To appear in *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Lawrence Erlbaum.

Failure-Driven Learning as Input Bias

Michael T. Cox and **Ashwin Ram**
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
{cox,ashwin}@cc.gatech.edu

Abstract

Self-selection of input examples on the basis of performance failure is a powerful bias for learning systems. The definition of what constitutes a learning bias, however, has been typically restricted to bias provided by the input language, hypothesis language, and preference criteria between competing concept hypotheses. But if bias is taken in the broader context as any basis that provides a preference for one concept change over another, then the paradigm of failure-driven processing indeed provides a bias. Bias is exhibited by the selection of examples from an input stream that are examples of failure; successful performance is filtered out. We show that the degrees of freedom are less in failure-driven learning than in success-driven learning and that learning is facilitated because of this constraint. We also broaden the definition of failure, provide a novel taxonomy of failure causes, and illustrate the interaction of both in a multistrategy learning system called Meta-AQUA.

Introduction

In the absence of a complete and correct domain theory, a system that goes beyond rote learning must perform an inductive leap. An inductive leap must be constrained and guided in some manner (Mitchell, 1990/1980). Bias has been used to constrain the hypothesis space, the criteria for selecting hypotheses, and the language in which examples are represented. However, we echo the sentiment of Provost and Buchanan (1992) when they criticize current trends in formalizing bias as being too narrow. They argue instead that there should be a return to Mitchell's original definition of bias as "any basis for choosing one generalization over another, other than strict consistency with the observed training instances." We argue further that any basis for changing the background knowledge of the system (other than strict consistency with the input), whether learning by generalization or specialization, whether using inductive, deductive, or analogical processes, amounts to a bias. Such a view allows many additional factors to contribute to the overall bias of a system. This paper concentrates on one such factor: we expand the notion of bias to include input preferences for systems that select their own training data. We call such preference a system's *input bias*.

One such input bias is determined by a preference for failed experiences, rather than successful ones. The detection of a failure in an input stream is then equivalent to self-selection of training examples. Although by filtering examples of successful performance the reasoner may miss some opportunities (and thus bias what can be learned), the input bias trade-off focuses the learner on examples that may require less inference and that guarantee something worth learning exists.[1] If the reasoner has perfect knowledge, no failure can ever occur (oracles by definition do not fail); thus, failure implies a flaw in knowledge, whereas successful examples may or may not contain any useful lessons. For failure to occur, the learning system must be associated with some performance task. In the simplest case, the task may be attribute prediction or classification. For example, when a decision tree misclassifies an instance, ID3 (Quinlan, 1986) uses the instance for learning. In general, a failure-driven approach to learning and reasoning concentrates on contradictions, unexpected successes, surprises, and impasses during the performance task to indicate when attention is warranted (e.g., Birnbaum, Collins, Freed, & Krulwich, 1990; Hammond, 1989; Kolodner, 1987; Newell, 1990; Ram, 1993; Schank, 1982; Sussman, 1975; VanLehn, Jones, & Chi, 1992).

A *failure* is defined as an outcome other than what is expected (or a lack of some outcome). If the system incorrectly analyzes some input, or solves some problem incorrectly, so that its expected solution is different than the actual solution given some criteria or feedback, then a failure has occurred. This is the traditional notion of failure and will be termed a *contradiction*. Moreover, if the system expects that it will not be able to compute any answer or the correct answer, but it does nonetheless, then another failure class exists called an *unexpected success*. Alternatively, if the system has no expectation, yet an event occurs which should have been expected, then a *surprise* exists. Finally, an *impasse* is considered a failure by definition when no solution is generated.

The categories summarized in Table 1 make explicit the bias that underlies the selection of examples based on failure. Contradictions, unexpected successes, surprises, and impasses occur because of the states of the system's background knowledge and the quality of input given to the system. The background knowledge contains not only domain

1. If the reasoner mistakenly believes that there was a failure, but actually there was not, then this itself constitutes a failure from which to learn. The challenge in such a case is to detect the actual failure.

Table 1: Taxonomy of Reasoning-Failure Causes

	Domain Knowledge	**Knowledge Selection**	**Processing Strategy**	**Strategy Selection**	**Goal Generation**	**Goal Selection**	**Input**
Absent	Novel Situation	Missing Association	Missing Behavior	Missing Heuristic	Missing Goal	Forgotten Goal	Missing Input
Wrong	Incorrect Domain Knowledge	Erroneous Association	Flawed Behavior	Flawed Heuristic	Poor Goal	Poor Priority	Noise
Right	Correct Knowledge	Correct Association	Correct Behavior	Correct Choice	Correct Goal	Correct Association	Correct Input

knowledge, but knowledge of processing strategies (e.g., operators, reasoning schemas, or reasoning components[2]) and goal structure. Additionally, if the indexing problem is taken seriously (see Cox, 1994), then selection of domain knowledge, processing strategies and goals can be faulty because of the reasoner's applicability information (hence the three columns concerning selection). In each of these seven categories, the particular piece of information may generate a failure because it is either incorrect or absent.[3] The overall goal is to implement a general system for mapping symptom (failure type such as impasse) to fault (reasoning failure cause such as missing association) to learning strategy (repair of the system's background knowledge by means such as index learning).

We have produced an implementation of a multistrategy learning system called Meta-AQUA (Ram & Cox, 1994) that learns strictly from failure. The system takes advantage of the categories from Table 1 by using case-based methods to associate specific reasoning-failures during its performance task (e.g., contradiction or impasse) with general causal patterns (composed of structures representing cells of Table 1) that explain the failure. Such introspective explanations then form the basis from which the reasoner can generate a set of learning goals that a planner can achieve by selecting various learning strategies from its repertoire of learning methods. Section 2 discusses the bias of failure-driven learning with two examples from Meta-AQUA. Section 3 compares the complexity of failure-driven learning with success-driven learning in terms of the degrees of freedom for interpreting examples. Finally, Section 4 concludes with a short discussion.

2. Collins, Birnbaum, Krulwich, & Freed (1993) use the term "reasoning component" to specify the reasoning knowledge a system uses to perform reasoning tasks. Their theory has concentrated on repairing the components (reasoning strategies), whereas our theory has stressed repair of domain knowledge and the related associations in memory.

3. Note also that in social interactions between agents, failure may occur because of the same causes in the background knowledge of or input to external agents. Our analysis will not, however, consider such possibilities further.

Failure-Driven Input Bias: Two Examples

Before discussing the advantages of using a failure-driven input bias, we will illustrate how Meta-AQUA uses this bias. The performance task in Meta-AQUA consists of using its knowledge of terrorism to understand news stories in a similar domain (drug smuggling). That is, given a stream of concepts representing a story sequence, the task is to create a causally connected conceptual interpretation of the story. If the system fails, its subsequent learning subtasks are (1) *blame assignment* — analyze and explain the cause of its misunderstanding; (2) *decide what to learn* — form a set of learning goals to change its knowledge so that such a misunderstanding is not repeated on similar stories; and then (3) *strategy selection* — choose or construct a learning method by which it achieves these goals (Ram & Cox, 1994).

Meta-AQUA's background knowledge (BK) consists of frame representations based on conceptual dependency theory (Schank, 1975) and explanation pattern (XP) theory (Schank, 1986; Ram, 1991; 1993) augmented with a dynamic memory (Schank, 1982). The BK includes general facts about dogs and sniffing, including the explanation that dogs bark when threatened, but it has no knowledge of police dogs. The BK also contains cases of gun smuggling, but the system has not experienced drug interdiction. The learning task in Meta-AQUA is to learn from failures, incrementally improving its ability to interpret new stories by adjusting the BK. Learning is effective if, given stories that are similar to previously failed ones, it does not repeat its errors.

A Common Contradiction

The following example illustrates a common pattern of failure in systems that are learning new concepts. When a concept is being learned, it may be overly specialized. Slight variation on the concept will cause the system to try to explain it, but without experience with the concept, the system may generate an inappropriate explanation. The proper explanation may not be known because the situation is novel. Thus, a bias toward failure will provide a number of learning opportunities.

Given the short story below, Meta-AQUA attempts to understand each sentence by incorporating it into its current story representation, explaining any anomalous or interesting features of the story, and then learning from any reasoning failures.

S1: A police dog sniffed at an airport passenger's luggage.
S2: The dog suddenly began to bark at the luggage.
S3: The authorities arrested the passenger for smuggling drugs.
S4: The dog barked because it detected 2 kilos of marijuana in the luggage.

Depending on the knowledge of the reader, numerous inferences can be made from this story, many of which may be incorrect. In the story, sentence S1 produces no inferences other than that sniffing is a normal event in the life of a dog. However, S2 produces an anomaly because the system's definition of "bark" specifies that the object of a bark must be animate. So the program (incorrectly) believes that dogs bark only when threatened by animate objects. Since luggage is inanimate, there is a conflict. This anomaly causes Meta-AQUA to ask itself why the dog barked at an inanimate object. It hypothesizes that the luggage somehow threatened the dog. S3 posits an arrest scene that reminds Meta-AQUA of an incident in which weapons were smuggled by terrorists; however, the sentence generates no new inferences concerning the previous anomaly. Finally, S4 causes the original question generated by S2, "Why did the dog bark at the luggage?" to be retrieved. Instead of revealing the anticipated threatening situation, however, S4 offers another hypothesis: "The dog detected drugs in the luggage."

At this point, the system has detected a reasoning failure. Meta-AQUA uses a case-based approach to explain its reasoning failures (i.e., perform blame assignment). Characterizing its error as an expectation failure (a special case of contradiction failure), it retrieves a structure called a meta-explanation pattern (Meta-XP), which is then bound to the trace of reasoning that led to the failed explanation. The Meta-XP has representations for incorrect domain knowledge (faulty conceptual definition), erroneous association (bad index for one explanation), novel situation (lack of knowledge about another explanation), and missing association (missing index for the absent explanation), all of which are cells in Table 1. This structure then aids the system in posting a number of learning goals that, if achieved, will modify the system's BK so that similar errors are not repeated in future episodes (see Cox & Ram, 1994; Ram & Cox, 1994 for further details).

In order to achieve its learning goals, Meta-AQUA uses a least-commitment planner to choose or construct a learning plan. Learning plans consist of sequences of calls to various learning algorithms in its repertoire. The resulting plan is (1) to perform abstraction on the constraint on the object at which dogs often bark; (2) to perform explanation-based generalization (EBG) (DeJong & Mooney, 1986; Mitchell, Keller, & Kedar-Cabelli, 1986) on the newly acquired explanation, producing a generalized pattern that dogs bark at containers when detecting contraband, and then (3) to index the two explanations with respect to each other, so that they each can be retrieved when appropriate. Meta-AQUA instantiates and then executes this plan.

Using a failure-driven input bias allows Meta-AQUA to avoid overhead computation that is less likely to produce useful information. For example, it wastes no computations trying to interpret and learn from the first sentence. It generates explanations during understanding only when anomalies or other interesting concepts are detected; it attempts learning only when reasoning fails. In this example, the reasoning failure is detected when a conflicting explanation is provided by the story. In the next example, the conflict is detected by inference.

A Baffling Situation (Impasse)

The next example demonstrates how Meta-AQUA handles an impasse in which it cannot generate an explanation for an anomaly. The explanation is in its memory, but it does not have the proper index with which to retrieve the explanation. In effect, it has forgotten the explanation. Given a bias for failure, this demonstrates that forgetting represents an interesting opportunity to learn (Cox, 1994) and a novel interpretation of failure (failure of the memory system instead of the inference system).

After processing the previous story, Meta-AQUA's BK contains two explanations for why dogs bark: the memory has an explanation for dogs that bark because they are threatened (indexed by dog-barks-at-animate-object) as well as the explanation for dogs that bark because they detect contraband (indexed by dog-barks-at-container).

Meta-AQUA is then given a second story.

S1: The police officer and his dog enter a suspect's house.
S2: The dog barks at a pile of dirty clothes.
S3: The police officer looks under the clothes.
S4: He confiscates a large bag of marijuana.
S5: The dog is praised for barking at the occluding object.

Although the initial sentence, S1, causes no unusual processing, the second sentence, S2, is interesting to Meta-AQUA because the system has recently changed its concept of `dog-bark`. The system therefore poses a question asking why the dog barked. Unfortunately, because it is barking at neither an animate object nor a container, no XP is retrieved to produce a cause of the event. The question-answering process is subsequently suspended because of the impasse, and the question is indexed in memory. Because it can produce no legitimate explanation to answer its question, Meta-AQUA uses the opportunistic strategy of suspending the question in the hope that the story will provide further information.

Sentence S3 causes the system to postulate a possible causal link between the S2 and S3 simply because of tempo-

ral relation; however, no evidence directly supports it. S4 reminds the system of a case in which contraband was confiscated. The system thus infers that the suspect was most likely arrested. Finally, S5 causes a reminding of the earlier question about the dog barking at the pile of laundry. The reasoning that was associated with this previous question is resumed. The system also infers a causal relation from S5. Although the sentence does not explicitly assert it, Meta-AQUA concludes that the dog's detection of the marijuana caused the dog to bark in the first place. As a result, this conclusion answers the original query.

Reviewing the trace of processing that led up to this conclusion, Meta-AQUA characterizes its condition as being baffled; that is, it could not answer a question either because it did not have an answer or because it could not remember one (i.e., did not have the proper index or association between question context and the abstract explanation pattern that was most appropriate). The system retrieves a Meta-XP based on this characterization, which helps it explain its own reasoning failure. The Meta-XP is a declarative representation of retrieval failure (impasse because of either the missing association or novel situation cells from Table 1). The Meta-XP suggests that a knowledge-expansion goal be spawned to generalize the newly inferred explanation. A knowledge-organization goal is also spawned in order to index the generalized explanation in memory. These goals can be achieved by performing EBG on the new explanation and then indexing it by the context in which the system encountered the explanation.

The system is not able to determine *a priori* whether an explanation actually existed in memory which it could not recall (thus the cause of the failure was actually a missing association), or whether it lacks the knowledge to produce the explanation (or the cause could be novel situation). It thus poses a question about its own Meta-XP, "Does such a structure, M, exist in memory?" The answer is obtained by performing EBG and watching for a similar explanation in memory when it stores the new explanation via the indexing algorithm. It thus finds the explanation produced by the previous story at storage time. Generalizing the two, it produces a better explanation: dogs bark at objects that hide contraband, not simply at containers. So that these types of explanations will not be forgotten again, it indexes the explanation by potential hiding places.

Meta-AQUA uses its knowledge of failure to direct its effort toward situations most likely to benefit inference and learning. The following section shows that such a strategy is useful because the possible explanations for failure are more constrained than those in successful performance would be.

The Degrees of Freedom in Learning

Many systems invest too much computational overhead in evaluating examples that, in ordinary performance situations, provide few or no useful opportunities to learn. For example, PRODIGY (Minton, 1990) has no input bias. As a consequence, it learns from every input and then must delete useless knowledge. Alternatively, desJardins' (1992) PAGODA uses a given input's expected utility of predicting features in the environment to filter input examples. Like Meta-AQUA, the system uses a goal-directed learning approach to formulate a set of learning goals that direct and guide the system's learning. However, Meta-AQUA uses an explanation of the system's own failure to generate these goals, while PAGODA uses the expected utility of the input. But it is more tractable to let failure feedback from the environment filter the input for useful candidates for goal formulation, rather than calculating the utility of all instances, because fewer instances exist on which to perform computation. Moreover, learning will be simpler in the remaining examples because, as will be shown below, fewer degrees of freedom generally exist for blame assignment when learning from failure than for credit assignment when learning from success.

To illustrate the utility of failure-driven bias, consider the following. During the Persian Gulf oil embargo of Iran, a tragic event occurred that resulted in the death of innocent civilians.[4] The *USS Vincennes* shot down an Iranian commercial airliner after an engagement with Iranian gunboats on July 3, 1988. On the basis of conflicting information, the captain of the *Vincennes* mistook the airliner for an enemy F-14 fighter aircraft and ordered it shot down. Although the incident was controversial, an official investigation concluded that the captain acted in a proper manner given the rules of engagement and the circumstances under which the captain made such a decision.

Instead of this incident being simply a negative example of the category F-14, let us propose three classifiers: one represents the concept "friendly target," another recognizes "neutral targets," and a third classifies "enemy targets." Let us also assume for the sake of simplicity that the friendly-target concept returns negative because of no electronic signature. The remaining two concepts return a value and a confidence level. The captain's quandary stems from the low confidence returned by the positive identification from the enemy classifier, along with an equally low confidence for the negative classification of neutral aircraft. Given no noise in the data and that an unambiguous result occurs (no possibility of both true or both negative), Table 2 summarizes the possible explanations for answers to the question "Is the reported plane neutral?"

In both failed cases (the shaded cells: false positive and miss), there is only one possibility. If, as in the actual incident, there is a miss (i.e., the actual answer is positive but the expected outcome is negative), then the concept of neutral plane must be overly specialized, since it rejects a positive example; whereas the concept of enemy plane must be overly general since it accepts a negative example. Blame assignment for the converse case, that of a false positive, is equally unambiguous. If the concept of neutral plane mistak-

4. Details of this incident are taken from Thagard (1992).

enly recognizes an example of an enemy target, then it must be overly general; and at the same time, if the classifier of enemy planes rejects the same example, then the concept must be overly specialized.

Table 1: Is the plane neutral? Possible causes[a]

Expected Outcome \ Actual Outcome	Neg	Pos
Pos	**False Positive** $\overline{\mathbf{N}} \wedge \underline{\mathbf{E}}$ **(crew dies)**	Hit $\overline{\mathrm{N}} \wedge \underline{\mathrm{E}}$ $\mathrm{N} \wedge \mathrm{E}$ $\overline{\mathrm{N}} \wedge \mathrm{E}$ $\mathrm{N} \wedge \underline{\mathrm{E}}$
Neg	Correct Rejection $\underline{\mathrm{N}} \wedge \overline{\mathrm{E}}$ $\mathrm{N} \wedge \mathrm{E}$ $\underline{\mathrm{N}} \wedge \mathrm{E}$ $\mathrm{N} \wedge \overline{\mathrm{E}}$	**Miss** $\underline{\mathbf{N}} \wedge \overline{\mathbf{E}}$ **(innocents die)**

a. N= Neutral target; E = Enemy target; Overscore = overly general; Underscore = overly specialized; Light shading = failed prediction.

In successful examples of performance, many more degrees of freedom exist with which to do credit assignment. In positive identifications of neutral aircraft, in which the neutral classifier returns true and the enemy classifier returns false, both classifiers may still be incorrect in general. That is, although a neutral classifier may be overly general, it can still return true on all positive examples. Likewise, the enemy classifier can be overly specialized and still reject all negative examples. Simply because a particular target is correctly identified, we do not have much information as to the classifier's overall performance. In the case of correct rejection, an overly specialized neutral classifier may still reject a particular enemy aircraft, and even though the enemy classifier may properly accept a particular enemy example, it may still be overly general and succeed. Success gives little information as opposed to failure.

Failure-driven input bias is limited, however. Although failure may constrain learning, some systems may not be able to use this fact because a particular inductive policy (the strategy used to make bias choices based on the underlying assumptions of the domain) may influence a learning system toward certain results (Provost & Buchanan, 1992). Provost and Buchanan show that inductive policies can bias a learner toward speed of acquisition rather than accuracy (when time is a limited resource, for example) or toward accuracy instead of speed (when safety is a high priority). Likewise, in the *Vincennes* scenario, even though failure may facilitate learning, life-critical tasks require that the performance system not choose a course that results in failed examples. The approach of LEX (Mitchell, Utgoff, & Banerji, 1983), which generates learning examples on the basis of their expected utility, irrespective of any inductive policy, is unacceptable. The crew of the *Vincennes* strove for hits and correct rejections despite the fact that much could be learned from examples like the unfortunate incident (miss) that did occur. The consequences of both false positives and misses require an inductive policy that biases the performance system toward accuracy and away from learning optimization.

Conclusions

Markovitch and Scott (1993) characterize learning systems in terms of filters placed in an information flow through a system. This paper investigated bias at the front end in the information flow. Markovitch and Scott call such a filter *selective experience*. They divide selective experience into three types: error-based, uncertainty-based, and miscellaneous heuristics. The examples presented in this paper are error-based, although the scope of the selective-experience filter in Meta-AQUA goes beyond their formulation because, as explained in the introduction, error has numerous variations, only one of which (contradiction) Markovitch and Scott consider. Moreover, they claim that error-based filters are useful only when the input is in the form of problem/solution tuples. During its impasse (example #2), however, Meta-AQUA generated no solution, yet as shown in the example section, the system still learned a valuable lesson from the experience.

Meta-AQUA filters input examples in a relatively passive manner. It waits for failures to occur, then processes them by explaining the failure, deciding what to learn, and selecting a learning strategy. Another issue to pursue would be to have the system try to actively generate failed experiences in order to test or disprove some hypothesis or to generate learning experiences for some performance task. As the previous section asserts, however, the system must be sensitive to inductive policies concerning the task domain. Currently, the ability of a system to actively challenge itself and its knowledge is beyond the scope of our research.

We are currently extending the Meta-XP representations to address the problems of missing input and noise. Meta-AQUA should reason about the input to decide, for example, whether a particular failure is caused by noise (e.g., faulty data collection device), or whether the same error stems from novel situations (e.g., inexperience with the domain). Some of the categories in our failure-cause taxonomy are included in related research. For example, Owens (1991) identified a number of similar categories. One of the failures that he addresses is error due to time constraints. Integrating this kind of failure into our framework is an open question. However, Owens' taxonomy is specific to planning failures, whereas our taxonomy is independent of both the performance task and domain. For example, by adding a few conceptual definition for a new domain, Meta-AQUA processes and learns from the following story that parallels the first example of Section two.

S1: A person enters the handball court.
S2: The person suddenly hit a handball.
S3: He hit the ball because he wanted to have fun.

As before, S1 is skimmed and, because Meta-AQUA believes that people hit animate objects, S2 generates an anomaly. It explains the anomaly by concluding that the person is trying to hurt the ball. When given a new explanation, Meta-AQUA generalizes it, indexes the new explanation with respect to the hurt explanation, and loosens the constraint on the object of `hit` to include toys as well as animate objects. Meta-AQUA uses the same Meta-XP as a pattern of failure and guide to learning as in the previous story.

One of the goals of this paper is to expand the scope of learning bias. Failure-driven filtering of input examples provides a valuable bias because the degrees of freedom when explaining examples of failure are less than when explaining success. Meta-AQUA uses explicit knowledge of reasoning failures to perform blame assignment and to generate a set of learning goals which guide its learning. The methods outlined here provide a first step toward modeling a general ability to use failure to guide both blame assignment and the selection of quality examples.

Acknowledgments

This research was supported by AFOSR under grant C-36-X26 and by the Georgia Institute of Technology. We thank Foster Provost and Kenny Moorman for comments on an earlier draft of this paper, Susan Farrell for an excellent proof, and Janis Roberts for suggesting the handball example. The anonymous reviewers also provided quality criticism and feedback.

References

Birnbaum, L., Collins, G., Freed, M., & Krulwich, B. (1990). Model-based diagnosis of planning failures. In *Proc. of the 8th Nat. Conf. of Artificial Intelligence* (pp. 318-323).

Collins, G., Birnbaum, L., Krulwich, B., & Freed, M. (1993). The role of self-models in learning to plan (pp. 83-116). In *Foundations of knowledge acquisition: Machine learning*. Boston: Kluwer. (Also available as Tech. Rep. #24, Institute of the Learning Sciences, Northwestern University, Evanston, IL, April, 1992).

Cox, M. T. (1994). Machines that forget: Learning from retrieval failure of mis-indexed explanations. In *Proc. of the 16th Annual Conf. of the Cognitive Science Society.* Hillsdale, NJ: LEA.

Cox, M. T., & Ram, A. (1994). Choosing Learning Strategies to Achieve Learning Goals (pp. 12-21). In M. desJardins & A. Ram (Eds.), *Proc. of the 1994 AAAI Spring Symposium on Goal-Driven Learning*, Menlo Park, CA: AAAI Press.

DeJong, G., & Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning, 1*(2), 145-176.

desJardins, M. (1992). Goal-directed learning: A decision-theoretic model for deciding what to learn next. In *Proc. of the ML-92 Workshop on Machine Discovery* (pp. 147-151).

Hammond, K. J. (1989). *Case-based planning: Viewing planning as a memory task*, vol. 1 of *Perspectives in artificial intelligence*. San Diego: Academic Press.

Kolodner, J. L. (1987). Capitalizing on failure through case-based inference. In *Proc. of the 9th Annual Conf. of the Cog. Science Society* (pp. 715-726). Hillsdale, NJ: LEA.

Markovitch, S., & Scott, P. D. (1993). Information filtering: Selection mechanisms in learning systems. *Machine Learning, 10* (2), 113-151.

Minton, S. (1990). Quantitative results concerning the utility of explanation-based learning. *Art. Intel., 42,* 363-392.

Mitchell, T. M. (1990). The need for biases in learning generalizations. In J.W. Shavlik and T.G. Dietterich (Eds.), *Readings in machine learning* (pp. 184-191). San Mateo, CA: Morgan Kaufmann. (Originally published in 1980.)

Mitchell, T. M., Keller, R., & Kedar-Cabelli, S. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1*(1), pp. 47-80.

Mitchell, T. M., Utgoff, P., & Banerji, R. (1983) Learning by experimentation: Acquiring and refining problem-solving heuristics. In R. Michalski, J. Carbonell, & T. Mitchell (Eds.), *Machine leaning: An artificial intelligence approach* (pp. 163-190). San Mateo, CA: M. Kaufmann.

Newell, A. (1990) *Unified theories of cognition*. Cambridge, MA: Harvard Univ. Press.

Owens, C. (1991). A Functional Taxonomy of Abstract Plan Failures, In *Proc. of the 13th Annual Conf. of the Cognitive Science Society.* Hillsdale, NJ: LEA.

Provost, F. J., & Buchanan, B. G. (1992). Inductive policy. In *Proc. of the 10th Nat. Conf. on Artificial Intelligence* (pp. 255-261). Menlo Park, CA: AAAI Press.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*, 81-106.

Ram, A. (1991). A theory of questions and question asking. *J. of the Learning Sciences, 1*(3&4), 273-318.

Ram, A. (1993). Indexing, elaboration and refinement: Incremental learning of explanatory cases. *Machine Learning, 10*(3), 201-248.

Ram, A., & Cox, M. T. (1994). Introspective reasoning using meta-explanations for multistrategy learning. In R. Michalski & G. Tecuci (Eds.), *Machine learning: A multistrategy approach IV* (pp. 349-377). San Mateo,CA:M. Kaufmann.

Schank, R. C. (1975). *Conceptual information processing*. Amsterdam: North-Holland.

Schank, R. C. (1982). *Dynamic memory: A theory of reminding and learning in computers and people*. Cambridge, MA: Cambridge Univ. Press.

Schank, R. C. (1986). *Explanation patterns: Understanding mechanically and creatively.* Hillsdale, NJ: LEA

Sussman, G. J. (1975). *A computer model of skill acquisition*. New York: American Elsevier.

Thagard, P. (1992). Adversarial problem solving: Modeling an opponent using explanatory coherence. *Cognitive Science, 16* (1), 123-149.

VanLehn, K., Jones, R. M., and Chi, M. T. H. (1992). A model of the self-explanation effect. *J. of the Learning Sciences, 2* (1), 1-60.