A comparative survey of integrated learning systems

Timothy Cain

Department of Information and Computer Science
University of California, Irvine, CA 92717

May 31, 1990

Technical Report 90-11

Abstract

This paper presents the *duction* framework for unifying the three basic forms of inference - deduction, abduction, and induction - by specifying the possible relationships and influences among them in the context of integrated learning. Special assumptive forms of inference are defined that extend the use of these inference methods, and the properties of these forms are explored. A comparison to a related inference-based learning framework is made. Finally several existing integrated learning programs are examined in the perspective of the duction framework.

Contents

1 Introduction

A number of integrated learning programs have been introduced recently, such as OCCAM [Pazzani 86], GEMINI [Danyluk 87], UNIMEM [Lebowitz 86], IOU [Mooney 89] and IOE [Flann 89]. These systems share the common idea that different inference-based learning strategies can perform better when combined by using each one's strengths to support the other's weaknesses. Until recently there was little work on classifying the different learning strategies and determining how they might support one another, so that good combinations of the methods could be proposed. This paper examines learning in the perspective of the three main kinds of inference - deduction, abduction, and induction - and the relationships among them, collectively referred to as *duction*. Several existing integrated learning systems will then be classified according to the duction framework.

In this paper, we are more interested in learning as the acquisition of new knowledge rather than learning as the improvement of performance. These two types of learning are formalized in [Dietterich 86] as symbol level learning (SLL), which improves computational performance without changing the knowledge level description of the system, and knowledge level learning (KLL), which increases the knowledge level description over time. While the framework in the next section can describe systems that perform SLL, the inference-based learning strategies we will seek are those that perform KLL, mainly because we wish to eventually develop a normative model of nondeductive knowledge level learning. One possible way to develop such a model is to examine the constraints that can be placed on an inductive learning component by other inference-based learning components. Some such constraints are discussed in Section 2.3.

This paper is organized as follows. Section 2 defines the three inference methods and describes the relationships among them, and Section 3 extends these inferences to using assumptions. Section 4 explains a related inference-based theory of learning, based on a different set of inference methods. Section 5 explains the terms used to describe an integrated learning system and then surveys twelve such systems. Finally, Section 6 summarizes this survey and presents some conclusions.

2 The three inference methods

The three kinds of implicative inference methods are deduction, abduction, and induction. In terms of causal relationships, deduction yields an effect given its cause, while abduction yields a suspected cause given its effect. Induction, the third form of inference, yields a proposed relationship between the cause and the effect. Casting duction as scientific discovery, induction is the process of theory formation from a set of data, and produces a tentative set of laws describing the data. Deduction can be viewed as making predictions based on this set of laws, allowing them to be tested. Abduction is the explanation of the results by the theory, and may invoke theory revision if the results are not covered well by the theory.

We shall see that all three forms of inference are necessary in a system that claims to be a reasonable model of learning, because each method performs a distinct function in learning and their relationships form a powerful set of learning strategies. Before examining these interrelationships, the formal definitions of the three basic types of inference are presented, followed by a comparison of their strengths and weaknesses.

2.1 Definitions

The following definitions of the three inference methods are in syllogistic terms, using major and minor premises and a conclusion. The major premise is a rule, say P $\rightarrow$ Q, while the minor premise is the antecedent P and the conclusion is the consequent Q of that rule. Given these terms[1] then

deduction Given a major premise (P $\rightarrow$ Q) and a minor premise (P), the conclusion (Q) may be deductively inferred.

abduction Given a major premise (P $\rightarrow$ Q) and a conclusion (Q), the minor premise (P) may be abductively inferred.

induction Given a minor premise (P) and a conclusion (Q), the major premise (P $\rightarrow$ Q) may be inductively inferred.

The relationship among the definitions of the three inference methods can be represented succinctly, as shown in Figure 1, where each method has its two requirements outlined and its result left uncircled. We shall use the result to categorize each learning procedure in this survey. This scheme classifies forward-chaining systems (such as some theorem provers) and backward-chaining systems (such as EBL, see [MKK-C 86, DeJong 86]) as purely deductive learners. A purely inductive method, or SBL, would be used by any data-driven, domain-independent systems, such as BACON [Langley 89] and ABACUS [Falk 86]. Very few purely abductive[2] systems exist; these include include AMAL [O'Rorke 89b] and AbE [O'Rorke 89a].

Each method has its own set of uses in a learning program. Induction is well-known as the provider of the domain theory, so it tends to serve as the basic learning component in a system. Abduction's strength is the ability to provide a hypothesis, a new (and therefore learned) piece of information that serves as an explanation for a set of events and can

[1]Throughout this paper, these terms may be used interchangeably: P for minor premise, P $\rightarrow$ Q for major premise, and Q for conclusion. Also, P and Q will occasionally be called the cause and effect, respectively.

[2]Possibly to be called HBL, or Hypothesis-Based Learning.

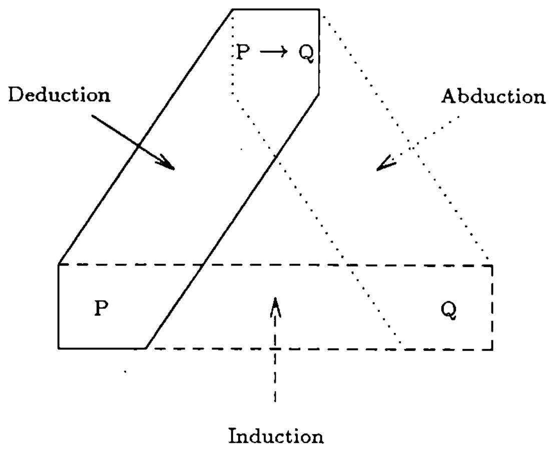

Figure 1: Relationship among deduction, induction, and abduction.

be tested for its validity. Deduction appears in learning most often as explanation-based learning, which finds an operational definition for a concept. Deduction is also used for hypothesis verification by predicting what must follow from the hypothesis and testing the truthfulness of those predictions . According to Dietterich's descriptions, then, EBL performs symbol level learning, in which we are less interested than other types of deductive learning.

2.2 Comparison of the inference methods

Before examining the relationships among the three inference methods, it might be best to present them individually. A common statement made with respect to inference methods is that deduction is the only valid method. Valid in this context means if the method is given true statements, then its inferred statement is also true. For deduction, if the minor premise and major premise are known to be true, then the conclusion *must* also be true. So if

All men are mortal

is known to be a true statement, and

Socrates is a man

is also true, then

Socrates is mortal

must be true. Unfortunately, abduction and induction are not valid inference methods. If all men are mortal and Socrates is mortal, abduction infers that Socrates is a man, which may not be true. Socrates could be a woman, and all women happen to be mortal too. Similarly, if Socrates is a man and Socrates is mortal, induction may infer that all men are mortal, which again may not be true. There may be a few immortal men, and Socrates is not one of them.

But more can be said about the three inference methods than just deduction is valid and the others are not valid. Charles Peirce, who characterized scientific inquiry in terms of the three inference methods (see [Fann 70]), called deduction *explicative* and abduction and induction *ampliative*[3]. Explicative methods can infer what must follow from a given set of knowledge, but only ampliative methods can add information beyond the deductive closure to that set of knowledge. Dietterich would say that these ampliative methods perform nondeductive knowledge level learning, which cannot be described at the knowledge level, because they add knowledge in an unjustified (i.e., invalid) manner.

Peirce introduced a scale to describe the continuum along which the three inference methods existed. This scale is shown in Figure 2. At one end of the scale is *security*, which he defined as a measure of the method's approach to certainty. Deduction is completely secure because its inferences are valid as long as its major and minor premises are true. Induction is not secure because it can produce laws that are too general, but at the very least each law holds for the data from which it was induced[4]. Abduction is the least secure because it produces a hypothesis with no evidence, although some may be gathered afterwards to support it.

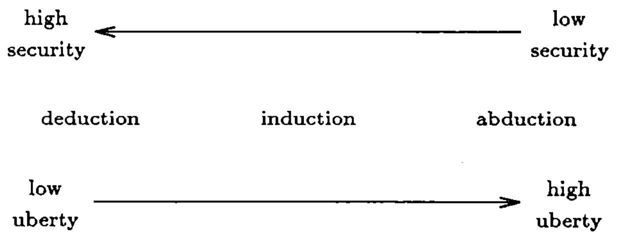

Figure 2: The security/uberty scale

[3]These terms correspond to analytic and synthetic, respectively, which are both used in the machine learning literature.

[4]Actually, the prognosis for induction is even more grim. Dietterich in [Dietterich 89] proves that any purely inductive system can only learn a fraction of the possible concepts that can possibly cover the data, and suggests studying the role of prior knowledge and its effect on the inductive process as a means of improving induction's performance.

At the other end of the scale is *uberty*, which Peirce described as the method's value in productiveness. Deduction does not produce any new knowledge and thus has a low degree of uberty. Induction can discover new laws, but is unable to do anything with these laws. They must be passed to deduction for testing, explanation, etc. Abduction can produce new hypotheses, which represent new sources of knowledge and can be used by any of the three inference methods - deduction can test the hypothesis, induction can relate it to other knowledge, and further abduction can attempt to explain it.

The juxtaposition of security and uberty is useful for noticing the strengths and weaknesses of the three inference methods. The most secure form of inference is deduction, but it has the least uberty of the three methods. On the other hand, abduction has the greatest amount of uberty, but it is not very secure at all. Induction seems to be the best "middle ground" approach, until one realizes that induction alone is neither very useful nor very secure, because it does nothing with the domain theory it builds and cannot test it. The best approach is a combination of all three methods, so that the usefulness of abduction is backed by the security of deduction, and induction can benefit from and lend support to them both. These relationships among inference methods are discussed in the next section.

2.3 Relationships among the inference methods: duction

We will use the term *duction* to refer collectively to all three inference methods and their relationships. These relationships consist of one inference method trying to improve another's performance, through such actions as providing new hypotheses, directing attention to a set of laws, or validating an inferential result. While each inference method is interesting in its own right, the truly powerful learning strategies grow out of the relationships among the three methods. Some findings on these relationships are presented here, grouped by the influencing method.

2.3.1 Inductive influences

Induction is the primary knowledge level learning mechanism in this framework. Any rules not provided by a teacher or other outside source must originate from the induction component. This component provides the domain theory to the abductive and deductive components, so insuring the validity of these laws is extremely important. Although the process of deduction is valid, it may still produce incorrect inferences if the induced rules are not correct. Fortunately, both deduction and abduction can serve to help induction produce valid rules.

Induction can also influence itself, by performing induction on the rules produced by induction. This kind of self-directed influence, described in [Russell 86], allows an inductive component to build up a set of higher-level regularities that themselves instantiate into laws. For example, after observing several Americans speaking English, a hypothetical system may induce that all Americans speak English. Similarly, after seeing several Mexicans speaking Spanish, the system may induce that all Mexicans speak Spanish. Using induction on the these two rules, the system may produce a higher-order rule, that all people of a given nationality speak a common language. This regularity allows rules, such as all Soviets speak Russian, to be created with just one instance. These new rules will tend to be correct, because they are supported through the higher-level rule by the instances that support the other similar rules.

2.3.2 Abductive influences

Abduction is mainly used to form explanations. These explanations relate an event to its probable cause by using the rules such as those produced in an inductive phase or those provided by a teacher. When the explanation is complete (i.e., all of the causes are known to be true), then the explanation is also deductively secure, since deduction could build the same explanation structure starting from the causes that abduction built starting with the effects[5].

Sometimes the explanation is incomplete, in the sense that some of its antecedents are unprovable under the current set of laws. This problem can be addressed in several ways: the antecedent can be assumed to be true (as in AbE [O'Rorke 89a]), the entire explanation can be abandoned (as in standard EBL [DeJong 86]), or the inductive component can be invoked, either to empirically determine the truth of the antecedent or to find a new law to explain it. We find the latter method preferable, since it takes advantage of the integrated framework and attempts to learn more about the antecedent in question. This directed query to the inductive component also provides it with an attention mechanism (e.g., its goal is to find what is associated with P, and not just to find any associations in the data). Fawcett uses exactly this method in [Fawcett 89] to find rules for unproven antecedents.

Abduction can also direct induction in another way. After several explanations are completed by the abductive component, it may call upon induction to find higher-level relationships among the explanations, as suggested by [Russell 86]. This induction over explanations is the basis of the IOE system [Flann 89], although IOE uses deduction, not abduction, to form its explanations, that necessarily are complete.

Abduction can also direct the deductive component by providing it hypotheses to test. The deductive component derives what must be true if the hypothesis was valid, and then seeks the validity of these conclusions. For instance, the system may abduce that P is the cause of some event, and this hypothesis is sent to the deductive component for testing. Every conclusion that follows from P and the present set of knowledge could be explored, and each conclusion that is found to be true adds to the certainty level of the hypothesis. This support for the hypothesis represents deduction supporting abduction.

[5]This symmetry of abduction and deduction on secure explanations may be one reason the two inferences are so often confused. See also the discussion of abduction versus assumptive deduction in Section 3.

2.3.3 Deductive influences

Of course, deduction can do more than support abductive hypotheses. If any conclusion is found to be contradicted by known facts, then the associated hypothesis is weakened or abandoned altogether. Similarly, the deductive component can test the induced domain theory, by looking at the conclusions the rules imply from known facts. If the facts are certain, then any conclusion found to be true or false will support or weaken, respectively, the rule(s) used to deduce it.

A standard use for deduction is the construction of complete, valid explanations. Deduction is also useful in verifying that a set of observations obey a rule that is "uninduceable" in the sense that a given inductive method would not find a rule that would explain the data, but a rule existed nonetheless. While this framework was being developed, an experiment was conducted to determine whether deduction and induction could improve each other's performance (see [Cain 89]). A deductive component was added to BACON, a purely inductive scientific discovery system, so that it could test its induced numerical laws. One of the findings was that the improved BACON could deductively accept data that the inductive component would reject as having no pattern. For example, the inductive component would reject the ideal gas data in Table 1 because there is no correlative relationship between any two columns of data. But this data obeys the law

$$Pressure \times Volume = 50.0 \times Temperature$$

perfectly, and the deductive component can check this accordance quickly and efficiently.

Gas	Volume	Pressure	Temperature
Ideal	1.0	100.0	2.0
Ideal	5.0	10.0	1.0
Ideal	10.0	40.0	8.0

Table 1: Ideal-gas data

Another useful finding from the experiment was that the deductive component could determine useful variables upon which the inductive component should concentrate its search. After a set of laws were found to be incorrect, the deductive component would rearrange the data so that induction would initially search the space of laws that included the variables in the failed laws. For instance, if the law $PV = 50.0$ failed, then a new search would begin for laws that used the variables P and V. This rearrangement heuristic is based on the idea that for a law to fail in the deductive phase, the inductive phase must have earlier proposed some invalid law. Therefore, it is likely that, at the very least, *some* of the variables in the failed laws will appear in the final laws. In practice this heuristic reduced these failure-driven searches by an order of magnitude, in terms of number of relationships explored.

Finally, deduction can be used to "chunk" rules, as seen in a variety of systems and mentioned in [DeJong 86] and in [O'Rorke 89a, O'Rorke 89b] (which refers to this as macro construction). Chunking occurs when the proof tree for some set of conclusions is reduced to only its leaves; all internal nodes are removed. For example, the long chain of deductions $P \rightarrow R$, $R \rightarrow S$, $S \rightarrow T$, and $T \rightarrow Q$ can be replaced with $P \rightarrow Q$. This last rule is referred to as the operational definition for Q if P is a commonly or easily observed premise. Operational definitions are preferable because they speed up the deductive system, allowing it to conclude Q in one step rather than many.

3 Assumptive inference

A common misconception is that abduction is, in general, the method of making assumptions. Actually, all of the inference methods can support making assumptions, and each one has two assumptive forms, depending on which of the two required premises is assumed to be true. For example, if one has the major premise $P \rightarrow Q$ and assumes the minor premise P, then a form of assumptive deduction will infer the conclusion Q. Table 2 lists the six assumptive forms.

Form	Given	Assume	Infer
Assumptive deduction	$P \rightarrow Q$	P	Q
	P	$P \rightarrow Q$	Q
Assumptive abduction	$P \rightarrow Q$	Q	P
	Q	$P \rightarrow Q$	P
Assumptive induction	P	Q	$P \rightarrow Q$
	Q	P	$P \rightarrow Q$

Table 2: The Six Assumptive Forms

A quick (and by no means exhaustive) review of popular learning systems shows that the assumptive forms of deduction and abduction which require assuming a major premise are not used. Since they effectively represent assuming a law so a fact can be used, their absence seems justified. Similarly, assumptive induction (either form) is rarely seen, but it occasionally is invoked to form laws using assumptions made by assumptive deduction and assumptive abduction.

Of particular interest is a comparison between abduction and the useful form of assumptive deduction, the form that assumes the minor premise. Unfortunately, these two methods are often confused because they share the surface similarities of starting from a major premise $P \rightarrow Q$ and having a minor premise P of undetermined certainty. Abduction takes the major premise and the conclusion Q and abduces P *at an unknown level of certainty.* But assumptive deduction starts only with the major premise and assumes the minor premise is true, and then deduces the conclusion. Notice, though, that if P is assumed at a level of certainty C, Q can be deduced at the same certainty level C, although this certainty can be adjusted up or down through another (possibly empty) set of rules.

For example, let's say that a traffic light is controlled by either an internal timer, which switches the light from red to green every several seconds, or a pedestrian crosswalk button, which turns the traffic light red immediately. If a car is stopped at a red light, one can *abduce* that the timer has made the traffic light red or a pedestrian has pressed the button, but neither inference has a known level of certainty in the absence of further information. But if one assumes that the timer makes the light red 50% of the time, then one can assumptively *deduce* that the light will be red at least 50% of the time, because sometimes a pedestrian will make the light red, too.

Thus, the two forms of inference propagate certainty levels very differently. Abduction results in a premise P of unknown certainty from a conclusion Q of 100% certainty, while assumptive deduction results in a conclusion Q of certainty equal to the assumed premise P.

4 A related framework

The duction framework is not the only inference-based framework that has been proposed to the machine learning community. In [Michalski 89], another theory of integrating learning methods is proposed that uses a different basic set of three inference methods, in this case induction, deduction, and analogy. In the duction framework, analogy would be classified as a form of induction. However, Michalski chooses to view abduction as a form of induction. He derives a set of five learning strategies, which he believes can be used to integrate empirical learning, constructive induction, learning by instruction, constructive deduction, explanation-based learning, reinforcement learning, conceptual clustering, and learning by analogy. Since any input builds up the domain theory, Michalski calls his methodology multistrategy constructive learning.

4.1 MCL: multistrategy constructive learning

The basic idea behind these inference-based learning methods is that any input affects the learner's background knowledge, or domain theory, through one of three inference methods (deduction, induction, or analogy). Learning is only possible by performing inference and by having a memory in which to store the results of inference. Memory also holds the domain theory, if any, with which the learner begins. A particular learning strategy is selected by a function of the input, the domain theory, and the learner's current goal.

Michalski discusses the importance of explanations in an advanced learner as opposed to a beginning learning system, which simply learns by rote to build up its background knowledge. He divides an explanation into two parts: the explanatory hypothesis, which is proposed as "entailing" the set of observations through the use of domain theory, and the explanatory structure, which describes how the hypothesis and domain theory actually entail the observations. He further divides explanations into deductive explanations, which don't need the explanatory hypothesis because the domain theory is sufficient, and inductive explanations, which require explanatory hypotheses. Both kinds of explanations, he claims, are different from the generalizations produced by empirical inductive learning, which simply summarize instances, because a true explanation involves abstractions and causal relationships.

Michalski describes his five basic learning methods that may be used on input, depending on the input and the learner's goals.

1. *Empirical learning.* This is the creation of an hypothesis on the basis of the given input and domain-independent knowledge, using little or no background knowledge.

2. *Constructive induction as generalization.* This uses background knowledge to form a generalization of the given input.

3. *Constructive induction as abduction.* This learning form creates an hypothesis to account for the given input, by having the input deductively follow from the hypothesis using the domain theory.

4. *Constructive deduction.* This is also called *abstraction.* This method uses the domain theory to deduce a generalization of the input. It differs from generalization in that abstractions are deductively valid and thus always true. For instance, "John lives in California" deductively abstracts to "John lives in the United States" since California is part of the United States (a background fact).

5. *Explanation-based learning.* This is the construction of a deductive explanation using only the input and the domain theory (no hypothesis).

Similarly, Michalski describes five situations that may result when input is received by the system:

1. *The input is completely novel.* If no background knowledge can be generalized to account for it, then the input is simply stored away.

2. *The input contradicts the domain theory.* The learner either specializes its knowledge, or simply stores the input, depending on the input's confidence level and the amount of the domain theory that must be revised to account for it.

3. *The input is implied by the domain theory.* The deductive explanation is constructed, but what is done with this structure depends on the learning goal. For instance, the explanation may be generalized and used as an operational definition of the concept expressed by the input.

4. *The input is similar to the domain theory.* The system tries to match the input to its background knowledge "at a higher level of abstraction" (i.e., analogically). If this proves to be impossible, then the input is treated as novel (the first case).

5. *The input is already known.* That part of the background knowledge has its confidence level increased.

Michalski uses the classic "cup" domain as an example of how the different strategies would learn various aspects of the concept of a cup, given different starting information and goals. For example, given an example and some domain rules, his system could produce an abstraction of the example using constructive deduction or an abstract concept description using constructive induction as generalization. With the new abstract concept description, the system could produce an operational concept description using explanation-based learning. Unfortunately, the system has not yet been implemented.

4.2 MCL as duction

It is relatively straightforward to view MCL in the duction framework. Constructive deduction and explanation-based learning are both deduction. Empirical learning and constructive induction as generalization are both induction. Of course, constructive induction as abduction is abduction. MCL seems overly complicated and has some other problems.

For example, constructive deduction and explanation-based learning appear to be the same mechanism, except constructive deduction is not constrained since it has no target concept. This makes EBL a special case of constructive deduction, but begs the question, *when does one stop abstracting with constructive deduction?* Certainly not when the very end of the deductive chain is reached, for that could be far too general (e.g., "John lives in the universe."). Some reasonable stopping criteria can probably be found, but this is not mentioned.

Conversely, constructive induction as generalization and abduction seem very different mechanisms, which Michalski clumps together as one learning strategy. Michalski treats generalization as a special case of abduction, since both are "inverses" of deduction. This may mean they are both synthetic methods compared to analytic deduction, but they are by no means similar in underlying mechanism. In fact, one could argue that deduction, as EBL, is a special case of abduction in which the antecedents of the explanation are all known to be true[6]. But the real issue is classifying the methods based on what they learn, which for induction, deduction, and abduction are very different indeed.

On a final note, the paper seems somewhat contradictory at times. For example, Michalski uses constructive induction as abduction to learn domain rules (see Figure 2, page 19), but earlier (page 13) has this method using domain rules, not learning them. Unfortunately, there is no system yet for exploring the many possibilities MCL seems to offer. Overall, though, its concept as an inference-based theory of learning is similar to the duction framework.

5 Survey of integrated learning systems

Several integrated learning systems have been produced in the past five years. This section details a dozen of these systems[7]. Almost without exception, these systems combine deductive and inductive logic, integrating EBL and SBL in various ways. This survey focuses on how the systems use the different inference methods to influence each other and improve the overall performance of the system. Unfortunately, this improvement is not always seen, and in some cases a degradation of performance can occur. This implies that integrated learning is not always superior to single-method learning.

Before describing the systems, some definitions of domain theory assumptions are required. These are assumptions that each learning system makes about the domain theory given as input. As with any assumption, it may not be true, but the learner will act as if the assumption applies to the given domain theory anyway.

Given two mutually exclusive concepts A and B[8], a domain theory is described as

Complete - can explain all examples of concept A and all examples of concept B

Correct - never explains an example of concept B as an example of concept A, and vice versa

Consistent - only explains examples of concept A as examples of concept A, and examples of concept B as examples of concept B

Incomplete - cannot explain all examples of concept A or all examples of concept B (this is an *overspecific* domain theory)

[6]Fawcett mentions this special case occurring in his system PLAUSEX [Fawcett 89].

[7]The perceptive reader will note that connectionist systems are avoided in the survey. The author felt their addition would considerably lengthen this survey and require even more use of jargon than without them. No slight towards these fine systems is intended.

[8]We do not say A and not(A) because many systems have difficulty with negation, especially as the head of a rule. Also, requiring that B be the negation of A is too specific, when all that is required here is mutual exclusion.

Incorrect - explains some examples of concept B as examples of concept A, or vice versa (this is an *overgeneral* domain theory)

Inconsistent - can explain an example of concept A as an example of concepts A and B, or explain an example of concept B as an example of concepts A and B

Single - can only produce one explanation for an example

Multiple - can produce several different explanations for an example. Different explanations here can mean different complete explanations, or partial explanations with different structures.

These assumptions are not all mutually exclusive, so it is possible for a system to assume that its domain theory is both incorrect and incomplete[9]. Instead they are pairwise mutually exclusive and exhaustive, meaning a theory is either complete or incomplete[10], but not both. From the definitions of the assumptions, it is relatively easy to show that correct theories are always consistent, but incorrect theories are not necessarily inconsistent. Note that the multiple explanation assumption does not presuppose that these explanations are inconsistent, but all inconsistent domain theories must support multiple explanations.

A Venn diagram of these relationships is shown in Figure 3, where the space of all possible domain assumptions is divided among incomplete, incorrect, and inconsistent theories. The space outside each of these imperfect theories represents the space of its opposite assumption, so the space outside of the incomplete space is the area of complete theories. The single and multiple explanation assumptions are not shown, since any domain theory can support either assumption, with the exception of inconsistent domains (as noted above).

Some of the systems act as if their assumptions are correct without checking them first. For example, the IOU system [Mooney 89] assumes that the supplied domain theory is incorrect, in that explanations that are built with the theory may apply to negative examples of a concept as well as positive ones. So even if the given theory is correct, IOU will try to "fix" it anyway.

Rajamoney and DeJong classify imperfect domain theories (corresponding to incomplete, incorrect or inconsistent theories, as defined here) in [Rajamoney 87] as incomplete or inconsistent, each with two types. While their system and the system presented here can be mapped onto each other and are not contradictory, this mapping is not one-to-one in either direction. Our system is preferred due to its direct mapping to the semantically simple meanings of overgeneral and overspecific. Also, Rajamoney and DeJong assume that multiple explanations are expected to be inconsistent, an assumption that is not forced under our framework.

[9]An incorrect and incomplete theory is often called imperfect.

[10]With respect to one concept, that is.

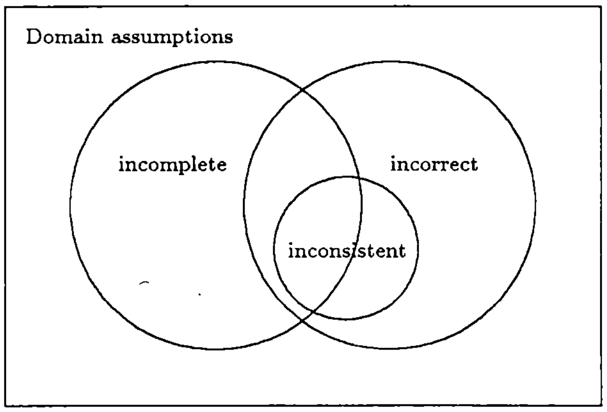

Figure 3: The relationships among imperfect domain theories

Most of the following twelve systems address the incomplete theory problem (see Table 3), although five systems attempt to solve the incorrect theory problem (and two of those try to solve both). The majority of the systems can only employ single-explanation domain theories. Only three systems can work with multiple-explanation theories. Three of the systems, ML-SMART, UNIMEM, and SaranWrap, appear to tolerate inconsistent as well as incorrect domain theories.

The terms *tight integration*, *supervised*, and *incremental* appear in Table 3. Tight integration means that the system can perform either of its inference-based learning strategies at any time. Compare this to a modular arrangement, in which one component completely finishes its learning and passes its results to another component, which then performs its learning strategy. The second term, supervision, means that the learning system is supplied with an oracle of some kind. Typical oracles are those that give the correct classification of a given instance or those that confirm or deny a system's generalizations. Finally, incremental systems are those that perform some learning with the input of each instance and never reprocess those instances. Nonincremental systems need to be given a set of instances all at once, and reprocess those instances whenever more input is received.

The following survey is organized by the domain assumptions of the systems. For the systems that assume single-explanation theories, six systems assume that the theory is incomplete, two assume incorrectness, and one assumes incorrectness and inconsistency. For the systems that assume multiple-explanation theories, one assumes the theory is incomplete and two can work with any kind of imperfect theory. The survey is followed by a summary of these integrated systems.

Main author	System	Combines	Supervised/ Incremental	Domain assumptions
Blum	RX	I-D	UnSuper/Nonincrem	Incom, Sing
Danyluk	GEMINI	D-I	Unsuper/Increm	Incom, Sing
Drastal	MIRO	D-I	Super/Nonincrem	Incom, Sing
Muggleton	CIGOL	A-I†	Super/Increm	Incom, Sing
Pazzani	OCCAM	D-I	Unsuper/Increm	Incom, Sing
Widmer	—	D-I†	Super/Increm	Incom, Sing
Flann	IOE	D-I	Super/Nonincrem	Incor, Sing
Mooney	IOU	D-I	Super/Nonincrem	Incor, Sing
Lebowitz	UNIMEM	I-D	Unsuper/Nonincrem	Incor, Incon, Sing
Fawcett	PLAUSEX	A-I	Super/Increm	Incom, Mult
Bergadano	ML-SMART	D-I†	Super/Nonincrem	Incom, Incor, Incon, Mult
Pazzani	SaranWrap	D-I†	Super/Nonincrem	Incom, Incor, Incon, Mult

† These systems tightly integrate their two inference methods. The three that combine deduction and induction prefer the former over the latter.

Table 3: System summary

5.1 Single-explanation learners

These systems (RX, GEMINI, MIRO, CIGOL, OCCAM, Widmer's system, IOE, IOU, and UNIMEM) assume that the domain theory supports only one explanation for a given example. The first six systems (RX through Widmer's system) work on incomplete domain theories, but each uses a different set of learning strategies or a different integration scheme of supervision, incrementality, and modularity. The next two systems, IOE and IOU, work with incorrect domain theories. Both use EBL to build explanations for a given set of input, but IOE then runs SBL over the explanations, while IOU runs SBL over the features that do not get used in explanations. Finally, UNIMEM can work with inconsistent domain theories as well as incorrect ones.. Although inconsistent domain theories necessarily can be used to explain an example in multiple ways, UNIMEM is supplied with a set of instances and uses this set to indicate which of the conflicting rules to use.

5.1.1 RX

Overview

In [Blum 86, Blum 82], Blum describes the RX system, a set of programs for discovering and confirming medical hypotheses with the use a large database. The system follows a model of clinical research that involves four modules:

- a *discovery process* that finds relationships among medical variables (this is the inductive component)
- a *study module* that designs studies for each hypothesis formed by the discovery module
- a *statistical analysis package* that applies the model formed by the study module to the data (this is the deductive component)
- a *knowledge base* that holds medical knowledge and patient records, and to which newly confirmed hypotheses are added (this is the domain theory)

All of the modules, with the exception of the statistical package, are described in the papers. The statistical package is the separate system IDL, the Interactive Data-analysis Language.

The discovery module looks for relationships among primary and derivable attributes of a patient's record, examining the attributes in pairs, searching for correlations (an $O(n^2)$ process). Any number of patients in the database can be examined by the discovery module, with the remaining patients serving as a testbed for the hypotheses. For the experimental trials in the paper, ten patients were used by the discovery module.

The study module is the largest of the four parts of RX. It accepts a hypothesis from the discovery module (or directly from a human researcher) and generates a model of the hypothesis that is plausible to test based on the medical and statistical knowledge in the database. The study module tries to show that the hypothesis is not spurious (i.e. the effect has some other cause besides the one being postulated). Production rules act as heuristic guides for the study module in doing such tasks as controlling confounding variables. The model is then passed to the statistical analysis package, which confirms or disconfirms the hypothesis using a larger set of patient records in the database.

The knowledge base used data from ARAMIS, the American Rheumatism Association Medical Information System. This database is hierarchical, where each node represents a schema containing property-value pairs of medical and statistical information.

Critique

This system is interesting in that it does not come from the machine learning community and its integration of learning methods predates any other system described here by at least four years. In addition, RX has a real, complex domain theory, and its study module for designing experiments to test hypotheses has no equivalent in any other known integrated learning system[11]. RX has been tested using data on real problems and has been fairly successful.

[11]However, see [Kulkarni 88, Falk 88] for examples of machine discovery systems with the capability to propose and interpret experiments.

However, RX has its limitations. Its use of pairwise causation limits the system so that it cannot discover multiply-caused diseases (e.g., where $A \wedge B \rightarrow C$). RX can use incomplete domain theories, but noise does not appear to be tolerated, which is odd for a real domain. For example, the pneumonia case (page 171 of [Blum 82]) requires the patient's temperature to be over 38 degrees Celsius. If the temperature is 37 degrees, this exacting rule seems to imply the pneumonia is ruled out. Perhaps there is a parameter that determines how close two values can be to be considered equal (cf. BACON in [Langley 89]). Evidence for several hidden parameters appears in the paper. For example, the causality rules on page 172 mention repeated observations where A generally precedes B and the intensity of A is correlated with the intensity of B. No mention is made of how many observations are necessary, how often A must precede B, or how closely A and B must correlate to trigger the causality rules. These hidden parameters may be finely tuned for RX's particular domain.

Since Blum was concerned with knowledge acquisition, RX never uses its knowledge to form hypotheses in unknown cases, an abductive process. The addition of an abductive diagnostic module to RX would be an interesting project.

5.1.2 GEMINI

Overview

Danyluk's GEMINI [Danyluk 87] system combines EBL and SBL together to generalize an incomplete domain theory. A subsystem of GEMINI, a learner/analyzer module, is described that can generalize and explain an event by using a rule base, a concept hierarchy, and an incident hierarchy (of previously generalized events). EBL and SBL are combined in the sense that, while each is used in a distinct phase, each provides the other with information that directs its learning during its phase.

First, the EBL component builds an explanation of a new event, using the rule base as its domain theory. This explanation is not generalized but is instead passed to the SBL component. This component compares the features of this explanation to those of other similar events in the incident hierarchy, producing a generalized explanation of the events. A concept hierarchy is provided so that the SBL component can make inexact matches. For example, if one event occurred in a bookseller's store and another occurred in a grocery, then the SBL component would generalize the event location to `store`. The SBL component also uses contextual information when generalizing, so the times of the two events above may be generalized to `business hours` rather than `daytime`, because the location is a public place. Contextual clues are based only on those features that were in the explanations, since they are guaranteed to be important.

Thus, the EBL component provides the SBL component with an explanation, which can be viewed as a group of relevant features and their context. Danyluk claims that the SBL component, in turn, can provide the EBL component with causality clues to help it select rules to explain complex events. This SBL influence is not yet implemented and therefore is briefly described, but Danyluk relates the process to UNIMEM's extraction of predictability information from a set of data (cf.). Finally, since the SBL component is performing the generalization of an explanation, it controls the range of possible generalizations of an event that can be made by the system.

Critique

There are some problems with this integration of EBL and SBL. Explanations are generalized by the SBL component and, unless the matches are exact, this generalization is constrained by the concept hierarchy. For example, `bookseller` and `grocery` must appear in the hierarchy under `store`, or GEMINI will not find any similarity between the two locations. This implies that the concept hierarchy must be complete, but solving such a complete domain theory problem is what GEMINI is supposed to do. Danyluk has replaced a complete domain theory with an incomplete rule base and a complete concept hierarchy.

Another problem is that the SBL phase only looks at features mentioned in the explanation. With the incomplete domain theory, the EBL could fail to produce an explanation. The SBL phase would then have nothing to generalize. Perhaps the SBL component should only give more attention to the features that appear in the explanation but still use unexplained features, similar to IOU. Then two unexplained events could be compared for the features they share, and these common features could be made into a (possibly overspecific) rule to be later generalized by GEMINI.

5.1.3 MIRO

Overview

Drastal and Czako [Drastal 89] describe MIRO, a system that combines deduction and induction for concept formation. Basically, the system uses deduction to form an abstraction space, and then performs induction in this space, rather than the original space.

More specifically, deduction finds the abstraction space $\mathcal{A}$ by constructing the set $L_{\mathcal{A}}$, the abstract concept description language. A bottom-up deductive procedure is used on an and/or graph induced from the domain theory; the procedure finds a set of pairs (x, α) where x is a descriptor and α is a pointer to a proof that establishes x as true or false. There is one pair for each positive and negative training example. The entire set makes up $L_{\mathcal{A}}$.

Induction proceeds through candidate elimination. A positive instance, called the seed, is selected from the training set, and a set G of partial concept descriptions is formed that excludes all negative examples and includes a subset (not necessarily proper) of the positive examples. The subset is guaranteed to include the seed. The elements of G are then specialized by adding descriptors from the abstract language $L_{\mathcal{A}}$. All positive instances covered by this set are removed from the list of positive training examples, a new seed is selected, and the process continues.

Critique

The authors present convincing evidence that induction in the abstraction space needs fewer examples than the same inductive method in the original space. However, while they state that the method is able to extend an incomplete domain theory, they do not say what happens when no proof is available for a particular descriptor x_j. This event could occur when the value of x_j is unknown or the domain theory is too specific to explain x_j's value.

Drastal and Czako also assume there is only one way to explain the value of a descriptor during the deductive phase (this is the single explanation assumption). There may be more than one abstraction space, defined as several ways of proving the value of descriptor x. If so, one of these spaces could be better than another for performing induction. More exploration of this idea could have interesting results, such as how to find the best space without doing induction in all of them.

5.1.4 CIGOL

Overview

Muggleton and Buntine [Muggleton 88] describe CIGOL, an inductive/abductive learning program that can generate and generalize new predicates and thus define its own vocabulary for concept descriptions. The basis of CIGOL is a method of inverting resolution, a deductive process, leading to its opposite inference, which is induction if a rule is inferred and abduction if a predicate is inferred. All generalizations of predicates are confirmed by a teacher before CIGOL accepts them.

CIGOL uses three operators to perform inverse resolution: "V" operators take the clause at their base and one of their arm clauses, and produce the other arm clause; they are called ***absorption*** or ***identification*** operators depending on whether the literal resolved on is positive or negative in the supplied arm clause. "W" operators find a clause A such that A resolves on a common literal L with two other clauses C_1 and C_2 to form B_1 and B_2. Since L is resolved away and does not appear in B_1 or B_2, "W" operators introduce new predicates, and the operators are called ***intra-construction*** if this new predicate is negative and ***inter-construction*** if this new predicate is positive. The third operator type is truncation, which handles the case when the base clause(s) of "V" and "W" operators are empty, which occurs at the root of every refutation tree.

Some examples of concepts that CIGOL has learned are list-reverse, list-minimum, ordered-binary-tree-insert, and merge-sort. While learning some of these concepts, CIGOL invented predicates such as "append" and "less-than" (the names were supplied by the user). CIGOL is the only system here that can invent new predicates to add to its representation language.

Critique

Presently, CIGOL only uses absorption, intra-construction, and truncation. Since both of the remaining operators, inter-construction and identification, construct positive terms, this implies some sort of difficulty with performing inverse resolution involving a negative term as the basis. No explanation for the lack of implementation of these operators is given, though.

Several assumptions are made in the paper. The first, the assumption of separability, assumes that there are no common literals in the two arm clauses after substitution, except for the literals resolved on. The second assumption is that the arm clause used by a V operator, absorption or identification, is a unit clause. These assumptions would appear to limit the space of possible generalizations made by CIGOL, but again, no explanation is given in the paper.

An interesting task would be to remove the user from CIGOL's loop. Examples would come from the environment, predicate generalizations would be tested directly by CIGOL rather than asking the user (although CIGOL would need to handle probabilities, since nothing short of exhaustive testing can confirm a theory), and names of sub-concepts would be arbitrary. A partial test, which ran CIGOL in an automatic mode where all questions to the user were assumed to be answered in the affirmative, is described in [Muggleton 89]. Unfortunately, this test did not involve the generation of any new predicates by CIGOL.

5.1.5 OCCAM

Overview

Pazzani, Dyer, and Flowers [Pazzani 86] present OCCAM, an oft-cited "classic" of integrated learning. OCCAM combines EBL and SBL to generalize events and extend its incomplete domain theory. OCCAM's basic algorithm takes each new event and explains why it is similar to other events that share its most specific generalization. Similar to GEMINI's concept hierarchy assisting its SBL component, OCCAM maintains a theory of causal relationships to help its EBL component establish explanations for the similarities among the events. If prior causal theories confirm this explanation, a new explanatory generalization is created. If they deny the explanation, it is discarded. If the prior causal theories neither confirm nor deny the explanation, then it is marked as a tentative generalization. If a future event contradicts the tentative generalization, it will be discarded. Otherwise, a variety of strategies can be employed to confirm the tentative generalization, including increasing confidence as it successfully predicts new events until some confirmatory threshold is reached.

This description is an overview of *basic* OCCAM. Several extensions have been made to the system since its description in [Pazzani 86]. Two extensions in [Pazzani 88] allow OCCAM to work more effectively with incomplete domain theories and to work with incorrect domain theories. The first extension allows OCCAM to induce a missing explanatory link after seeing several examples by using an abstract explanation formed from a causal pattern. For example, after seeing examples of economic sanction, OCCAM can use an abstract explanation that a state of an object enables an action upon it and induce that suppliers can sell their goods for a greater price if the demand increases. The second extension allows OCCAM to detect incorrect schemas by their incorrect predictions and correct them by rederiving the schema with a revised domain theory. For example, OCCAM may have a rule that tall people try to preserve the health of others. While a foundational example may correct this rule, a coercion schema based on the old rule may still exist and may fail to explain a coercion example. OCCAM determines that the schema incorporates a rule that is no longer supported and uses EBL to derive a new coercion schema.

In [Pazzani 89b], OCCAM is further extended to detect those rules that are preventing an example from being explained. OCCAM can then generalize these overspecific rules, correcting its incomplete domain theory. For example, if OCCAM had an overly specific rule that required two interacting countries to have free economies, a coercion schema using that rule may fail to explain a coercion example involving a country with a controlled economy. OCCAM can detect this rule by finding that it causes the difference between the example's features and the schema's features. The rule is then corrected so that it accounts for the example.

Critique

As with Fawcett's system (cf. PLAUSEX), it is possible for a set of events to cause an incorrect tentative generalization to be confirmed and enter basic OCCAM's domain theory as an explanatory generalization.

For example, OCCAM can work in a domain of inflating balloons where the theory is empty. After given a positive and negative example, a tentative generalization is produced that balloons need to be red if they are to be inflated. In their paper, this generalization is refuted by a later contradictory event of a green balloon being inflated. But if the tentative generalization is coincidentally supported by later events, it will be confirmed and become an explanatory generalization. This new rule,

```
inflatable(X) :- isa(X,balloon), color(X,red).
```

results in a domain theory that is both overspecific, since not all inflatable balloons can be explained by it, and overgeneral, since not all red balloons can be inflated. Since

OCCAM cannot perform the specialization needed to correct an overgeneral theory, using the incorrect generalization will degrade the system's performance.

It has not yet been determined the variety of different incomplete and/or incorrect domain theories that can be handled by the extensions to OCCAM. For example, the above rule may not be detected as overspecific if it and another overspecific rule *both* contributed to an imperfect schema. The blame assignment in [Pazzani 89b] can only correct schemas that are "close" to explaining the example, where close means that only one inference rule causes all of the differences between schema and example. Further research into advanced blame assignment algorithms would improve the scope and performance of OCCAM (and many other systems) tremendously.

5.1.6 Widmer's system

Overview

Widmer presents a system in [Widmer 89] that combines SBL and EBL in a more finely-meshed integration than the other systems surveyed here, except for ML-SMART and SaranWrap. While the majority of the other systems have distinct phases in which SBL or EBL is taking place, Widmer's system can be viewed as always performing EBL and calling upon SBL to support non-deductive links in the explanation. In effect, it is performing EBL with weaker links.

Widmer identifies three types of links explaining a feature `F` in the explanation tree. The first is the standard deductive link connecting `F` with a set of conditions `Cond` from a rule `F :- Cond`. The second is determination-based analogy, where `F` is explained by reference to another situation having the same combination of features, including `F`. The third type of link explains `F` by finding a similarity between the present situation and another situation that includes `F`; either certain features of the situation indicate that it belongs to a certain class, or a particular rule almost fits the current situation and can be generalized to fit. The latter leads to incremental generalization, while the former is inductive generalization.

Critique

Widmer has produced a good system, one of the five surveyed incremental integrated learning systems[12]. In addressing the incomplete theory problem, this system risks overgeneralizing the domain, but supervision by a human teacher prevents this from occurring.

Unfortunately, this system suffers from the same problem as GEMINI in that it needs a separate set of information to guide its SBL component. In this case, a set of determinations

[12]GEMINI, PLAUSEX, CIGOL, and OCCAM are the other incremental systems.

are required instead of a concept hierarchy[13]. Presumably, these determinations are correct, as Widmer makes no provisions for correcting them. Since determinations can lead to incorrect generalizations, the presence of the teacher here is required if the system is to be prevented from overgeneralizing the domain theory. Removing this oracle from the system would be an interesting extension to this work.

5.1.7 IOE

Overview

Flann and Dietterich introduce IOE, or Induction Over Explanations, in [Flann 89]. IOE is used to specialize an incorrect theory by taking an example of a concept TC and performing EBL, but the resulting explanation is taken as the necessary and sufficient conditions for another concept C. Thus C is a specialization of TC. EBL is not used to operationalize TC, but to correctly define C.

The algorithm is clearer when compared to EBG [MKK-C 86, DeJong 86] and mEBG, or multiple-example EBG. EBG constructs an explanation for why an example is an instance of a target concept and then generalizes this explanation. mEBG constructs an explanation for each one of several examples, finds the largest common subtree among these explanations, and then generalizes this subtree. IOE also generalizes the largest common subtree produced by an mEBG component, but it also finds constraints on the variables in this generalized subtree, in effect finding special cases of the concept definition learned by mEBG. While EBG may learn about one type of cup and mEBG may learn about the features common to all cups, IOE can also learn about all-plastic cups, cups with metal sides, and many other specialized cups.

Critique

Flann and Dietterich state that IOE's space of specializations is larger than EBG's and mEBG's. In fact, IOE's space encompasses the other two's spaces, because IOE uses mEBG's result as the basis for its specializations. Therefore, IOE can produce any result from mEBG, and both in turn can produce any result from EBG.

The authors point out that since IOE is capable of more specialization, its domain theory can be more general than the other methods' domain theories. A more general theory, they argue, is easier to construct, and they demonstrate that IOE can work equally well in a chess domain made by Flann or one made by someone else with no knowledge of IOE. However, they admit that IOE requires more examples than EBG and mEBG, but they show that the amount is not excessive and can be quite small in domains with many

[13]No concept hierarchy implies that Widmer's SBL component must make exact matches, another limitation (but one shared by most integrated systems).

uniformly-distributed values for each feature.

The authors neglected to mention that IOE assumes all relevant features are in the mEBG component's explanation subtree[14]. Since the domain theory is assumed to be incorrect, some negative examples may be explained by this subtree, and the distinguishing feature of these negative examples may be an unused feature. For example, suppose the target concept TC is defined as

```
object(X) :- ball(X).
```

and the desired specialized concept C is

```
object(X) :- ball(X), red(X).
```

Some positive examples of C are given as

```
ball(obj1), red(obj1).
ball(obj2), red(obj2).
ball(obj3), red(obj3).
```

IOE would explain this collection as all being balls (so C equals TC in this trivial case) which is an overgeneralization of C. The unused feature `red` must be included to correct the definition of C. However, since it does not appear in TC, IOE will not add it.

5.1.8 IOU

Overview

Mooney and Ourston [Mooney 89] describe a system combining EBL and SBL to specialize an incorrect theory. The idea behind IOU (Induction Over the Unexplained) is straightforward - use SBL to remove negative examples that are explained as positive by the overgeneral domain theory. The algorithm is also straightforward. After disjunctively combining the explanations (from the EBL component) for the positive examples, any negative example *not* explained by this combination is discarded, as its already distinguished as being negative. The positive and remaining negative examples are passed to the SBL component, which tries to correctly classify them using the features[15] that do not appear in the explanations. Those features that distinguish positive from negative examples are conjunctively added to the explanations formed by the EBL component, thus specializing the theory.

[14]Notice that IOU makes the opposite assumption. It looks for additional features among the unused features instead. GEMINI makes the same assumption as IOE, but GEMINI tries to fix an incomplete theory, not an incorrect one.

[15]Features in IOU are propositional only. No predicates are used.

Critique

Although their MLW89 paper is not clear, the EBL component appears to send the SBL component a list of features such that each feature is missing from every explanation. The alternative is to send SBL a list of features such that each feature is missing from any one of the explanations, but this alternative will probably send most or all of the features to SBL.

So assume the first case - features missing from every explanation go on to SBL. This prunes a lot of features, but examples can be generated that IOU cannot handle. Say a target concept is

```
object :- red ball or blue cube
```

and the overly general domain theory defines object as

```
object :- ball or cube
```

Examples are classified as positive (+) or negative (-) by a teacher:

```
+: red ball
+: blue cube
-: red cube
-: blue meanie
```

IOU's EBL component classifies the first three as objects. The color features are missing from every explanation, so they are passed to SBL:

```
+: red
+: blue
-: red
```

No color pattern distinguishes positive examples from negative ones, and IOU fails. In fact, IOU will fail in any situation where the nonexplanatory features that distinguish positive from negative examples are not true for every positive example.

5.1.9 UNIMEM

Overview

UNIMEM [Lebowitz 86] is one of the original integrated learning systems (cf. OCCAM). Whereas many of the systems described here perform EBL followed by SBL, Lebowitz chooses to first perform SBL, and then run EBL over the generalizations that SBL produced rather than over the original instances. This method avoids running EBL on atypical cases, as the SBL component will not incorporate them.

UNIMEM first runs its SBL component over the data, extracting predictability information from it. This information takes the form of predictive features, which are likely to be causes in the explanation to be produced. The SBL component can find these predictive features using correlative measures. For example, if A always appears in generalizations with B, but B appears in some generalizations without A, then B cannot cause A because then A would appear wherever B appears. But A *may* be a cause of B, making A a predictive feature. Thus, the SBL component not only makes a generalization to describe some data, but it indicates which of the features in the generalization may be causes and which are effects.

The EBL component can use this set of predictive features, along with a heuristic set of domain rules, in a causal explanation (e.g., using A as an "explanation" for B). Since the EBL component is guided by the SBL component, the domain theory it uses does not need to be perfect, and in fact can be incorrect and inconsistent. Each predictive feature may or may not explain nonpredictive features in the generalization, through the use of the domain theory. If any predictive feature is not used as a cause, then UNIMEM will check to see if it is explainable with the other predictive features. In this manner, coincidental correlations in the data are corrected.

Critique

Lebowitz admits that this SBL/EBL method has some problems, most noticeably with conjunctions of predictive features. For example, if three features together predict a fourth, but none alone predict the fourth, then the SBL component can not find this predictive conjunction. Since the three features can occur individually without the fourth, they will not be seen as causes of that feature. This reduces the scope of applicability of UNIMEM tremendously, since such important domains as medical diagnosis rely on multi-causal explanations.

Lastly, UNIMEM unfortunately does not correct the domain theories it uses. Its SBL component can guide the EBL component away from the incorrect and inconsistent rules by focusing it on the predictive features, but those rules exist nonetheless. In a domain with noisy data, the SBL component could direct the EBL component to use an incorrect rule, leading to poor generalizations.

5.2 Multiple-explanation learners

These systems do not assume that the domain theory supports only one explanation. The first system, PLAUSEX, assumes that the domain theory is incomplete and tries to generalize it by adding rules formed from the explanations. It is capable of numerically ranking the different explanations and selecting the best one for rule formation. The last two systems, ML-SMART and SaranWrap, use tight integrations of EBL and SBL and can work with any kind of imperfect domain theory.

5.2.1 PLAUSEX

Overview

Fawcett's system [Fawcett 89], called PLAUSEX, is one of the two systems described here that uses abduction. The author describes an incremental method of learning, combining abduction and induction and using incomplete domain theories[16]. The algorithm consists of generating all possible partial explanations (i.e., explanations with unproven antecedents) for an example, and then ranking them using relative and absolute measures. The highest-ranked partial explanation is then used to inductively learn a new rule for each unproven antecedent. The left-hand side of the rule is the unproven antecedent, and the right-hand side are all of the unused features in the example, that is those features *not* used in the explanation. After several new rules have been added to the rule base, inductive generalization can be used to remove irrelevant features from the rules by removing right-hand terms that do not appear in all of the rules. PLAUSEX does not yet perform this last inductive step.

Of particular interest is the relative and absolute measures Fawcett uses to rank the partial explanations. Two absolute measures are given: don't accept an explanation that cannot prove particular "special" antecedents (like `is_a`) and don't accept an explanation that leaves unproven an antecedent that another explanation can prove. For example, if a complete explanation has been found for the antecedent `is_stable`, then PLAUSEX will reject any explanation that contains a partial explanation of `is_stable`.

The four relative measures are to prefer explanation X over explanation Y if: X uses fewer rules than Y, X uses more example features than Y, X contains fewer unproven antecedents than Y, and X uses more specific rules than Y. The first three measures are bound up in a numerical evaluation function $f(x)$ (where x is each antecedent) that can rank partial explanations.

Critique

Fawcett states that a large amount of pruning is performed by the second absolute measure, that of rejecting partial explanations that leave an antecedent unproved when a complete explanation for it has been found elsewhere. This may cause a problem if

[16]PLAUSEX can actually use more of the incomplete domain space than other systems, since it can use domain rules that only partially explain a positive example, while other systems that use deduction need to build a full explanation to be able to take advantage of those rules.

PLAUSEX is working on a simple binary explanation tree and can ground the left subtree or the right subtree but not both. A system like AbE would produce 2 partial explanations, with the first having the left subtree grounded and the right unexplained, and the second having the right subtree grounded and the left one unexplained. PLAUSEX would produce nothing, which may or may not be acceptable. Of course, this example can be generalized to n partial explanations with n subtrees, and even worse pathological examples can be created. The point is that this heuristic may prevent PLAUSEX from exploring parts of the explanation space that contain desirable partial explanations.

The problems for PLAUSEX grow worse still. Fawcett claims that his work addresses the incomplete theory problem identified in [MKK-C 86]. He states that it is not assumed that every observable feature is provided to the system. From these statements, it can be shown that his system may solve the incomplete theory problem by transforming the domain theory into an incorrect theory instead. Thus, Fawcett will also need to address this problem.

Incorrectness is introduced into the domain theory in the following manner. Suppose a domain theory for cups is incomplete and does not possess a definition for stable(X). Suppose the true definition is

```
stable(X) :- part_of(X,B), is-a(B,bottom), flat(B).
```

but the system is never given `flat(B)` as an observed feature (which is one of Fawcett's assumptions, that every observable feature is not provided). After several explanations involving `stable(X)`, the system may have induced several rules such as

```
stable(X) :- part_of(X,B), is-a(B,bottom), color(B,white).
stable(X) :- part_of(X,B), is-a(B,bottom), color(B,red).
stable(X) :- part_of(X,B), is-a(B,bottom), material(X,ceramic).
```

that may be generalized into

```
stable(X) :- part_of(X,B), is-a(B,bottom).
```

Although this rule will correctly identify actual stable cups, it is too general and will recognize many unstable cups. This rule makes the domain theory *incorrect*, and PLAUSEX is designed to complete the domain theory, not correct it.

5.2.2 ML-SMART

Overview

This system tightly integrates SBL and EBL. Rather than perform the inductive and deductive steps in distinct phases, ML-SMART performs EBL and calls upon SBL to help it with imperfections in its domain theory. ML-SMART can thus handle theories that are incomplete, incorrect, and inconsistent.

Unlike standard EBL algorithms that need only one positive example, ML-SMART requires a training set of positive and negative examples. The system runs until it has gathered a group of explanations that explain all of the positive examples but none of the negative ones. In this fashion, ML-SMART is capable of working with domain theories that can explain an example in multiple ways. In a sense, the system only does specialization, but gathers enough different explanations to cover all of the positive examples.

The learning algorithm works as a form of best-first search that tries to find the smallest set of explanations that cover the positive examples. Starting with the nonoperational predicate to be learned, at each step ML-SMART replaces a nonoperational predicate with its antecedents from the domain theory. If more than one rule exists, then all sets of antecedents are tried. If a new explanation is overgeneral and explains some negative examples, it can be specialized in a variety of ways, including adding terms or making variables into constants to discriminate positive from negative examples. An explanation is ranked according to how many positive and negative examples it explains and the number of operational and nonoperational predicates it contains.

If, at any time, one of the frontier[17] explanations explains only positive examples, then it is marked as consistent[18] and is no longer expanded. When enough of these consistent explanations have been found to cover all of the given positive examples and none of the remaining frontier explanations look promising, then ML-SMART halts, having found a complete and correct definition of the nonoperational predicate to be learned.

Critique

The system is described in too little detail to be readily critiqued. Search control is so poorly explained that their example in the cups domain appears contrived. Other issues that should have been explored include testing the effects of different domain theories (complete and correct, only incomplete, only incorrect, etc.) on the size of the search tree, and using domain theories where multiple explanations can be correct but not necessarily inconsistent. Also, the system is nonincremental in that every training example must be provided and classified before ML-SMART will begin, unlike Widmer's system that can incrementally complete a domain theory.

[17]ML-SMART is a best-first search algorithm. Therefore, an ordered frontier of unexpanded nodes is maintained.

[18]This means consistent with the examples, not the domain theory.

5.2.3 SaranWrap

Overview

Pazzani and Kibler [Pazzani 90] introduce SaranWrap, an extension to Quinlan's inductive FOIL program [Quinlan, in press]. FOIL constructs a domain theory for a given set of positive and negative instances by creating clauses that explain all of the positive examples but none of the negative ones. These clauses are produced by starting with the trivial clause *true* and specializing it to cover only a subset of the positive examples. Specialization of the clause is performed by adding variabilized predicates that maximize information gain. The disjunction of all of the clauses forms a domain theory that explains all of and only the positive examples (i.e., the theory is complete and correct and therefore consistent).

Saranwrap extends FOIL in two ways, both of which involve limiting its search during the extension of a clause. The first extension uses semantic constraints, such as type constraints and multiple-argument constraints (all of the variables in a predicate must differ). The second extension gives FOIL the ability to exploit partial domain theories, with both operational and nonoperational predicates and with nonoperational concept definitions. SaranWrap uses information gain to determine whether to accept, specialize, or delete predicates in the supplied clauses. The information gain metric also guides the operationalization of predicates.

Critique

It is interesting to note that, given a complete and correct domain theory, SaranWrap will simply accept each clause verbatim. It will not attempt to "fix" a domain theory that is not broken[19]. But given no domain theory at all, SaranWrap will perform a hill-climbing inductive search, guided by its information gain metric. Thus, SaranWrap resembles OCCAM in that it too prefers EBL over SBL, but can operate without a perfect (or even partial) domain theory if it must.

SaranWrap is very much like ML-SMART. Both systems are given a set of positive and negative examples, and search for a set of clauses (or explanations, if you will) that cover all of the positive examples and none of the negative ones. Of the two systems, SaranWrap appears to have a better search control and is explained more clearly than ML-SMART. On the other hand, ML-SMART uses best-first search rather than hill-climbing. Theoretically it can find better sets to cover the positive examples than SaranWrap, at the expense of more memory devoted to the search operation in the form of a search tree.

An incremental SaranWrap would be an interesting system. This would require a substantial change in SaranWrap's current algorithm, including its information gain metric.

[19]Unlike IOU, for instance.

However, such tightly integrated incremental systems are possible, as we saw in Widmer's system.

5.3 Summary

A classification by domain assumptions of the twelve integrated systems is shown in Figure 4 below. Nine systems make the single-explanation assumption, and six of these nine systems assume the domain is incomplete. Interestingly, all of the incremental learners (see Table 3) assume the domain theory is incomplete, and of the seven systems that assume incomplete theories (the six single-explanations systems and PLAUSEX), five systems are incremental.

Of the twelve systems, ten use a deductive-inductive combination, and eight of these use deduction before induction, a marked preference to use EBL before SBL. The largest group of these systems are nonincremental deductive-inductive learners that assume their domain theory is incomplete and that it supports only a single explanation for a given example.

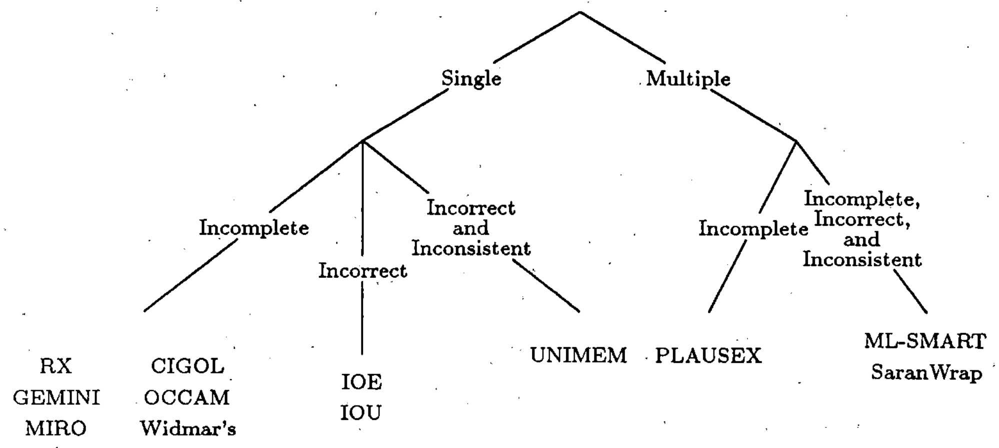

Figure 4: The surveyed systems classified by domain assumptions

Only two of the systems, PLAUSEX and CIGOL, use abduction, and both combine it with inductive inference. Both have unique capabilities that come directly from their abductive components. PLAUSEX can generate partial explanations that allow it to use certain incomplete domain theories that some of the other deductive systems can not. CIGOL can invent new predicates as hypotheses of what may explain its examples, and can therefore extend its representation language. The success of these two systems implies a usefulness of abduction that awaits exploitation when combined with the other inference methods.

The majority of these systems use one technique, usually EBL, followed by another, usually SBL, in distinct phases. Four systems (marked with a † in Table 3) use a tightly integrated approach in which both kinds of inference may be used at each step in the algorithm. While the tightly integrated deductive-inductive learners have a preference for deduction, two of these systems, ML-SMART and SaranWrap, can use any kind of imperfect domain theory, even empty theories where deduction is useless.

These two systems also highlight the problem of trying to perfect incomplete domain theories without also tolerating incorrect theories. Some of the systems, such as PLAUSEX and OCCAM, transform incomplete theories into incorrect ones, that they are then unable to fix. Other systems, such as IOE and IOU, cannot tolerate *all* incorrect theories, only a proper subset. Both of these problems stem from the use of an SBL component, with the former systems using the component to generalize theories and the latter performing specialization. In other words, the low security of induction can lead to poorly developed theories no matter what the condition of the original domain theory. At present, complete supervision, as found in ML-SMART and SaranWrap, is the only solution to this problem in integrated learning systems.

Flann and Dietterich comment in [Flann 89] that incorrect domain theories are easier to create than incomplete theories. Essentially, it is simpler to produce an overgeneral theory that may cover some negative examples than an overspecific theory that covers absolutely no negative ones. Unfortunately, the majority of the integrated learning systems here are designed for incomplete theories rather than incorrect ones.

Finally, none of the systems use all three inference methods as defined here. RX would be the easiest system in which to add a third inference technique, but the addition of an abductive component would change the nature of the system from a knowledge acquisition program to a diagnostic performance system. The other systems would be more difficult, but the successful pairwise combinations of the inference methods suggest that a triply integrated system could be quite powerful. Such a system, if classified like the others, might appear as in Table 4. The DUCTOR would consist of a tight integration of abduction, deduction, and induction and be capable of using and correcting any kind of imperfect domain theory.

Main author	System	Combines	Supervised/ Incremental	Domain assumptions
Cain	DUCTOR	A-D-I†	Unsuper/Increm	Incom, Incor, Incon, Mult

Table 4: A hypothetical system

The DUCTOR will be incremental and unsupervised. These topics have not been discussed in depth in this paper, mainly because advantageous combination of learning strategies appears to be the main direction of integrated learners at this time. GEMINI, PLAUSEX, CIGOL, OCCAM, and Widmer's system are the incremental systems surveyed here, while RX, GEMINI, UNIMEM, and OCCAM are all unsupervised programs. The advantages of incremental and unsupervised learning are numerous, but the effects caused by different combinations of inference methods in these systems are more interesting.

Until recently, no integrated learners have been compared, either formally or empirically, because they appeared to be solving different problems. With the framework provided here, integrated learning systems can be classified by inference method and domain theory assumptions, and similar systems can be compared theoretically and by their performance on learning tasks.

6 Conclusion

This paper presented a framework for integrated learning based on the three basic forms of inference - deduction, induction, and abduction. A dozen integrated learning systems were examined and critiqued with this framework in mind, and a summary of the systems was provided based on their similarities and differences. Finally, the suggestion for a triply integrated system was made in the hopes that several past successes with doubly integrated learning have paved the way for more complex and more powerful integrated learners.

7 Acknowledgements

Discussions with Michael Pazzani, David Ruby, Wendy Sarrett, Steve Morris, Kamal Ali, and David Aha were invaluable. Their criticism vastly improved the quality of this paper.

References

[Bergadano 88a] Bergadano, Francesco and Giordana, Attilio. "A Knowledge Intensive Approach to Concept Induction." In *Proceedings of the Fifth International Workshop on Machine Learning*, pp. 305-317. Ann Arbor: Morgan Kaufmann, Inc., 1988.

[Bergadano 88b] Bergadano, Francesco, Giordana, Attilio and Saitta, Lorenza. "Automated Concept Acquisition in Noisy Environments." In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, pp. 555-578, July 1988.

These papers present a framework called ML-SMART for integrating similarity-based and explanation-based learning. In general, the system searches for concept descriptions, starting from a set of positive and negative instances and guided by statistics, heuristics, and domain theory. Taken alone, the statistics lead to an inductive best-first search, the heuristics drive the search in domain-independent directions (such as specializing formulas that have few alternative bindings or that are easy to read by humans), and the domain theory can be used to order the search towards promising areas of the tree.

[Blum 86] Blum, Robert. "Computer-Assisted Design of Studies Using Routine Clinical Data." In *Annals of Internal Medicine*, Vol. 104, pp. 858-868, 1986.

[Blum 82] Blum, Robert. "Relationships from a Large Time-Oriented Clinical Database: The RX Project." In *Computers in Biomedical Research*, Vol. 2, pp. 164-187, 1982.

Both of these papers describe the RX system, a set of programs for discovering and confirming medical hypotheses with the use a large database. The system follows a model of clinical research that involves 4 modules: a discovery process, a study module, a statistical analysis package, and a knowledge base. The discovery module looks for relationships among primary and derivable attributes of a patient's record, examining the attributes in pairs, looking for correlations (an $O(n^2)$ process). The study accepts a hypothesis from the discovery module (or directly from a human researcher) and generates a model of the hypothesis that is plausible to test based on the medical and statistical knowledge in the database.

[Cain 89] Cain, Timothy. "The DUCTOR: A System for Scientific Discovery." Unpublished manuscript, 1989.

This paper describes the DUCTOR, a discovery system that combines inductive and deductive components. The inductive component of the system is BACON (cf. [Langley 87], with relaxed correlative detectors so that some exceptions to rules are allowed. The deductive component tests the laws produced, and when laws fail, it can suggest variables upon which the inductive component should start its search for new laws.

[Danyluk 87] Danyluk, Andrea P., "The use of explanations for similarity-based learning", *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pp. 274-276. Milan, Italy: Morgan Kaufman, 1987.

This paper briefly explains an approach to integrate explanation-based learning and similarity-based learning. A learner/analyzer module is described that can generalize and explain an event by using a rule base, a concept hierarchy, and an incident hierarchy (of previously generalized events). When the EBL phase is completed, its explanation provides the SBL phase with a list of features to be examined. For example, if the explanation of a bombing involved the location of the bombing, then the SBL phase will compare the location of two events (as well as other features indicated in the EBL explanation) when performing a match. Similarly, the SBL phase can produce information such as possible causal direction (c.f. [Lebowitz 86]) which is useful to the EBL phase. Danyluk also points out that the SBL phase can handle inexact matches, unlike other systems that integrate empirical and explanation-based learning.

[DeJong 86] DeJong, Gerry and Mooney, Raymond. "Explanation-based Learning: An Alternative View." In *Machine Learning 1*, pp. 145-176, 1986.

The authors present a slightly different view of explanation-based learning than found in [MKK-C 86]. In addition to identifying a problem of undergeneralization in that paper's EBG mechanism and criticizing their notion of operationality criteria, the authors also advocate a different mechanism for generalization and a broader name for these kinds of algorithms (explanation-based learning (EBL) rather than EBG). They also conclude that an EBG mechanism should allow for learning from observation, rather than just learning from internal planning. A complete version of their method is shown performing on a kidnapping example.

[Dietterich 89] Dietterich, Thomas G. "Limitations on Inductive Learning." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 124-128. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

Dietterich proves in this paper that inductive learning from examples can only learn a small fraction of the total number of hypotheses in a given space. Dietterich introduces the concept of FAC(frequently approximately correct)-learning, and provides proofs for the upper bounds on the number of concepts FAC-learnable from a given number of examples. He then shows that three inductive learning algorithms (ID3, backpropagation, and the classical algorithm for computing the maximally specific conjunctive generalizations) learn significantly less than the proven upper bound (the upper bound was 88 concepts, and the algorithms learned 8, 2, and 10 respectively). This suggests that either the upper bound can be reduced or the present algorithms improved.

[Dietterich 86] Dietterich, Thomas G. "Learning at the Knowledge Level." In *Machine Learning 1*, pp. 287-316, 1986.

This paper describes various learning systems in terms of their learning at the knowledge level. Roughly speaking, a system is said to learn at the knowledge level if its set of sentences with inferrable truth values increases over time. Some systems do not learn at the knowledge level. They improve their performance over time (i.e., they get faster), but their knowledge set is static. Other systems can increase their knowledge set in a deductive way, by taking input from the environment and "learning" everything in the deductive closure of its new knowledge set. Finally, some systems increase their knowledge in a manner not describable at the knowledge level. These systems make unjustified nondeductive leaps. While a theory of symbol level learning may be developed, a theory of nondeductive knowledge level learning may be difficult to develop, primarily due to the lack of a model of plausible reasoning.

[Drastal 89] Drastal, George, Czako, Gabor, and Raatz, Stan. "Induction in an Abstraction Space: A Form of Constructive Induction." In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 708-712. Detroit: Morgan Kaufmann, Inc., 1989.

This paper describes MIRO, a system that combines deduction and induction for concept formation. This system uses deduction to form an abstraction space. It then performs induction in this space rather than the original space.

[Falk 86] Falkenhainer, Brian C., & Michalski, Ryszard S. "Integrating Quantitative and Qualitative Discovery: The ABACUS System", *Machine Learning 1*, pp. 377-401. Boston, MA: Kluwer Academic Publishers, 1986.

This paper introduces ABACUS, a machine discovery program that combines qualitative and quantitative inductive learning. Discovery is performed in two steps. First, the equation discovery modules finds equations that summarize the given data. Then the precondition generation module derives a logical expression that describes when the equations are to be used. The result of these two modules are "if-then"-type rules that cover different parts of the data. Several artificial experiments demonstrate ABACUS's discovery capabilities, and one real experiment on a chemical problem is presented (with good results, too).

[Falk 88] Falkenhainer, Brian & Rajamoney, Shankar. "The Interdependencies of Theory Formation, Revision, and Experimentation." In *Proceedings of the Fifth International Workshop on Machine Learning*, pp. 353-366. Ann Arbor, MI: Morgan Kaufmann, Inc., 1988.

This paper unifies Falkenhainer's PHINEAS system, which performs verification-based analogical learning, with Rajamoney's ADEPT system, which performs experimentation-based theory revision. The resulting combination can form hypotheses about the state of the world, run experiments, and revise its theories.

[Fann 70] Fann, K. T. *Peirce's theory of abduction.* The Hague: Martinus Nijhoff, 1970.

This book describes the theory of abduction put forth by Charles Peirce. It is especially useful in understanding the changes in the theory over time, reflecting Peirce's ideas on the nature and scope of abduction.

[Fawcett 89] Fawcett, Tom E. "Learning from Plausible Explanations." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 37-39. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper describes a method of learning using incomplete domain theories. The algorithm generates all possible partial explanations (i.e., explanations with unproved antecedents) for an example, and then ranks them using relative and absolute measures. The highest-ranked partial explanation is then used in the SBL component by creating a maximally specific rule for each unproved antecedent using the features in the example, generalizing this rule, and adding it to the rule base.

[Flann 89] Flann, Nicholas and Dietterich, Thomas. "A Study of Explanation-Based Methods for Inductive Learning." In *Machine Learning 4*, pp. 187-226, 1989.

This paper introduces a new learning method called induction over explanations (IOE) and compares it to explanation-based generalization (EBG) and multiple example EBG (mEBG). EBG uses one example to produce an explanation of a concept that includes that example. mEBG produces an explanation tree for each of many examples, and then finds the largest subtree shared by all of the trees and generalizes this subtree. IOE also generalizes the largest common subtree produced by an mEBG component, but it also finds constraints on the variables in this generalized subtree, in effect finding special cases of the concept definition learned by mEBG.

[Ginsberg 88a] Ginsberg, Allen. "Knowledge-Base Reduction: A New Approach to Checking Knowledge Bases for Inconsistency & Redundancy." In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 585-589. St. Paul: Morgan Kaufmann, Inc., 1988.

This paper describes a new technique for detecting inconsistent and redundant rules in a knowledge base. Inconsistent rules are those rules which produce contradictory results given valid input data (i.e., data that itself does not violate semantic constraints, like a man being pregnant). Redundant rules are those which do not follow from other rules and can be satisfied by some valid input data. The system, KB-Reducer, generates the minimal input sets that would cause the knowledge base to assert some conclusion and examines these results. While computationally intractable in the worst case, KB-Reducer has been found empirically to have an acceptable run-time on "real-world" domains.

[Ginsberg 88b] Ginsberg, Allen. "Theory Revision via Prior Operationalization." In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 590-595. St. Paul: Morgan Kaufmann, Inc., 1988.

The paper describes a three-phase process of theory revision and presents results from the testing of the first two phases.The first phase of theory revision translates the theory by effectively completely operationalizing it, as if every example had been presented to it. This step is performed by KB-Reducer [Ginsberg 88a]. Next, an inductive component takes this reduced theory and alters it to match the data. Finally, the new reduced theory is retranslated back into the language of the original theory. Empirical tests show this method improved a domain theory by 17 to 27%.

[Hirsh 89] Hirsh, Haym. "Combining Empirical and Analytical Learning with Version Spaces." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 29-33. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper describes how version spaces might be used to create an integrated learning system. In brief, version space algorithms learn concepts by maintaining two boundary sets, S and G, which contain the most specific and the most general concept descriptions, respectively. Like MIRO [Drastal 89] and IOE [Flann 89], Hirsh's system first runs its deductive component on the data to create generalized instances. Then the system runs an incremental version space merging algorithm as its empirical learner. Hirsh shows how this method can work with no domain theory or an incomplete or inconsistent one.

[Kulkarni 88] Kulkarni, Deepak and Simon, Herbert A.. "The Processes of Scientific Discovery: The Strategy of Experimentation." In *Cognitive Science* **12**, pp. 139-175, 1988.

This paper introduces KEKADA, which models Hans Krebs discovery of the urea cycle. KEKADA can form hypotheses and carry out experiments, and reacts to surprises from these experiments. In general, KEKADA and its domain-independent heuristics are a model of scientific experimentation in many fields besides biochemistry.

[Langley 87] Langley, P., Simon, H. A., Bradshaw, G. L., & Zytkow, J. M. *Scientific Discovery.* Cambridge, MA: MIT Press, 1987.

This book describes the scientific discovery process and how weak learning methods in AI may be used in this process. It is an excellent source for descriptions of the progessive stages of BACON, a purely-inductive scientific discovery program which finds quantitative laws that cover empirical data. Other systems in the book include GLAUBER, which focuses on qualitative relationships, STAHL, and DALTON.

[Langley 89] Langley, Pat. "Unifying Themes in Empirical and Explanation-Based Learning." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 2-4. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This opening paper from the integrated learning session at MLW89 tries to illuminate the similarities between empirical and explanation-based learning rather than their differences. These comparisons were made along five lines, in which the popular belief of the two methods' differences is replaced with a claim of their underlying similarity. The five comparisons are along number of instances, use of search, use of domain knowledge, justification of method, and accuracy/efficiency measures.

[Lebowitz 86] Lebowitz, Michael. "Integrated Learning: Controlling Explanation." In *Cognitive Science*, Vol. 10, 1986.

This paper proposes combining explanation-based learning and similarity-based learning by using EBL techniques *during* SBL, at times when a number of confident generalizations have been produced. This method reduces the amount of use of EBL, which is computationally expensive, restricting its use to situations in which the resulting explanations will probably be sound.

[Michalski 89] Michalski, Ryszard S. "Multistrategy Constructive Learning: Toward a Unified Theory of Learning." In *Proceedings of the ONR Workshop on Knowledge Acquisition.* Arlington, VA, November 6-7, 1989.

In this paper Michalski presents his version of an inference-based theory of learning, which can serve as a framework for clarifying relationships among different learning methods. Michalski uses the theory as a basis for an integrated learning methodology called *multistrategy constructive learning.* The basic idea behind inference-based learning is that any input affects the learner's background knowledge through one of three inference methods (deduction, induction, or analogy). Learning is only possible by performing inference and by having a memory in which to store the results of inference and to store background knowledge. A particular learning strategy is selected by a function of the input, the background knowledge, and the learner's current goal. Michalski describes how the different strategies would learn various aspects of the concept of a cup, given different starting information and goals.

[MKK-C 86] Mitchell, T., Keller, R., & Kedar-Cabelli, S. "Explanation-based Generalization: A Unifying View." In *Machine Learning 1*, pp. 47-80, 1986.

In this seminal paper, the authors describe how EBG (explanation-based generalization) can be used to deductively derive generalizations using a single training example. This is, of course, the now-standard method of operationalizing a concept by building an explanation through backward-chaining in the domain theory rules and generalizing the resulting explanation by regressing the goal concept through the explanation. The generalized explanation is then compressed by removing all internal nodes,

expressing the concept as the deductive result of the conjunction of the leaf expressions. Several examples of EBG are presented, and several important research issues are discussed regarding the effect of imperfect theories on EBG, including the problems caused by incomplete, intractable, and inconsistent domain theories. The importance of combining EBG and empirical methods is also discussed. Finally, the authors suggest research into the area of automatically determining what needs to be learned to improve the performance of EBG.

Note: there is a problem with their generalization algorithm. See [DeJong 86].

[Mooney 89] Mooney, Raymond and Ourston, Dirk. "Induction Over the Unexplained: Integrated Learning of Concepts with Both Explainable and Conventional Aspects." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 5-7. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper describes an approach for integrating EBL and SBL which uses EBL to add explainable features to the target concept and SBL to add any correlated features. IOU finds the correct concept description with fewer examples than purely empirical learning systems, such as ID3. IOU can find correct concept descriptions in cases where other integrated learning systems, such as IOE, cannot, due to the presence of conventional (nonfunctional) features in the correct concept descriptions.

[Muggleton 88] Muggleton, Stephen and Buntine, Wray. "Machine Invention of First-order Predicates by Inverting Resolution." In *Proceedings of the Fifth International Workshop on Machine Learning*, pp. 339-352. Ann Arbor, MI: Morgan Kaufmann, Inc., 1988.

This paper describes CIGOL, an inductive learning program that is capable of generating and generalizing new predicates and thus define its own vocabulary for concept descriptions. The basis of CIGOL is a method of inverting resolution. While normal resolution is used to deduce the consequences of a set of laws, inverse resolution induces rules from examples. CIGOL uses three operators in performing inverse resolution: "V" operators take their base clause and an arm clause, and produce the other arm clause. "W" operators find a clause A such that A resolves on a common literal L with two other clauses C_1 and C_2 to form B_1 and B_2. Since L is resolved away and does not appear in B_1 or B_2, "W" operators introduce new predicates. The third operator type is truncation, which handles the case when the base clause(s) of "V" and "W" operators are empty.

[Muggleton 89] Muggleton, Stephen, Bain, Michael, Hayes-Michie, Jean, and Michie, Donald. "An Experimental Comparison of Human and Machine Learning Formalisms." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 113-118. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper descibes a test consisting of CIGOL versus several other machine learning programs and versus human agents. Several experiments were conducted using different sizes of training sets and different amounts of background knowledge for solving chess endgame problems. CIGOL performed very well, on par with the human subjects, but like them it was not tested on large training sets, as it runs too slowly. Although the test did not involve CIGOL generating any new predicates, it does present CIGOL running in a non-interactive mode.

[O'Rorke 89a] O'Rorke, Paul, Morris, Steve, and Schulenburg, David. "Theory Formation by Abduction: Initial results of a Case Study Based on the Chemical Revolution." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 266-271. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

[O'Rorke 89b] O'Rorke, Paul, Cain, Timothy, and Ortony, Andrew. "Learning to Recognize Plans Involving Affect." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 209-211. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

These two papers are included as examples of systems, AMAL and AbE, that use abduction. Both are PROLOG meta-interpreters that construct explanation trees, but AbE subsumes AMAL and can also evaluate partial explanation trees and perform a best-first search through the space of explanations.

[Pazzani 86] Pazzani, Michael, Dyer, Michael, and Flowers, Margot. "The Role of Prior Causal Theories in Generalization." In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pp. 545-550. Philadelphia: Morgan Kaufmann, Inc., 1986.

This paper describes OCCAM and its preference to use prior causal theories over correlation when it makes generalizations of incoming data on events. A number of experiments are reviewed which indicate that humans (and possibly animals) share this preference, as it can result in faster and less memory-intensive learning. OCCAM's basic algorithm is to input each new event, and explain why it is similar to other events which share its most specific generalization. If prior causal theories confirm this explanation, then a new explanatory generalization is created. If they deny the explanation, then it is discarded. If the prior causal theories neither confirm nor deny the explanation, then it is marked as a tentative generalization. If a future event contradicts the tentative generalization, it will be discarded. Otherwise, a variety of strategies can be employed to confirm the tentative generalization, including increasing confidence as it successfully predicts new events until some confirmatory threshold is reached.

[Pazzani 88] Pazzani, Michael. "Integrated Learning with Incorrect and Incomplete Theories." In *Proceedings of the Fifth International Workshop on Machine Learning*, pp. 291-297. Ann Arbor: Morgan Kaufmann, Inc., 1988.

This paper describes two extensions to OCCAM since its description in [Pazzani 86]. The first extension allows OCCAM to work with incorrect domain theories by revising its explanatory generalizations, or schemas, when those schemas produce incorrect predictions. The second extension allows OCCAM to work with incomplete domain theories by using an empirical learning approach that can create the missing explanatory link after seeing several examples.

[Pazzani 89b] Pazzani, Michael. "Detecting and Correcting Errors of Omission After Explanation-based Learning." In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pp. 713-718. Detroit: Morgan Kaufmann, Inc., 1989.

This paper describes an extension to OCCAM that can detect errors caused by an overspecific (i.e., incomplete) domain theory. OCCAM can determine the rule or rules at fault and revise them to account for the new example. This extension differs from the ones presented in [Pazzani 88] by its focus on performance examples rather than examples used to acquire background knowledge. Experimental results show that OCCAM with the extension can significantly improve its performance when using an incomplete domain theory.

[Pazzani 90] Pazzani, Michael and Kibler, Dennis. "The Utility of Knowledge in Inductive Learning." Unpublished manuscript, 1990.

This paper presents SaranWrap, an extension to Quinlan's FOIL program [Quinlan, in press]. FOIL inductively builds a domain theory given a set of positive and negative examples of a concept. SaranWrap improves

FOIL's performance by limiting the search it performs to find the clauses of this new domain theory. The extensions can be classed as incorporating semantic constraints and giving FOIL the ability to use partial (and possibly incorrect) domain theory knowledge.

[Pazzani,in press] Pazzani, Michael. "A Computational Theory of Learning Causal Relationships." In *Cognitive Science Journal.*

This paper introduces a theory of learning called theory-driven learning (TDL), which produces a theory of causation from a set of observations and theory of causality. The paper distinguishes between theories of causation and theories of causality. A theory of causation predicts state changes using a set of specific causal relationships between actions and states. A theory of causality specifies how to learn the causal relationships themselves. For example, if an action on an object is followed by a state change in the object, causality theory suggests that the action resulted in the state change. Causality theory aids the learner by focusing attention on important actions (i.e., ones that have effects). An subject experiment indicated that humans learn faster if the relationship to be learned is consistent with their theory of causality. TDL is more limited than similarity-based learning in terms of what it can learn, but TDL learns faster. EBL is faster and more accurate than TDL, but only if it is provided with a complete theory of causation. TDL is capable of providing EBL with a theory of causation, making the two forms of learning complementary.

[Pople 73] Pople, Jr., Harry E., "On the Mechanization of Abductive Logic." In *Proceedings of the Third International Conference on Artificial Intelligence*, pp. 147-152. Stanford: William Kaufmann, Inc., 1973.

This is one of the earliest papers on the use of abduction as an automated inferencing method for learning. Pople describes abduction in syllogistic terms, comparing it to the other two fundamental inference methods, deduction and induction. He then explains how a deductive linear resolution procedure can be used as an abductive procedure with only minor changes, mainly by retaining certain literals abandoned by deduction and producing alternate explanation structures. Pople provides the classic liver example to illustrate his combined deductive/abductive algorithm.

[Quinlan, in press] Quinlan, J. R. "Learning Logical Definitions from Relations."

[Rajamoney 87] Rajamoney, Shankar and Dejong, Gerald. "The classification, detection, and handling of imperfect theory problems", *Proceedings of the*

Tenth International Joint Conference on Artificial Intelligence, pp. 205-207. Milan, Italy: Morgan Kaufman, 1987.

This paper introduces a finer classification scheme for imperfect theories than the one presented in [MKK-C 86]. Two divisions of incompleteness and inconsistency are provided. Incompleteness type 1 occurs when a deductive proof cannot be completed because knowledge is missing. Incompleteness type 2 occurs when the knowledge is too general and may lead to invalid proofs or no proofs at all. Inconsistency type 1 occurs when the theory contains the wrong knowledge, and inconsistency type 2 occurs when the theory is missing knowledge that would prevent an invalid proof from being derived. Methods of detecting and handling these imperfect theories are discussed.

[Russell 86] Russell, Stuart J., "Preliminary Steps Toward the Automation of Induction." In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pp. 477-484. Philadelphia: Morgan Kaufmann, Inc., 1986.

Russell proposes a theory of induction that bases the plausibility of an inductively-produced rule on the amount of direct and indirect evidence that is associated with it. Direct evidence is the usual set of positive and negative instances that the rule correctly classifies. The indirect evidence comes from a set of higher-level regularities, of which this rule is an instance. Regularities, in turn, are based on their own direct and indirect evidence, i.e. its own set of instances and even higher-level regularities. For example, say you meet several Americans who speak English. This would lead you to suspect that all Americans speak English because of the set of direct evidence and the regularity that people of a particular country speak the same language. Russell presents several classes of regularity already identified and compares his theory of induction to Nelson Goodman's theory of projectibility.

[Sarrett 89] Sarrett, Wendy and Pazzani, Michael. "One-sided Algorithms for Integrating Empirical and Explanation-based Learning." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 26-28. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper describes a combination of SBL and EBL using one-sided algorithms, which never generate a hypothesis that is more general than the target hypothesis. The method of combination, IOSC, is to update the working hypothesis whenever a positive example is misclassified as negative. If EBL can explain the example as positive, then the hypothesis

has those features removed that appear in the example but not in the explanation. Otherwise, SBL removes features from the hypothesis that are not in the example. IOSC has been extended to handle k-CNF.

[Thagard 88] Thagard, Paul. "Explanatory Coherence." Unpublished manuscript, 1988.

This paper presents a theory of explanatory coherence to be used in the ranking of scientific hypotheses and provides an implementation of the theory in a connectionist program called ECHO. This theory consists of seven principles that relate the hypothesis to other propositions (e.g., supporting data, contradictory evidence, etc.). The accepted hypothesis is the one which is more coherent than its competitors. Examples of ECHO's application is provided in the domains of chemistry, biology, and law. Thagard also details the implications of his theory for artificial intelligence, psychology, and philosophy.

[Widmer 89] Widmer, Gerhard. "A Tight Integration of Deductive and Inductive Learning." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 11-13. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

Widmer presents an incremental, supervised learning system that tightly integrates empirical and explanation-based learning. By "tight", Widmer explains that both deduction and induction are used at various times during construction of an explanation, rather than one component performing all of its work and then transferring its results to the other component, which finishes the work. Widmer identifies three types on links explaining a feature `F` in the explanation tree. The first is the standard deductive link connecting `F` with a set of conditions `Cond` from a rule `F :- Cond`. The second is determination-based analogy, where `F` is explained by reference to another situation having the same combination of features, including `F`. The third type of link explains `F` by finding a similarity between the present situation and another situation that includes `F`; either certain features of the situation indicate that it belongs to a certain class, or a particular rule almost fits the current situation and can be generalized to fit. The latter leads to incremental generalization, while the former is inductive generalization.

[Wilkins 88] Wilkins, David. "Knowledge Base Refinement Using Apprenticeship Learning Techniques." In *Proceedings of the Seventh National Con-*

ference on Artificial Intelligence, pp. 646-651. St. Paul: Morgan Kaufmann, Inc., 1988.

This paper describes the semi-automatic correction of knowledge bases through the use of a human expert. The system observes the expert solving a problem (diagnosis is used as an example) and tries to explain every action the human performs. If the system cannot explain an action, then it attempts to learn the knowledge required to complete its explanation. A test showed that the system improved its performance by 42% after observing two diagnostic sessions.

[Wollowski 89] Wollowski, Michael. "A Schema for an Integrated Learning System." In *Proceedings of the Sixth International Workshop on Machine Learning*, pp. 87-89. Ithaca, NY: Morgan Kaufmann, Inc, 1989.

This paper attempts to explore the dichotomies between empirical data versus reasoning and analysis versus synthesis. Wollowski also highlights the difference between empirical knowledge and *a priori* knowledge. He gives a weak argument for why deduction and induction would not form a powerful integrated learning system. Finally, he concludes with an example of why analysis and synthesis is not necessarily always performed by deduction and induction respectively.