systems do not possess the extensive, explicit knowledge bases of EBL. They must instead have extensive explicit example bases which guide them in a search through all possible concepts representable in a given language. Problems may arise if the inductive bias is not sufficiently strong guiding the search or if the input data is noisy. A more fundamental problem is the possibility that the concept to be learned cannot be adequately represented in the language of the system. The following section lists the limitations that we propose to address.

2.3 Issues to be Addressed by the Proposed Research

The primary focus of the research proposed here is the incomplete theory problem of explanation-based learning. That is, we address the situation where EBL's assumption of a complete domain theory does not hold. We propose that a modified version of similarity-based learning be used to fill the gaps in a domain theory. We call the implementation of SBL "modified" as it will exploit contextual information gleaned from partial (and complete) explanations derived by EBL. A set of heuristics that make use of contextual clues bias the learning of SBL. Thus a secondary emphasis of the research is that of making explicit the inductive biases of SBL. The research will be implemented within Gemini, a system designed as a general integrated learning architecture.

2.4 Related Work

Rajamoney and DeJong [Rajamoney and DeJong 87] identify two ways in which domain theories used by EBL systems can lack knowledge. In the first, an explanation, or proof, cannot be completed by the EBL system because knowledge, usually in the form of a rule, is missing from the domain theory as illustrated in Figure 8. In order to complete the explanation, a rule or fact must be added to the domain theory. In the second case, proofs can be constructed leading to a conclusion, but they lack the detail required for a particular application. This problem may be seen as one of a wrong choice of granularity of the knowledge represented. We view Rajamoney and DeJong's second description of incompleteness as one of incorrectness of the domain theory. It will not be discussed here.

```
  description
                     ——> A
      of                          . . . .    C ——> concept to be learned
                     ——> B
 input example

               . . . . = missing rule to deduce C from A and B.
```

Figure 8: The Problem of Incompleteness

2.4.1 Rajamoney

Rajamoney [Rajamoney et al. 85; Rajamoney 88; Rajamoney 89] proposes a method called **experimentation-based theory revision**, implemented in the ADEPT system, as a solution to the problem of incompleteness in a domain theory. Recently, Michalski and Ko [Michalski and Ko 88] have also discussed the use of experimentation in addressing this problem. The discussion here focusses on Rajamoney's work.

ADEPT's domain theory is represented using Forbus's Qualitative Process theory [Forbus 84]. According to this theory, changes in the world such as boiling or evaporation are due to *processes*, which in reasoning may be used analogously to rules. Processes are composed of three parts:

1. individuals - a set of objects that participate in the process,
2. preconditions and quantity conditions - a set of conditions that must be satisfied if the process is active,
3. relations and influences - a set of statements about the world that must be true if a process is active.

Preconditions and quantity conditions are analogous to the *if* part of an *if-then* rule, while relations and influences are analogous to the *then* part. ADEPT's goal is to use processes in explaining viewed changes in the world.

Rajamoney presents three ways in which incompleteness can become evident in a domain theory of processes:

1. an active process, i.e., one whose preconditions are satisfied, has flawed influences and is, in fact, causing the phenomenon that ADEPT is trying to explain. Essentially, this means that the phenomenon should be in the *then* part of an existing active rule (or should be deducible from a chain of active rules), but is not.
2. an inactive process that could explain the phenomenon if active has flawed preconditions or quantity conditions and should be active. In other words, a rule in which the phenomenon appears in the *then* part should have fired, but did not. Its preconditions, or *if* part, must be modified so that the rule may become active.
3. a new process is causing the phenomenon. No modification of an existing rule is sufficient to complete the explanation.

ADEPT proposes new rules when it is unable to complete an explanation for a viewed change in the world. The types of rules proposed reflect the first two of the descriptions of inadequacy just discussed. That is, an existing active rule can be modified to include the viewed change in its *then* part, or an existing inactive rule can be modified so that its preconditions are satisfied. These constrain the number of rules that are proposed. Proposed rules are empirically tested for validity. The testing performed by ADEPT is not SBL per se, but rather empirical validation. We include this discussion of ADEPT, however, because the underlying premise of SBL is empirical validation. That is, SBL makes the assumption that patterns observed over many situations will continue to be observed. ADEPT assumes that if the proposed rules accurately explain other observed changes in the world then they will continue to apply.

In order to minimize dependence upon a user, ADEPT tests its proposed rules by designing experiments so that the results of some of them will be inconsistent with a number of the proposed rules. These rules are eliminated from consideration. In an example taken from [Michalski and Ko 88], a system is given the task of explaining why a wine bottle placed in a freezer shattered. In the absence of more specific information, the system might propose that cold causes glass to contract while the volume of the contents of the glass remains the same. Alternately the liquid might expand. An experiment could be devised in which a glass container of water was placed in a freezer. If the container did not break, the first proposed rule would be invalidated.

Presumably ADEPT must possess a theory of experiment design for the domain to which it is applied. Given that good experiment design is a non-trivial problem, such a theory must be complex and would therefore be subject to the problem it is supposed to address, that of incompleteness. Furthermore, since the experiments would not exhaustively test all situations, a rule accepted by the system as appearing to hold in the experimental case might be incorrect. For instance in the example given above, the container might break despite the fact that the proposed rule is wrong.

A positive aspect of the work is that the number of newly proposed rules is constrained. This is important because it would be computationally infeasible to test exponential numbers of rules for validity. It is not clear, however, that the rule proposition mechanism is constrained sufficiently.

2.4.2 Hall

Hall describes a method for learning new rules called **Learning by Failing to Explain** [Hall 86]. In this method, a system that fails to find an explanation for an input example is given a new, analogous example by a teacher. The system learns by analyzing the analogue and comparing its explanation using similarity-based methods to the incomplete explanation.

Hall's system works in the domain of logic circuit design. Given as input a function name and an implementation of that function as a logic circuit, the system's goal is to prove that the implementation given is correct. Rules in the domain theory of the system have the form LHS ⇒ RHS, where LHS denotes a function, such as *PLUS*, and RHS is the description of an implementation for the LHS. The RHS may refer to other functions as well as to specific circuit implementations. For example, as shown in Figure 9, a function F0 might be decomposed into three subfunctions F1, F2, and F3, which might be implemented by circuits x, y, and z, respectively. The tree given in the figure is a proof that the implementation is correct. Rules used to construct the proof might be:

```
F0 ⇒ F1 F2 F3
F1 ⇒ x | q | r | s , where | indicates disjunction
F2 ⇒ y
F3 ⇒ z
```

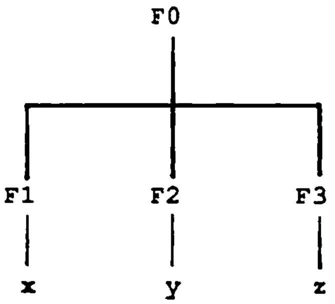


Figure 9: A Proof Tree for Logic Circuits

In Hall's system an explanation failure arises when the system is unable to prove that an input structure implements a given function. In the above example, this would occur if the system were missing the rule F1 ⇒ x | q | r | s as depicted in Figure 10. When the system signals a failure, a teacher provides an example from which the system can learn the rule it needs. It is assumed that in the failed proof the system is unable to link at most one subfunction to structures within the given implementation. That is, at most one rule is missing from the proof and it directly links the unexplained subfunction to features of the implementation.

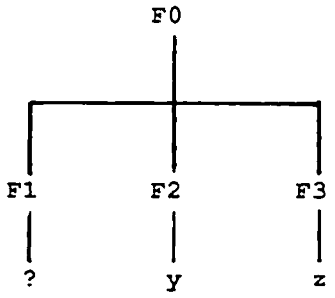


Figure 10: A Failed Proof for Logic Circuits

The teacher gives the system a new input structure that

1. implements the same function as that given to the system initially and that
2. contains as a subfunction the one for which the proof could not be completed.

A teacher-provided analogue to the example given above might be the one for which we give a proof in Figure 11.

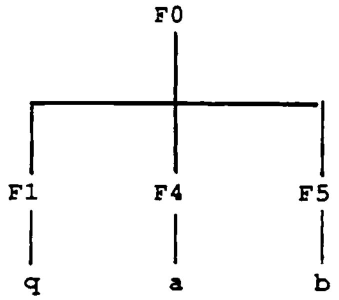


Figure 11: Proof for a Teacher-Provided Analogue

The system derives as complete an explanation as possible for the new input. Assuming that all rules linking a specific circuit implementation to subfunction F1 are missing, a partial proof for the analogue would look like that in Figure 12. Next, corresponding parts of the two implementations are matched, where matching is defined as functional equivalence - in essence, the proofs are matched. In our example, the system might determine that F2 and F4 are functionally equivalent, as well as F3 and F5. The system assumes that corresponding unmatched parts are equivalent to each other, i.e., that they both implement the same function. In the example above, a and y match since F2 and F4 are functionally equivalent; similarly, b and z match. The yet unmatched parts, q and x, are assumed to both implement the unproved function F1. Rules corresponding to the separate implementations are created and generalized. For the example above, the system would learn the rule: F1 ⇒ x | q.

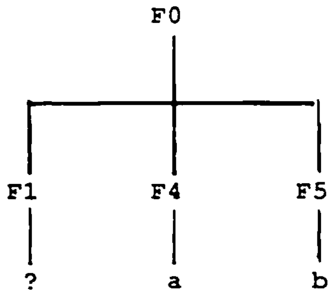


Figure 12: Partial Proof for Teacher-Provided Analogue

Hall's system works under the optimistic assumption that its domain theory will be very nearly complete. This must be the case if at most one subfunction may be unproved in the explanation of an entailing function. Hall's method is dependent upon a user not only for this nearly complete theory, but for training examples from which to learn, which must be specially tailored to the failure situation. Fortunately the input provided by the teacher does place strong constraints on the number of new rules proposed by system. Recall that in Rajamoney's work the number of new rules proposed, although somewhat constrained, could be potentially large.

2.4.3 Pazzani

Pazzani, in his system OCCAM [Pazzani et al. 86; Pazzani 87; Pazzani et al. 87; Pazzani 88], which predicts and explains the outcome of events, uses general knowledge about causality to propose cause/effect rules to be added to an incomplete domain theory. Pazzani states a clear preference for knowledge-based methods over empirical ones. He believes that EBL is always to be preferred to SBL and that a system should only fall back on SBL as a last resort. In addition to the domain theory used by EBL, he provides OCCAM with a base of *generalization rules* that function as templates for new rules to be created.

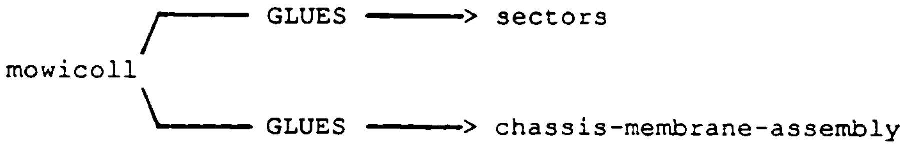
Input to OCCAM is the description of an event. OCCAM's goal is to predict an outcome for it. The chain of reasoning from the description of the event to the predicted outcome is a proof that the outcome will occur. A failure occurs if a proof cannot be derived that links aspects of the event to a predicted result. When this happens, OCCAM instantiates a generalization rule that will complete an explanation. Generalization rules encode information reflecting a theory of causality, but are otherwise independent of a domain. In theory they could be used by a performance system that predicted the outcome of meteorological events as well as by another that predicted the outcome of chemistry experiments. A typical example of a generalization rule is: "If an action on an object precedes a state change for the object, then the action may cause the state change." Given two examples, one in which a balloon could not be blown up and one in which it was first stretched and then blown up, OCCAM would propose a rule that stated that stretching a balloon causes it to be in a state from which it can be blown up. Rules are tentatively proposed by OCCAM as the generalization rules are not guaranteed to be instantiated correctly in all cases. For example, if a balloon were placed in water before being blown up, OCCAM would propose a rule that stated that dipping the balloon in water caused it to be blown up later. Rules are validated empirically as more examples are seen by the system. Thus, at any time, OCCAM's rule base may be incorrect.

A major strength of OCCAM is that the rules proposed by the system are constrained by the base of generalization rules which are applicable across a number of domains. However, they are specific to domains involving causality. It is not clear that an analogous base of rule templates could be devised for, say, Hall's logic circuit domain. Unlike Hall's system, however, Pazzani's is not dependent upon the presence of a teacher.

2.4.4 Kodratoff and Tecuci

DISCIPLE [Kodratoff and Tecuci 87; Kodratoff 87] uses an incomplete domain theory to pose questions to a user who then teaches the system missing rules. DISCIPLE has been applied to the domain of designing technologies for the manufacture of loudspeakers. A typical explanation is a plan for the manufacture of a loudspeaker. When the system is unable to complete a plan because knowledge is missing from the domain theory, a user supplies a rule that will complete the plan. DISCIPLE uses a combination of explanation-based and similarity-based learning to generalize that rule, thereby making it more widely applicable.

DISCIPLE begins the process of rule generalization in explanation-based mode, trying to explain why the rule given by the user is valid. Suppose that during the phase of constructing a plan, DISCIPLE had needed to *ATTACH sectors ON chassis-membrane-assembly*. Suppose that a user provided the following as a method to achieve that goal: *APPLY mowicoll ON sectors, PRESS sectors ON chassis-membrane-assembly.*[4] It is not assumed that the domain theory can provide a complete explanation, but only that it contains enough information to propose partial explanations. A best partial explanation is selected by the teacher. In the example described above, a user selected



as the best explanation proposed by DISCIPLE. Next DISCIPLE enters an SBL phase in order to generalize the rule. It searches the domain theory for information that matches the graph edges of the explanation selected. For example, the following information was extracted from

[4] From [Kodratoff 87], page 27 1. *Mowicoll* is roughly translated as *Romanian glue.*

DISCIPLE's domain theory as being similar to the explanation above:

```
           /----- GLUES -----> centering-device
          /
neoprene
          \
           \----- GLUES -----> chassis-assembly
```

Before the matching and generalization of SBL can proceed, the user must validate all examples as being similar and relevant in the current context.

Kodratoff and Tecuci's system relies heavily upon interaction with a teacher, who gives the system the information it is missing. Rather than counting on the user to provide a general rule, however, it asks for one appropriate for a particular situation and uses explanation-based and similarity-based techniques to guide the user in providing it with information sufficient for generalization of the rule. Constant supervision by the user can stop DISCIPLE from wasting computational resources considering irrelevant examples and assures correctness of the learned rules. This, however, places a heavy burden on the teacher.

3 Proposed Research: Problem and Issues

3.1 Statement of the Problem

As introduced briefly above, we propose to study the problem of performing explanation-based learning in the absence of a complete domain theory. More specifically, we will address the problem of detecting and filling gaps in the knowledge used by an EBL system.

3.2 Hypotheses

In this section, we describe observations that lead us to hypothesize the claims of the proposed research. We believe that the hope for a perfectly engineered theory for any complex domain is unrealistic. In the absence of human intervention, a system must be able to detect and to fill gaps in its own knowledge. The use of an auxiliary domain theory is one possible mechanism for filling gaps. However, we will, in many cases, be unable to rely on the auxiliary theory's having been perfectly engineered. That is, the use of an auxiliary domain theory simply shifts the problem to another level, as the auxiliary theory may be viewed as a part of the actual domain theory. We believe that, ultimately, a system must rely on "weak" learning methods such as SBL that can function without explicit domain knowledge. Although we believe that weak methods are ultimately necessary, we also believe that knowledge should be combined with them whenever possible.

Consider a situation in which an EBL system is given input consisting of the name of a concept "CUP" and an example of that concept, as in Section 2.1.1 above. Say that in attempting to show that the example is an instance of the concept the EBL system is only able to construct the partial explanation shown in Figure 13. This would arise if the domain theory contained no knowledge that would allow liftability to be deduced from the features describing the cup. If there were only one way to prove that the example was a cup, then the gap would fairly easily be detected by backward chaining from the goal concept. If, on the other hand, there were multiple competing explanations we could rely on heuristic means for finding the gap as in [Fawcett 89]. Now say we know that the concepts of openness, stability, and liftability are independent in that they do not share sets of features from which they are deduced. Then an SBL system wishing to link features of the cup to the subgoal of liftability does not have to consider any of the features used to explain the concepts of openness and stability. That is, in creating a rule (or rules) that would allow liftability to be inferred from features of the example, certain features could be ignored. In essence, SBL could learn a new rule *in the context of the partial explanation that the rule was to complete.* In this simple case, the partial explanation itself is the context, and a certain type of information - the features already used - may be extracted from it.

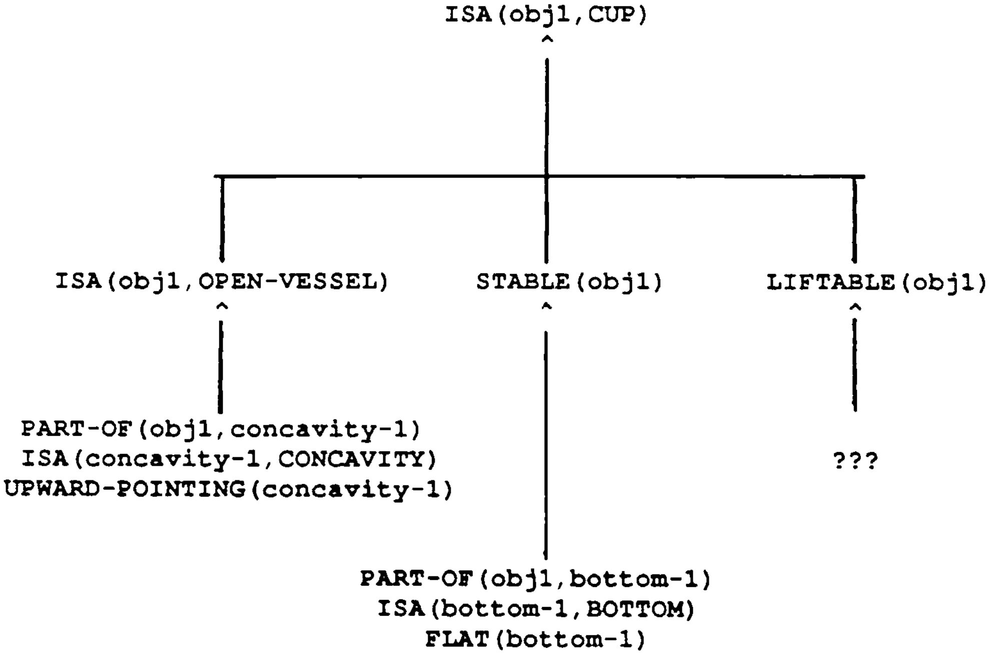


Figure 13: Partial Explanation of a Cup

More complex contexts arise if a system has memory of previous examples seen. For instance, an earlier explanation in which a currently unproved subgoal had actually been deduced, could provide additional information. The context in this case would be extended to include the complete explanation as well as the partial one. A still different context would arise if the final goal of the earlier example were different from that in the current, only partially explained, example.

We hypothesize that:

- Sets of partial and complete explanations provide contexts in which the knowledge to fill a gap in a proof can be learned.
- A number of types of information extracted from various contexts are independent of any particular domain.
- The ways in which the information is to be exploited may be implemented as a set of explicit domain-independent biases for SBL.

These hypotheses are explained in more detail in the following sections. The research proposed here is expected to verify the hypotheses.

3.3 Issues

In this section we discuss in detail a number of issues that must be addressed in order to verify the hypotheses listed above.

3.3.1 Explicit Contexts

During the course of explaining that an input example is an instance of a particular concept, an explanation-based learning system might find that its domain knowledge is inadequate to complete the proof. In such a case, new information must be learned. Standard techniques of similarity-based learning might be used to induce the missing knowledge, but this would require that many examples be seen and that they be representative of the knowledge to be learned in that they not contain similarities that are only coincidental. Rather than using such "weak" methods alone, we claim that the partial explanation sets a context in which the missing knowledge is to be learned. Additional contextual information may be found in previously derived partial, as well as complete, explanations. Contextual information may be exploited by a similarity-based learning system in order to constrain the number of hypothesized concept definitions that must be considered.

We have identified a number of dimensions that might define explanatory contexts from which to learn. Among these are:

1. The unproved subgoal is not at all provable given the domain theory *-vs-* Rules exist that would allow it to be deduced, but they do not apply to the current example;
2. Prior **complete** explanations were derived in which the currently unproved subgoal **was proved** *-vs-* No prior complete explanations were derived in which the subgoal was proved;
3. The unproved subgoal has appeared in explanations in the past in which the final goals were the same as the current final goal concept *-vs-* The unproved subgoal has appeared in explanations in the past in which the final goals differed from that of the current example.

The space defined by these dimensions is shown in Figure 14. Note that these contexts are entirely domain independent.

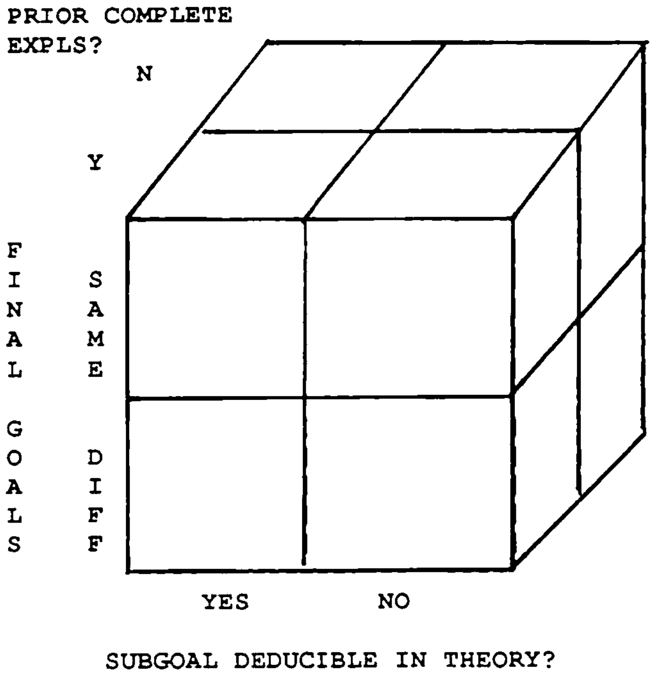


Figure 14: A Space Defining Explanatory Contexts

3.3.2 Explicit Biases

In general, similarity-based systems that perform inductive learning possess little explicit knowledge comparable to the domain theory of EBL. However, implicit knowledge exists that allows SBL to find concept definitions. The choices of input representations and algorithms for finding similarities are two examples of such **inductive biases** [Mitchell 80]. If information provided by explanatory contexts is to be useful to SBL, some mechanism must exist by which that information may be extracted and exploited. We claim that such mechanisms are heuristics that constitute a set of strong inductive biases.

Heuristics for exploiting information from explanatory contexts might include, among others:

1. lowering the priority of features already used in the proved parts of partial explanations;
2. lowering the priority of features used in the proved parts of the current explanation and not used multiple times in any single past explanation;
3. lowering the priority of features with high occurrence in past complete explanations.

Behind each of these heuristics is a rationale for its use. The reason for the use of the first method is that very often subparts of explanations do not interact on their lowest levels. Recall, for example, the proof for the cup above, where the cup's liftability had no relationship to its being an open vessel other than that these were both required to fulfill the definition of a cup. The rationale behind the second is that the history of a feature's interactions with parts of an explanation is a good predicter of the way it will interact in the current case.

The methods, or inductive biases, given here are heuristics. That is, there is a rationale for using each one, and they appear to work in many cases. They are, however, not guaranteed to work. That is, using them might result in the induction of incorrect knowledge. We claim, however, that the appropriateness of sets of these inductive biases may be determined empirically. Those performing best in various contextual situations may be stored in a knowledge base of explicit biases. Contextual information extracted from explanations then acts as an index into that knowledge base for use by SBL.

An issue to be addressed is that of selecting the set of biases that allow the "best" knowledge to be induced using contextual information.[5] We believe that the first step in identifying such "optimal" biases is by identifying the set of biases that *can* be applied in various contexts. Consider, for instance, a simple case in which no complete explanations have ever been derived that refer to the subgoal missing from a partial proof. Then no heuristics, or biases, that make use of information in complete proofs will apply. Figure 15 characterizes the applicability of the biases given above to the contexts given earlier. We propose that the selection of "optimal" biases be made empirically. Discussion of the empirical selection process is presented in Section 4 below.

3.4 Additional Issues

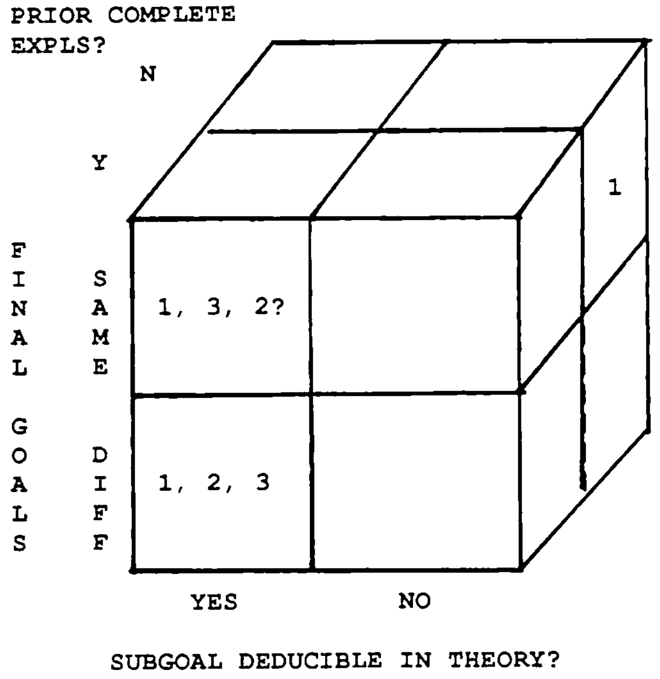
3.4.1 Representation of the Domain Theory

In order to verify our hypotheses in the context of an explanation-based learning system we must select a representation for the domain theory to be used by EBL. The representation chosen should have the following characteristics:

1. It should be a representation commonly occurring in existing EBL systems;
2. It must allow an easy mechanism to be found for the detection of gaps in the knowledge base;
3. It must provide a mechanism for learning knowledge that would link features of input examples to unconcluded subgoals.

We propose that essentially any rule-based representation is adequate for fulfilling these requirements. *If-then* rules are a common representation for domain theories. They are amenable to backward chaining from a goal, defined in the case of EBL to be the name of a concept. A gap would be detected by EBL if, in the process of backward chaining, a subgoal existed for which no proof could be found. Finally, most rule representations allow for the

[5] A definition of "best" will be given in the section on empirical selection of biases.



1. lower priority of features used

2. lower priority of features not used > 1 time in complete explanations

3. lower priority of features with high occurrence in complete explanations

Figure 15: Matching Biases to Contexts

binding of instance feature values to variables in the rules. An added benefit of this choice of representation is that our method for rule induction can be extended for general expert systems knowledge acquisition.

3.4.2 Representation of the Learned Knowledge

The representation of the learned knowledge must conform to that of the domain theory since it is to be incorporated into the theory to make it more complete. Thus our specific choice of representation for the rule base will provide a constraint on our choice for the output of the rule induction algorithm.

4 Proposed Research: An Integrated Learning Architecture for Empirical Characterization of Optimal Inductive Biases

4.1 The Gemini System

4.1.1 A Model for Integrated Learning

A natural solution to the problems of EBL and SBL is to integrate the two, as the methods are complementary in nature. Because EBL analyzes an input example by explaining it, it is able to provide SBL with additional information about the features describing the input. For example, only a subset of the features of an input instance will actually appear in an explanation. SBL could assume that these were more relevant, and thus more important, than those that did not. In general, an explanation, or partial explanation, sets a context in which the features describing an input instance may be viewed. SBL can be used to learn information that is missing from the domain theory used by EBL. This model for integrating EBL and SBL is shown in Figure 16. The model is implemented in the Gemini integrated learning system described in the next section. Gemini is unique in that it combines EBL and SBL in such a way that they are mutually dependent on each other's strengths. Other work in the development of integrated learning systems has concentrated on the use of one method as a pre- (or post-) processor for the other.

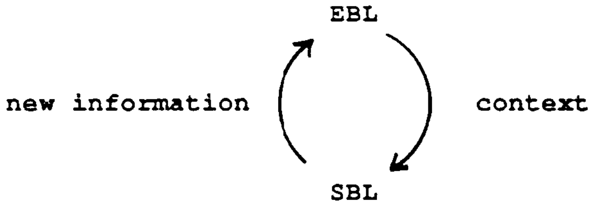


Figure 16: Integrating EBL and SBL

4.1.2 System Architecture

Gemini is composed of two communicating subsystems carrying out explanation-based learning and similarity-based learning. We will refer to these as the EBL sybsystem (EBLS) and the SBL subsystem (SBLS). Gemini's architecture is shown in Figure 17. In this section we will introduce first EBLS and then SBLS, describing their inputs, outputs, and any built-in knowledge they require.

The EBL Subsystem

The EBL subsystem of Gemini works largely as would any explanation-based learning system. It takes as input from the environment a goal which is the name of a concept and an example of it. EBL learns by first constructing an explanation, or proof, describing why the input example is an instance of the given concept and then forming a generalization consistent with the explanation. In order to construct explanations, EBLS requires a domain theory, which in Gemini is a rule base. In addition, EBLS shares access with SBLS to a static semantic memory that defines hierarchical relationships among entities in a given domain.

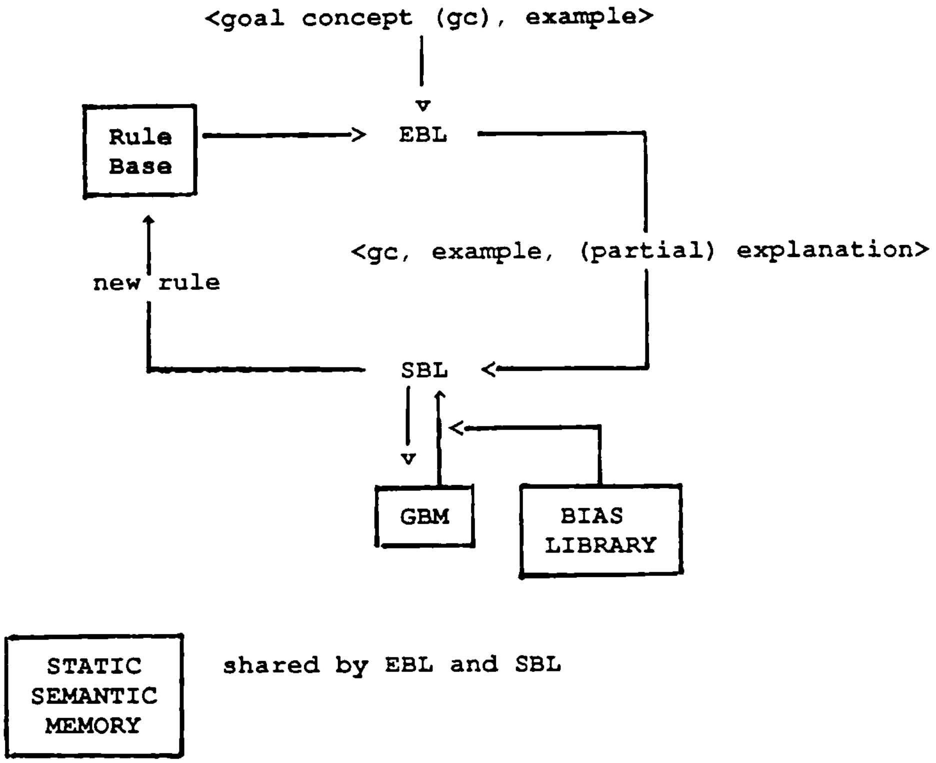


Figure 17: Gemini Integrated Learning Architecture

More specifically, the static semantic memory is a directed acyclic graph representing relationships among feature values of input instances. It is essentially a semantic net in which edges are restricted to the ISA relationship.

It is not assumed that the rule base given to the EBL subsystem is complete. Therefore, cases will arise in which only a partial explanation can be derived for a given <goal concept, example> input pair. This signals to the SBL subsystem that new rules must be induced to fill the gaps in the partial explanation. The output from EBLS to SBLS is a triple containing the goal concept (gc) name, example, and derived partial explanation. In order to provide SBLS with a base of additional information for use during rule induction, EBLS also provides it with <gc, example, explanation> triples when explanation construction is successful.

The SBL Subsystem

The SBL subsystem of Gemini is composed of two distinct SBL phases. The goal of the first phase is the categorization of input instances into general classes. The learning goal of the second phase is the induction of new rules to complete partial explanations derived by EBLS. Input to SBLS are triples containing the name of a goal concept, an example of that concept, and an explanation describing why the example is an instance of the concept. An incomplete explanation signals SBLS that a new rule must be induced. In order to do the induction, Gemini requires a set of input examples from which to learn. Rather than being provided with those sets directly by a teacher, Gemini places all input examples that it considers to be similar into groups. This is implemented by the first phase of SBLS introduced above. In order to organize inputs, the first phase of SBLS builds and maintains a **Generalization-Based Memory (GBM)** [Lebowitz 86a]. The GBM is a directed acyclic graph describing relationships among the input instances seen by SBLS. Terminal nodes of the graph are input triples containing the goal concept, example, and (possibly partial) explanation output by EBLS; internal nodes represent generalizations that classify the input instances that are their children. The GBM built need not be a single connected structure; it may be a set of unconnected graphs.

When an input containing only a partial explanation is seen by SBLS, related examples are retrieved from the GBM in order to perform the induction of the new rule. Partial matching of input instances is aided by SBL's access to the static semantic memory described for the EBL subsystem above. In addition, built into SBLS is a knowledge base of explicit inductive biases from which to select in forming new rules. Sets of partial explanations form an index into the knowledge base. Unlike the other built-in knowledge required by both the EBL and SBL components, the base of inductive biases is independent of the application domain.[6]

4.2 Experimentation with the Gemini System

We propose that Gemini be used to empirically verify the claims made above:

1. that partial explanations set a context for the learning of new rules, and
2. that explicit domain-independent inductive biases may be invoked by similarity-based learning to exploit the context information provided.

More specifically, we propose:

1. to verify that sets of explicit inductive biases can exploit context information in order to induce correct rules, and
2. select from possible sets of biases those that empirically converge on the correct rule with the consideration of fewest examples - i.e., those that converge most quickly.

The following sections discuss the proposed experiments. We begin with a description of assumptions we will make. We then describe the actual experiments and conclude with a method for analyzing our results.

4.2.1 Assumptions

The extent of incompleteness of a domain theory can vary. Empty theories, that are missing all information, and complete theories are the two extremes. The correctness of a domain theory can vary similarly. In order to clarify the scope of the problem to be addressed so that appropriate experiments may be designed, we make a number of assumptions.

[6] The knowledge base of explicit biases is actually at least somewhat dependent upon domain in that it will be implemented to work with a particular knowledge representation.

The first assumption we make is that the domain theory, represented as a rule base, does not contain any incorrect rules. The domain theory might contain rules that are too specific and that must be generalized to apply fully; but it will not contain rules that are too general. We also assume that the domain theory is tractable. That is, all necessary deductions can be made in a "reasonable" amount of time.

The nature of the missing knowledge is that complete rules are missing; they were simply not included when the domain theory was constructed. This could be manifested in one of two ways. First, it might mean that there is no way to deduce a given subgoal with the domain theory for any input example. This occurs when a subgoal to be proved is not found in the conclusion of any rule in the theory. Second, it might mean that there exist ways to deduce the given subgoal, but not for the example under consideration. This occurs when the subgoal appears in the conclusion of some rule, but the premise of that rule cannot be concluded from the example using the domain theory. These correspond to the first contextual dimension described in Section 3.1.1.

We asssume that the learning algorithm will not receive noisy input. We define noisy input to be any pair of a concept name and an example, where the example is *not* an instance of the concept.

The experiment should simulate learning of new rules during the normal course of use of the EBL system. That is, input should not be tailored to learning any particular missing rules.

4.2.2 The Experiment

Evaluation of the use of explicit biases based upon contextual information from partial explanations will be done empirically. We propose to study a minimum of two distinct domains in order to show that the biases identified are generally applicable. Domains under consideration are network fault diagnosis, radio fault diagnosis, terrorist event stories, and assembly task planning for robotics. For each domain a complete, correct, and tractable theory must exist. It is preferable that no domain theory be designed specifically for the purposes of testing the claims above, as this would provide a better measure of whether the biases were constructed independent of the experiment.

We propose to delete rules at random from the complete theories given for the domains. Questions to be answered are:

1. Can the missing rules be created using the base of explicit inductive biases that exploit contextual explanatory knowledge?
2. If the induced rules differ from those deleted, to what extent do they differ?
3. Can we identify sets of biases that guarantee correctness for a particular representation?
4. Can we identify sets of biases that perform best empirically, even if they don't guarantee correctness?

Answers to these questions will involve extensive runs of Gemini, varying parameters corresponding to:

- the selected domain;
- the degree of completeness of the domain theory;
- the number of input examples required to trigger induction;
- the subsets of all explicit inductive biases identified.

4.2.3 Analysis of Results

The study described above involves extensive experimentation with Gemini, varying the values of four distinct parameters. This will provide us with a large volume of data for analysis. In this section we describe how that data can be used in answering the questions posed above.

First, we wish to determine whether the identified set of explicit biases is sufficient to induce rules deleted from the domain theory. This is a simple matter of comparing the output of the rule induction algorithm with the set of rules deleted.

Second, if the rules induced by SBL are not identical to those deleted, we wish to determine to what extent they differ. A learned rule must fill the gaps of at least those partial explanations from which it was induced. It is preferable that the learned rule be more generally applicable than that. However, in order to maintain the assumption that the EBL subsystem have a correct theory with which to work, over-specialized rules are preferred to over-generalized rules. Our conjecture is that, in general, rules that require that additional constraints be met before firing are preferable to rules that could potentially fire in incorrect situations. We may find, however, that this is less important in practice than we now believe.

Third, given information about the structure of a domain theory, we would like to be able to prove that specific heuristics (or biases) will not result in overgeneralization. Say we knew that a domain theory was written in such a way that features of the input would never occur more than once in a single explanation. We could then safely remove from the set of features considered for induction all those that already appeared in a partial explanation.

Finally, we wish to select those sets of biases that allow us *in most cases* to induce the "best" rules with the fewest number of input instances. Given a measure corresponding to the requirements for goodness described above, this can be determined by considering the results of varied experiments simultaneously and selecting those sets of biases for which the "best" rules were found for each number of input examples. We may find that the optimal sets of biases differ among domains. We believe that our answers to the third question above will provide us with insights as to why this occurs.

5 Contributions of the Proposed Research

The research proposed above should result in a number of contributions. A major contribution is that of a method by which an explanation-based learning system can work without the requirement that its domain theory be complete. In particular, the research will devise an algorithm by which missing knowledge in the form of rules might be detected and learned by the use of similarity-based learning. The work proposed here makes use of a general theory in the form of explicit inductive biases that are domain independent. This is an improvement over earlier work that required an auxiliary theory of the specific domain of application (e.g., [Rajamoney 88; Michalski and Ko 88]). Our research differs from [Pazzani 88] in that we are specifically attempting to characterize those inductive biases that will lead to general rules without overgeneralizing them. Pazzani's work allows incorrect information to be, at least temporarily, incorporated into the domain theory. Finally, rather than relying upon heavy interaction with a teacher, Gemini's rule-learning component is relatively independent. This differs from [Hall 86] and [Kodratoff 87].

In addition to applying similarity-based methods alone to the problem of learning rules, the research will identify a set of context types indicated by the amount of information gleaned from partial explanations. These will correspond to a set of explicit inductive biases that exploit the information provided in a domain-independent manner. Thus another contribution is the departure from implicit inductive biases to an enumeration of the biases upon which a similarity-based learning system might call. The extent to which the biases aid convergence to the appropriate learned rule will be characterized and verified empirically. Earlier work in making the biases of inductive learning systems explicit is described by [Russell and Grosof 87]. They, however, consider a different problem from ours, that of explicitly selecting the vocabulary (or feature sets) over which induction is to be performed.

The general learning model proposed in this paper can be considered the first to integrate explanation-based and similarity-based learning in a relationship of mutual dependence. Earlier systems that combine EBL and SBL have treated one as a pre- (or, in some cases, post-) processor for the other. For example, [Lebowitz 86b] first does SBL in order to create generalizations which are later explained. (See [Mooney and Ourston 89] for more on systems that integrate the methods unidirectionally.)

Finally, although it has primarily been discussed in the context of EBL and SBL, the learning mechanism proposed can be applied to learning any knowledge base information represented as rules. This is of relevance to the general problem of knowledge acquisition for expert systems. Our work marks a departure from earlier work in this area in that it does not rely upon active communication with an expert (e.g., [Teiresias 83; Smith et al. 85; Van Lehn 87; Wilkins 88]).

6 Research Plan

In this section we propose a schedule for the completion of the research described above. We begin with an enumeration of the relevant work completed to date.

6.1 Work Completed to Date

Work immediately applicable to the research described above includes the following:

A first implementation of the Gemini integrated learning system has been completed. In this version of the system, a number of decisions were made about the representations of all relevant inputs, outputs, and initial knowledge bases of the system. All input examples of concepts are represented as hierarchical frame structures. That is, they are frames in which slot fillers may be values, pointers to subframes that describe a feature of the input in more detail, or nil, which indicates that the value is not known. Contextual information is not extracted from partial explanations by SBL, but rather extracted during the EBL phase and used to annotate the input frames in a manner useful to SBL. The algorithm by which this is accomplished is described in [Danyluk 87]. The rule base used by the EBL component in deriving proofs is a set of priority-ordered *if-then* rules. The premise, or *if*, part of a rule is a conjunct of terms referring to subgoals and/or the existence of specific input example features. Disjuncts are represented as separate rules. Variables may appear in the premises of a rule, enabling the specification of relationships between entities in the input; no variables appear in the conclusions of rules. Conclusions specify subgoals to be proved and are assumed to refer to the input example as a whole.[7] Further restrictions on the current implementation of Gemini include the assumption that for any input pair of a goal concept and an example, a unique proof exists explaining why the example is an instance of the concept and the assumption that no more than one gap will be found in any partial proof. Both of these assumptions will be relaxed in completing the research.

Three types of contextual information provided by (possibly partial) explanations of input examples have been identified. These include:

1. information about input feature usage in similar partial explanations;
2. information about input feature usage in dissimilar partial explanations corresponding to seemingly similar input examples when only strict correlation of input features is considered;
3. information about input feature occurrence on a global scale.

Inductive biases corresponding to the three types of contextual information have been devised. To date there has not been any specific mechanism or algorithm for identifying context types or biases. This would be important to achieve in future work. Preliminary testing of the biases

[7] Although these restrictions were placed on the representation of rules in the first Gemini implementation, they are unnecessary. The only restriction that will be necessary for future implementations is that the rules allow backward and forward chaining.

has been performed in the domain of network fault diagnosis as described in [Danyluk 89]. Representation of input examples for this domain does not exploit the hierarchical frame structure allowed by Gemini. The representation for the domain is entirely flat. The building of the GBM used in the induction of new rules has also been tested in the domains of terrorist event news stories and software maintenance.

6.2 A Schedule for Completion of the Thesis

January 1989	Investigate and select tools for representation of initial system knowledge.
February - May 1989	Identify complete set of contextual information and corresponding inductive biases to be investigated empirically.
June - July 1989	Implement knowledge bases for radio fault diagnosis domain; select a new domain that has already been implemented.
August - September 1989	Implement knowledge base of explicit inductive biases; re-implement components of Gemini to adjust to any new knowledge representations selected.
October - December 1989	Extensive testing and empirical validation of the proposed biases.
January - May 1990	Write the thesis.

Note that implementation and testing, scheduled to begin in August and October, respectively, can be interleaved.

7 Acknowledgment

This work was supported in part by the Defense Advanced Research Projects Agency under contract N00039-84-C-0165.

Addendum

The purpose of this addendum is to propose a method by which some results of the research, specifically the Gemini system, can more clearly be compared with existing machine learning systems. The concept learning method implemented in Gemini integrates two earlier machine learning techniques: similarity-based learning (SBL) and explanation-based learning (EBL). SBL and EBL are approaches that can be placed on opposite ends of a spectrum describing purely inductive to purely deductive techniques. Gemini falls somewhere between these two extremes. A potential problem in comparing Gemini against other machine learning systems is that, although there are many systems close to the two ends of the spectrum, there are very few between them that share Gemini's goal of concept learning. As a result, a thorough comparison would have to consider SBL and EBL systems as well as those that integrate the two. There are a number of ways in which the comparison can proceed. I will discuss both analytical and experimental methods of comparison.

Gemini can be compared analytically to existing machine learning systems according to a number of different metrics. Descriptions of a number of metrics follow. With the description of each I have included a brief discussion of the way I believe Gemini will compare after more thorough analysis:

- *Number of input examples required for concept learning* - In general, Gemini will require more examples than EBL, which often requires only one. It should need fewer examples than SBL. Without the heuristics that act as inductive biases, Gemini works essentially as a pure SBL system. After early experiments it appears that Gemini is able to find a concept definition with fewer examples if it uses heuristics than if it does not [Danyluk 89].
- *Amount of time for induction of a rule when a system has its input examples* - I conjecture that Gemini's time will be analogous to - i.e., within a constant factor of - SBL systems. This is because most of the work of extracting information from explanations is done before the actual induction takes place.
- *Additional time required to organize examples, to annotate them with extracted information, and to store them* - This is currently the area where Gemini suffers most. The building of Gemini's memory is alone exponential in the number of input examples. As currently implemented, Gemini's memory building component is not sensitive to the order in which it receives input. It allows an input example to appear in multiple places in the memory simultaneously, rather than choosing a single best place for storage. After early experiments it appears that this might not be necessary. It may be possible to get comparable results even if some features of the memory are restricted, for example, allowing it to be sensitive to the order of input. I believe that the currently exponential time can be easily cut to be polynomial in the number of inputs, if not less.
- *Additional memory required to store input examples* - This varies widely from method to method. The version space technique [Mitchell 78], an example of SBL, doesn't have to store input examples, but it must maintain S- and G-sets. In particular G-sets may become exponentially large. Incremental versions of ID3 [Quinlan 86], another SBL system, store all examples seen, as does Gemini. EBL does not actually store input examples, but has problems associated with the storage of potentially useless concept definitions [Minton 88].

- *Memory required to store the domain theory* - As it assumes a less than complete theory than EBL, Gemini is no worse than EBL is. Clearly, it will require more space than SBL. Relative to "mixed" systems that rely on an auxiliary domain theory (e.g., [Rajamoney 88]), Gemini should be at least comparable if not better, as the space required to hold the code for heuristics is minimal.
- *Time required to generate explanations* - Gemini is no worse than any system using EBL.

Each of these metrics should be discussed, and a comparison to other systems can be presented in tabular form. However, as Gemini does better along some metrics than in others, a case cannot be made that it is objectively better than other machine learning systems. Depending upon available resources, final goals of the learning system, and subjective preferences, it will sometimes be better and sometimes worse than others.

An additional comparison may be done by experimentally evaluating the output of Gemini against other systems. As noted above, there is no lack of machine learning systems at the two extremes of the inductive to deductive spectrum. However, there are very few in between. My conjecture is that Gemini, possessing a fair amount of explicit domain knowledge will do "better" than knowledge-poor systems, i.e., those like SBL that are closer to pure induction. We define "better" to mean learning the closest correct, i.e., not overgeneral, definition using the fewest number of input examples. Similarly, I expect that Gemini will not perform as well as more knowledge-intensive techniques.

It would be interesting to verify the conjectures made. However, that may not be possible due to the amount of time such testing would require. The problem does not lie with the testing itself, and therefore cannot be overcome by utilizing as many department resources as possible. The problem lies with the choice of domains. Those domains in which Gemini will be tested are not the same as those used for testing by other systems. Furthermore, although there have emerged standard comparison domains for inductive systems, there are no standard comparison domains for deductive systems. One possibility is to use one of my domains as a comparison domain. In order to do this, I would have to get other systems, encode my domain in them, and then do the testing. If I were to do this, I could be accused of having "fixed" the encoding so that the systems would perform in the way I had predicted. A more fair test might be to use one of the standard test domains for inductive systems. In order to accomplish this I would have to encode a theory for that domain in Gemini's representation as well as for use by at least one standard deductive system.

Rather than pursuing any of these options for testing, I propose to test Gemini only against that system that most closely resembles it in terms of inputs, outputs, and the amount of information assumed to exist within the system. The system that appears to be closest is any one of the incremental versions of ID3, for example ID5 [Utgoff 88]. In doing the comparison, I would consider a partial decision tree to be analogous to a partial rule base. ID3 is a good comparison system as it is recognized in the machine learning community as a standard against which other work should be compared. The comparison will, however, come at a price, given that I would have to use one of the ID3 domains in Gemini. One such domain is the diagnosis of soybean diseases.

In summary, I have discussed two means by which Gemini can be compared against existing machine learning systems. I believe it is important to characterize the tradeoffs in selecting a system, as described by an analysis of system performance according to particular metrics, as well as to evaluate the system's performance in terms of its output.

References

[Danyluk 87] Danyluk, A. P.
The Use of Explanations for Similarity-Based Learning.
In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 274 - 276. Milan, Italy, 1987.

[Danyluk 89] Danyluk, A. P.
Rule Induction for Incomplete Domains: An Integration of Machine Learning Methods.
Technical Report TN-89-049, Philips Laboratories, 1989.

[DeJong 81] DeJong, G. F.
Generalizations Based on Explanations.
In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 67 - 69. Vancouver, B. C., Canada, 1981.

[DeJong 83] DeJong, G. F.
Acquiring Schemata through Understanding and Generalizing Plans.
In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*. Karlsruhe, West Germany, 1983.

[DeJong and Mooney 86]
DeJong, G. and Mooney, R.
Explanation-Based Learning: An Alternative View.
Machine Learning 1(2):145 - 176, 1986.

[Ellman 89] Ellman, T.
Explanation-Based Learning: A Survey of Programs and Perspectives.
Computing Surveys , 1989.

[Fawcett 89] Fawcett, T.
Learning from Plausible Explanations.
In *Proceedings of the Sixth International Machine Learning Workshop*, pages 37 - 39. Cornell University, 1989.

[Fisher 87] Fisher, D. H.
Conceptual Clustering, Learning From Examples, and Inference.
In *Proceedings of the Fourth International Machine Learning Workshop*, pages 38 - 49. Irvine, California, 1987.

[Forbus 84] Forbus, K.
Qualitative Process Theory.
Artificial Intelligence 24:85 - 168, 1984.

[Hall 86] Hall, R. J.
Learning by Failing to Explain.
In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 568 - 572. Philadelphia, Pennsylvania, 1986.

[Kodratoff 87] Kodratoff, Y. and Tecuci, G.
DISCIPLE-1: Interactive Apprentice System in Weak Theory Fields.
In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 271 - 273. Milan, Italy, 1987.

[Kodratoff and Tecuci 87]
Kodratoff, Y. and Tecuci, G.
What is an Explanation in DISCIPLE?
In *Proceedings of the Fourth International Machine Learning Workshop*, pages 160 - 166. Irvine, California, 1987.

[Lebowitz 86a] Lebowitz, M.
Concept Learning in a Rich Input Domain: Generalization-Based Memory.
Machine Learning: An Artificial Intelligence Approach, Volume II.
Morgan Kaufmann, Los Altos, California, 1986.

[Lebowitz 86b] Lebowitz, M.
Integrated Learning: Controlling Explanation.
Cognitive Science 10:219 - 240, 1986.

[Lebowitz 87] Lebowitz, M.
Experiments with Incremental Concept Formation: UNIMEM.
Machine Learning 2(2):103 - 138, 1987.

[Michalski and Ko 88]
Michalski, R. S. and Ko, H.
On the Nature of Explanation or Why Did the Bottle Shatter?
In *Proceedings of the AAAI Symposium on Explanation-Based Learning*, pages 12 - 16. Stanford University, 1988.

[Michalski and Stepp 83]
Michalski, R. S. and Stepp, R. E.
Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy.
IEEE Transactions on Pattern Analysis and Machine Intelligence 5(4):396 - 409, 1983.

[Minton 88] Minton, S.
Learning Effective Search Control Knowledge: An Explanation-Based Approach.
PhD thesis, Carnegie-Mellon University Department of Computer Science, 1988.

[Mitchell 78] Mitchell, T. M.
Version Spaces: An Approach to Concept Learning.
PhD thesis, Stanford University Department of Computer Science, 1978.

[Mitchell 80] Mitchell, T. M.
The Need for Biases in Learning Generalizations.
Technical Report CBM-TR-117, Rutgers University Department of Computer Science, 1980.

[Mitchell 84] Mitchell, T. M.
Toward Combining Empirical and Analytical Methods for Inferring Heuristics.
Human and Artificial Intelligence.
North Holland Publishing Company, Amsterdam, 1984.

[Mitchell et al. 86] Mitchell, T. M., Keller, R. M., and Kedar-Cabelli, S.
Explanation-Based Generalization: A Unifying View.
Machine Learning 1(1):47 - 80, 1986.

[Mooney and Ourston 89]
Mooney, R. and Ourston, D.
Induction over the Unexplained: Integrated Learning of Concepts with Both Explainable and Conventional Aspects.
In *Proceedings of the Sixth International Machine Learning Workshop*, pages 5 - 7. Cornell University, 1989.

[Pazzani 87] Pazzani, M. J.
Inducing Causal and Social Theories: A Prerequisite for Explanation-Based Learning.
In *Proceedings of the Fourth International Machine Learning Workshop*, pages 230 - 241. Irvine, California, 1987.

[Pazzani 88] Pazzani, M. J.
Learning Causal Relationships: An Integration of Empirical and Explanation-Based Learning Methods.
PhD thesis, UCLA Department of Computer Science, 1988.

[Pazzani et al. 86] Pazzani, M., Dyer, M., and Flowers, M.
The Role of Prior Causal Theories in Generalization.
In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 545 - 550. Philadelphia, Pennsylvania, 1986.

[Pazzani et al. 87] Pazzani, M., Dyer, M., and Flowers, M.
Using Prior Learning to Facilitate the Learning of New Causal Theories.
In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 277 - 279. Milan, Italy, 1987.

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

[Rajamoney 88] Rajamoney, S. A.
Experimentation-Based Theory Revision.
In *Proceedings of the AAAI Symposium on Explanation-Based Learning*, pages 7 - 11. Stanford University, 1988.

[Rajamoney 89] Rajamoney, S. A.
Explanation-based theory revision: An approach to the problems of incomplete and incorrect theories.
PhD thesis, University of Illinois, Computer Science Department, 1989.

[Rajamoney and DeJong 87]
Rajamoney, S. and DeJong, G.
The Classification, Detection and Handling of Imperfect Theory Problems.
In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 205 - 207. Milan, Italy, 1987.

[Rajamoney et al. 85]
Rajamoney, S., DeJong, G., and Faltings, B.
Towards a Model of Conceptual Knowledge Acquisition Through Directed Experimentation.
In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 688 - 690. Los Angeles, California, 1985.

[Russell and Grosof 87]
Russell, S. J. and Grosof, B. N.
A Declarative Approach to Bias in Concept Learning.
In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 505 - 510. Seattle, Washington, 1987.

[Silver 86] Silver, B.
Precondition Analysis: Learning Control Information.
Machine Learning: An Artificial Intelligence Approach, Volume II.
Morgan Kaufmann, Los Altos, California, 1986.

[Smith et al. 85] Smith, R., Mitchell, T., Winston, H., and Buchanan, B.
Representation and Use of Explicit Justifications for Knowledge Base Refinement.
In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 673 - 680. Los Angeles, California, 1985.

[Teiresias 83] F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, eds.
Building Expert Systems.
Addison-Wesley, Reading, Massachusetts, 1983.

[Utgoff 86] Utgoff, P. E.
Shift of Bias for Inductive Concept Learning.
Machine Learning: An Artificial Intelligence Approach, Volume II.
Morgan Kaufmann, Los Altos, California, 1986.

[Utgoff 88] Utgoff, P. E.
ID5: An Incremental ID3.
In *Proceedings of the Fifth International Conference on Machine Learning*, pages 107 - 120. Ann Arbor, Michigan, 1988.

[Van Lehn 87] Van Lehn, K.
Learning One Subprocedure per Lesson.
Artificial Intelligence 31:1 - 40, 1987.

[Wilkins 88] Wilkins, D. C.
Knowledge Base Reifnement Using Apprenticeship Learning Techniques.
In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 646 - 651. St. Paul, Minnesota, 1988.

[Winston 72] Winston, P. H.
Learning Structural Descriptions from Examples.
The Psychology of Computer Vision.
McGraw-Hill, New York, 1972.

[Winston et al. 83] Winston, P. H., Binford, T. O., Katz, B., and Lowry, M.
Learning Physical Descriptions from Functional Definitions, Examples, and Precedents.
In *Proceedings of the Third National Conference on Artificial Intelligence*, pages 433 - 439. Washington, DC, 1983.