Beyond incremental processing: Tracking concept drift

Jeffrey C. Schlimmer and Richard H. Granger, Jr.

Department of Information and Computer Science
University of California, Irvine 92717
ArpaNet: Schlimmer@ICS.UCI.EDU, Granger@ICS.UCI.EDU

Abstract

Learning in complex, changing environments requires methods that are able to tolerate *noise* (less than perfect feedback) and *drift* (concepts that change over time). These two aspects of complex environments interact with each other: when some particular learned predictor fails to correctly predict the expected outcome (or when the outcome occurs without having been preceded by the learned predictor), a learner must be able to determine whether the situation is an instance of noise or an indication that the concept is beginning to drift. We present a learning method that is able to learn complex Boolean characterizations while tolerating noise and drift. An analysis of the algorithm illustrates why it has these desirable behaviors, and empirical results from an implementation (called STAGGER) are presented to show its ability to track changing concepts over time.

I Introduction

Sometimes a low barometer reading indicates rain coming, and sometimes it doesn't. Furthermore, for months after a volcanic eruption, previously good indicators of rain may become poor predictors, while other (previously poor) indicators may become predictive. Attempting to learn from experience about associations between events like these in the real world is confounded because (a) most associations are not perfectly consistent (hence observed instances of these associations contain 'noise'), and (b) associations change or *drift* over time. Learning in these environments is compounded by the fact that noise and drift interact: if at some point a particular good indicator fails to predict the intended outcome, is this just a noisy instance, or is it an indication that the concept is beginning to drift?

Nature has solved this problem in humans and animals: rats in classical conditioning experiments are able to tolerate noise and drift, even in extremely complex environments with many competing cues. However, few current machine learning systems are able to tolerate noise and drift, and hence cannot deal with complex reactive environments containing these qualities. We present a learning method that tolerates noise and drift, and we offer an analytical account of why it behaves as well as it does. The method is able to keep track of, and hence distinguish between, different types of noisy instances. Via formula based on Bayesian statistics, it tolerates *systematic* noise, but not random noise, distinguishes between noise and drift, and is able to track changing concepts over time. We have implemented this method in a computer program called STAGGER, and have tested it in a variety of environments, ranging from animal learning tasks to blocksworlds to chess endgames. We present some empirical findings reflecting the program's ability to track drifting concepts.

II Related work

Many successful learning systems have failed to deal with the issue of concept drift over time. Quinlan's ID3 (1986) program, for example, constructs a discrimination tree to characterize instances of a concept. This representation allows conjunctive, disjunctive, and negated characterizations. Quinlan has examined the ability of this method to accommodate varying levels of noise, concluding that its performance is close to optimal (Quinlan, 1986). However, the method is nonincremental, for it requires examining (and re-examining) a relative large number of instances and does not have mechanisms for modifying an existing tree to incorporate new instances. It is unable, therefore, to track changes in concept definitions over time.

The incremental nature of a learning algorithm does not guarantee that it will be able to deal with concept drift over time. Mitchell (1982), for example, reports on the version space learning method in which an appropriate description of observed instances is formed via a bidirectional search through a space of possibilities. Though relational information is utilized, the version space method assumes the strong bias that a conjunctive characterization can accurately capture the concept to be learned. In later work

(Mitchell, Utgoff, & Banerji, 1983), a modification was proposed which would form disjunctive descriptions or tolerate limited noise in instances (but not both, interestingly). Though this method is incremental, learned characterizations may not change and recross the search boundaries previously established in the version space as the definition of a concept drifts over time.

Langley's discrimination learning method (in press) is able to track changes in a concept definition over time. The learned concepts are expressed as a set of production rules, one of which influences expectation at a time. If the applicability conditions for an operator change, presumably recently learned productions would be weakened via strengthening while discrimination would propose new ones. Eventually, the new characterizations would be strengthened and overwhelm any previous learning. Because this method is based on a strengthening evaluation function, however, it does not distinguish between *types* of noise.

III A new learning method: STAGGER

The heart of STAGGER's learning method is based on a distributed concept representation composed of a set of dually weighted, symbolic characterizations. As each new instance is processed, a cumulative expectation of its identity is formed by using the pair of weights associated with characterizations. Learning occurs at two levels: adjustment of the weights and generation of new Boolean characterizations. This latter process constructs more general, more specific, and inverted versions of existing concept description elements. These new characterizations compete for inclusion in the concept description with the elements that were combined to form them.

A. Concept representation and matching

Concepts are represented in STAGGER as a set of dually weighted, symbolic characterizations. Each element of the concept description is a Boolean function of attribute-value pairs represented by a disjunct of conjuncts. An example element matching either small blue figures or square ones would would be represented as (`size = small` *and* `color = blue`) *or* `shape = square`. These characterizations are dually weighted in order to capture positive and negative implication. One weight represents the sufficiency of a characterization for prediction, or $(matched \Rightarrow pos)$, and the other represents its necessity, or $(\neg matched \Rightarrow \neg pos)$.

The mathematical measures chosen for the sufficiency and necessity weights are based on psychological learning results. In a classical conditioning experiment, a subject is given repeated presentations of a novel cue (NC) and an unpleasant stimulus (US). After extensive testing, Rescorla (1968) formulated the contingency law which states that subjects will learn an association between the two events only if the unpleasant stimulus is more likely following the novel cue that without it, or $p(US|NC) > p(US|\neg NC)$. In behavioral terms, this means that if one or the other stimulus frequently occurs alone, the subject still learns an association between the two cues. However, if each of the stimuli occur alone even a few number of times, learning about their association is severely impaired.

Real-world tasks also contain spurious events. For example, the descriptions of instances be subject to either *random* or *systematic* variation. An example of random noise would be a temperature sensor which is accurate to within 10% of its operating range. It may read too high on one occasion and too low on another; the *direction* of its error is random. Only a few authors have dealt with this possibility (e.g., Quinlan, 1986). However, it may often be the case that errors in description are the result of a systematic variation. For example, a rain gauge may leak and sometimes read lower, but never higher, than it should. The errors of this latter instrument are *systematically* of one type (only too low), though they may occur with an unpredictable frequency. The contingency law states that learning occurs in systematic cases but is dubious in situations with random variation.

With this in mind, STAGGER uses logical sufficiency (LS), or positive likelihood ratio, as a measure of sufficiency (Duda, Gaschnig, & Hart, 1979). Similarly, logical necessity (LN), or negative likelihood ratio, serves to measure necessity. They are defined as:

$$LS \quad \frac{p(matched|pos)}{p(matched|neg)} \qquad LN \quad \frac{p(\neg matched|pos)}{p(\neg matched|neg)}$$

LS ranges from zero to positive infinity and is interpreted in terms of odds. (Odds may be easily converted to probability $p = odds/(1 + odds)$.) An LS value less than unity indicates a negative correlation, unity indicates independence, and a value greater than unity indicates a positive relationship. LN also represents odds and takes on values from zero to positive infinity. However, an LN value near zero indicates a positive correlation, and a value greater than unity indicates negative correlation. For both LS and LN, unity indicates irrelevance. The LS and LN measures adhere to the contingency law, for it can be shown via algebraic manipulations that $LS > 1$ and $LN < 1$ if and only if $p(US|NC) > p(US|\neg NC)$ (Schlimmer, 1986).

Given a list of attribute-value pairs describing an instance, the distributed concept representation as a whole influences expectation of a positive or negative instance. Following the mechanism used by Duda, Gaschnig, and Hart (1979), the dual weights associated with each characterization are used together with estimated prior odds to

calculate the odds that a given instance is positive. Expectation is the product of the prior odds of a positive instance and the LS values of all matched characterizations and the LN values of all unmatched ones.

$$Odds(pos|inst) = Odds(pos) \times \prod_{\forall matched} LS \times \prod_{\forall unmatched} LN$$

The resulting number represents the odds in favor of a positive instance. This holistic approach differs from most machine learning systems in which a single characterization completely influences concept prediction.

B. Learning mechanisms

In addition to representing concepts in a distributed manner and using Bayesian measures to compute a holistic expectation, STAGGER incrementally modifies both the weights associated with individual characterizations and the structure of the characterizations themselves. These two latter abilities allow STAGGER to adapt its concept description to better reflect the concept.

The *sufficiency* and *necessity* weights associated with each of the concept description elements may be easily adjusted. Consider the possible situations that may arise when matching a characterization against an instance. Following the terminology used by Bruner, Goodnow, and Austin (1956), a positive instance is positive evidence which may either *confirm* the predictiveness of a characterization (if it is matched in this instance) or *infirm* the characterization's predictiveness (if it is unmatched). Similarly, a negative instance is negative evidence which either confirms an unmatched element or infirms a matched one. Table 1 summarizes these possibilities.

Table 1: Possible situations in matching a characterization to an instance.

Instance	Characterization	
	Matched	Unmatched
Positive	Confirming (C_P)	Infirming (I_P)
Negative	Infirming (I_N)	Confirming (C_N)

In terms of these matching events, the contingency law implies that learning occurs in cases involving at most one type of infirming evidence. In situations with even small amounts of both positive and negative infirming evidence, subjects fail to learn an association. The corresponding definition of systematic variation is the presence of only one type of infirming evidence while random variation is defined as both types of infirming evidence.

The weighting measures LS and LN may be easily calculated by keeping counts for each characterization of the possible situations listed in Table 1.

$$LS = \frac{C_P(I_N + C_N)}{I_N(C_P + I_P)} \qquad LN = \frac{I_P(I_N + C_N)}{C_N(C_P + I_P)}$$

The prior odds for a positive instance are easily estimated as $(C_P + I_P)/(I_N + C_N)$.

If STAGGER limited its learning to adjustment of the characterization weights, the distributed concept representation would be sufficient to accurately describe the class of "linearly separable" concepts (Hampson & Kibler, 1983). In this respect STAGGER is similar to *connectionist* models of learning when those models do not have any "hidden" units. The purpose of the hidden, internal units is to allow the encoding of more complicated concepts. Search processes in STAGGER serve an analogous purpose: individual characterizations are combined into more complex Boolean functions.

STAGGER searches through a space of possible characterizations as it refines its initial distributed representation of the concept into a unified, accurate one. Each possible Boolean characterization of attribute-value pairs may be viewed as a node in the space of all such functions. Figure 1 depicts a small portion of this space over a simple domain (each ellipse represents a Boolean function). Any two of the possible Boolean functions are partially ordered along a dimension of generality (Mitchell, 1982).

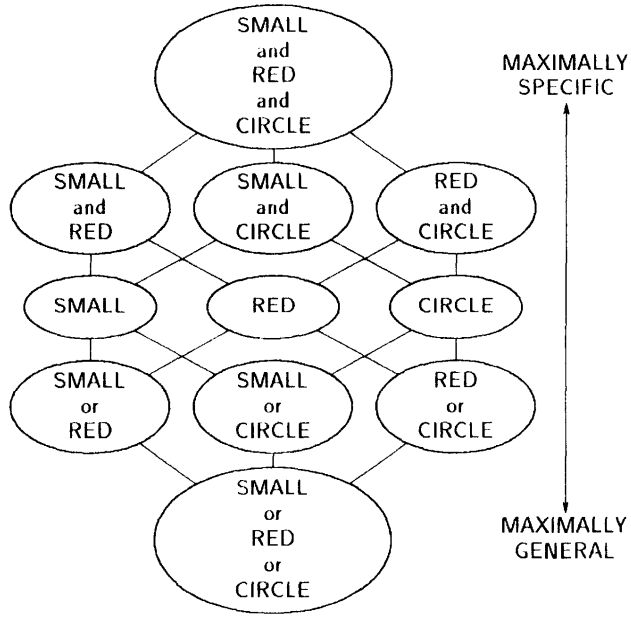

Figure 1: Partial characterization search space.

STAGGER's initial concept description consists of the simple characterizations in the middle of Figure 1 each with initially unbiased weights. Notice that this space is more than twice the size of that typically searched by a conjunction-only method like version spaces (Mitchell, 1982). Another interesting difference is that the version space method searches its space of characterizations from both sides toward the middle; STAGGER beam-searches from the simplest points in the middle outward toward both boundaries.

STAGGER's three search operators correspond to specializing, generalizing, or inverting characterizations. To make a concept description element more specific, search proceeds down a conjunctive path. Conversely, to make a more general element, search proceeds to a new disjunction. Lastly, a poorly scoring characterization may be negated; this does not raise or lower its degree of generality.

The conjunction, disjunction, and negation operators are not applied exhaustively; search is limited by proposing new elements only when STAGGER makes an expectation error. When a negative instance is predicted to be positive (an error of commission), the expectation is too general. Thus search is expanded toward a more specific characterization. On the other hand, a guess that a positive instance is negative (an error of omission) is overly specific; search is expanded to include a more general characterization. Either type of error also causes STAGGER to expand search by proposing the negation of a poor characterization. Table 2 summarizes the operators' preconditions.

Table 2: Search operator preconditions.

Error type	Search operator	
Commission	Specialize	AND[c1,c2]
Omission	Generalize	OR[c1,c2]
Either	Invert	NOT[c]

STAGGER follows a two-step process of choosing good arguments for the operators; one set of heuristics *nominates* potential arguments, and a second set *elects* the most predictive ones for inclusion in new characterizations.

The nomination heuristic specifies alternative groups of characterizations from which to form compounds. After STAGGER has made an error of commission, characterizations matched in this negative instance may be partially necessary, but are clearly not sufficient. Some elements must have suggested (via the matching process) that this instance was likely to be positive, but because this instance was negative, some necessary element was unmatched. Conjunction combines two necessary elements, so matched characterizations are nominated along with unmatched ones. If a disjunction is formed, elements which are unmatched in this nonexample are nominated since disjunction combines two sufficient characterizations, and no sufficient characterizations were present. Negation is used to invert characterizations which predict nonexamples. Its component is nominated from those characterizations which are matched in this nonexample. Similar heuristics apply for an error of omission. Table 3 summarizes STAGGER's nomination heuristics.

The election heuristic uses the weighting measures to narrow the possible operator arguments. Consider a situation leading STAGGER to appropriately propose a new conjunction. For example, the familiar concept *father*: a

Table 3: Nomination heuristic.

Error type	Function	Characterization nomination
Commission	AND[c1,c2]	Matched, Unmatched
	OR[c1,c2]	Unmatched, Unmatched
	NOT[c]	Matched
Omission	AND[c1,c2]	Matched, Matched
	OR[c1,c2]	Matched, Unmatched
	NOT[c]	Unmatched

parent and a male. The two characterizations (parent and male) are always matched in a positive instance (father) though they sometimes occur alone (a brother is male). This is negative infirming evidence (refer to Table 1). LN tolerates negative infirming evidence, and therefore elects criterial elements for conjunctions. By similar reasoning, the converse weighting measure, LS, elects high scoring characterizations to be used in forming new, disjunctive characterizations. New negated characterizations are elected equally by both measures. Table 4 summarizes these second step candidate election heuristics.

Table 4: Election heuristic.

Function	Election measure
AND[c1,c2]	$LN(ci) \ll 1$
OR[c1,c2]	$LS(ci) \gg 1$
NOT[c]	$LN(c) \gg 1$ or $LS(c) \ll 1$

New characterizations are introduced into the search frontier in a generate-and-test manner. The search operators generate new characterizations which are then either pruned from the frontier or established as part of it. To avoid being pruned, a new characterization must be more effective than its sponsoring components. If the new element surpasses a weight threshold, it is established and its components are pruned. Interim performance is assessed by examining recent changes in its weights. These changes are averaged, and if this average is very small, the element appears to be reaching an asymptote. If it is still below threshold, the characterization is pruned.

STAGGER will trigger backtracking when the weighting measures indicate that the new characterization is performing worse than when it was established. Its pruned components are reactivated and compete as the failing element did before. This amounts to chronological backtracking because moves through the search space are retracted in the opposite order from which they were proposed.

IV Tracking concept drift

An important feature of a learning mechanism is its responsiveness to changes in the environment. For instance, a fox learns to look for a changed coat color in his prey

as the seasons change. First, the learner must distinguish between randomness and genuine change. For a failed expectation, the question arises as to whether it was simply a noisy instance, and should be tolerated, or whether it indicates that the learned concept has drifted. STAGGER uses the Bayesian weighting measures to distinguish between events that indicate a change in the definition of a concept and those which are probably the result of noise. Secondly, does the amount of previous learning about a given concept definition affect subsequent relearning of a new definition? In humans and animals it does. The adage "It's hard to teach an old dog new tricks" roughly captures a main finding in learning (e.g., Siegel & Domjan, 1971). These studies indicate that the resiliency of learned concept definitions is inversely proportional to the amount of training; briefly trained concepts are more readily abandoned in the face of change than extensively trained ones.[1] Keeping counts of the evidence types in table 1 amounts to retaining a history of association, allowing STAGGER to model resiliency appropriately.

A. Empirical performance

Figure 2 depicts the performance of STAGGER on three successive definitions for the same concept: (1) `color = red` *and* `shape = squarish`, (2) `size = small` *or* `shape = circular`, (3) `color = (blue or green)`. The dashed vertical lines indicate when the definition of the concept was changed. Notice how performance falls immediately following the change because the previously acquired definition was not sufficient to characterize new, changed instances. In each of the three cases STAGGER formed the explicit, symbolic representation of the concept's definition and evaluated it as the best among those on the search frontier.

STAGGER addresses the noise versus change issue through the use of its weighting measures. When LS and LN indicate a change in the type of noise present, they trigger backtracking as explained above. On the other hand, more of the same type of noise does not lead to the modification of characterizations. Figure 3 depicts STAGGER's acquisition of the `color = red` *or* `size = squarish` characterization as in figure 2. After the dashed vertical line, positive instances were subjected to 25% negative infirming, systematic noise. That is, 25% of the positive instances were randomly assigned to either the positive or negative class; a situation similar to the leaky rain gauge. Notice that unlike figure 2, performance is not adversely affected, indicating that STAGGER is correctly distinguishing between noise and concept change.

Because STAGGER retains counts of situation types, it is

[1] In STAGGER, positive transfer arises as previous learning is used to augment the concept representation language (Schlimmer & Granger, in press).

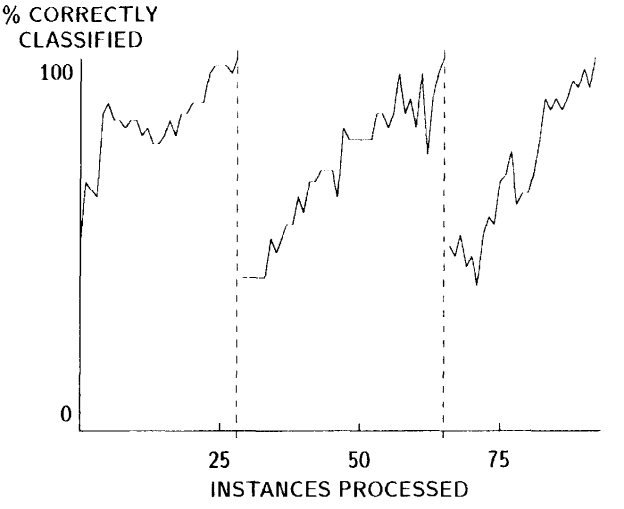

Figure 2: Tracking concept drift.

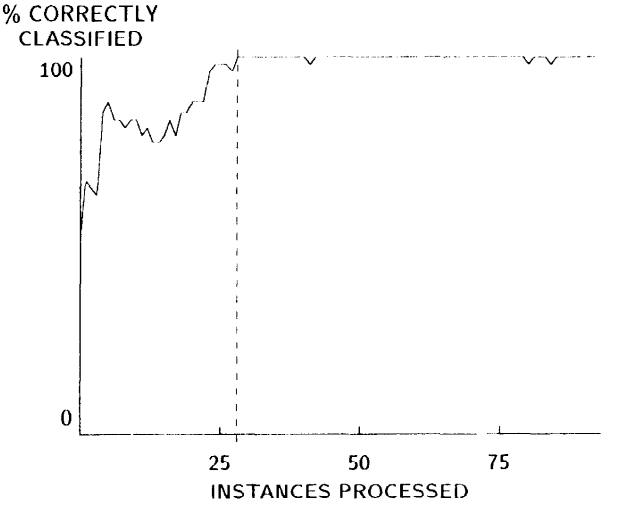

Figure 3: 25% systematic noise.

in effect keeping an abbreviated history of the correlation between a characterization and a concept definition. This allows the program to model the effects of varying amounts of previous learning on relearning resiliency at a gross level. Contrast figure 4 in which the program was given more than four times the amount of training for each concept before each change than in figure 2. Notice that the recovery learning is considerably faster (higher resiliency) in the minimal training case (figure 2).

In short, the heuristic demonstrated here is that briefly trained concepts are less likely to be stable and should therefore be abandoned more quickly in the face of change. On the other hand, extensively trained concepts are more stable and have a longer history of past success; they should be less resilient in the face of new evidence. Psychological studies indicate that natural learning mechanisms behave in this manner (Siegel & Domjan, 1971).

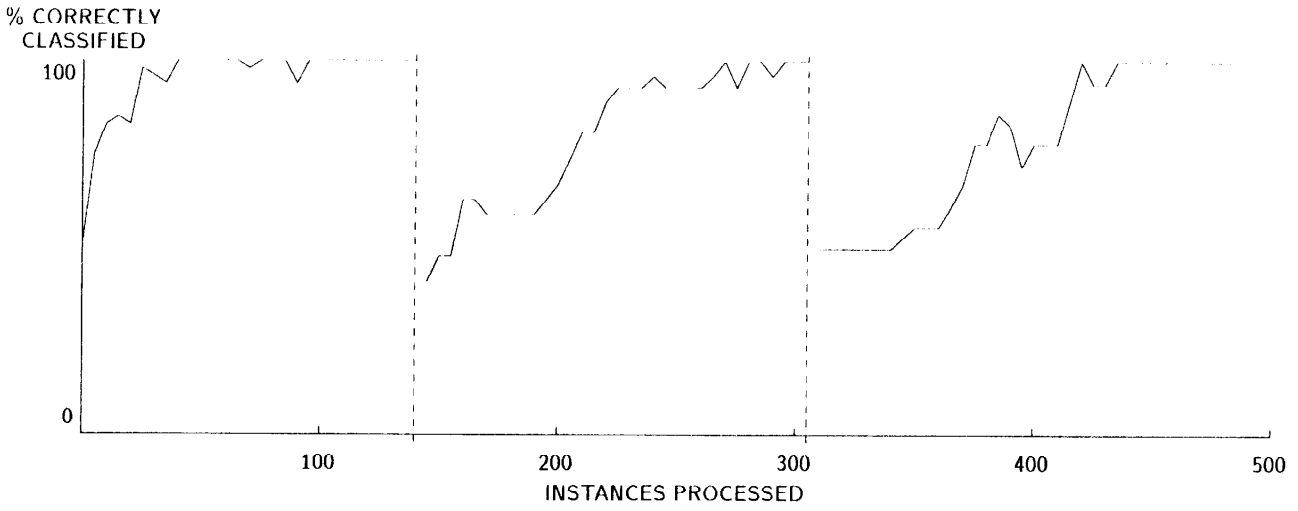

Figure 4: Tracking concept drift given overtraining.

V Conclusions

STAGGER is an incremental learning method which tolerates systematic noise and concept drift. It begins with simple characterizations and learns complex characterizations by conducting a middle-out beam search through the space of possible conjunctive, disjunctive, and negated characterizations. Backtracking allows tracking changes in concept definitions over time. Furthermore, the use of the Bayesian weighting measures affords the proper distinction between noise and genuine concept drift. By retaining numerical histories of events, STAGGER models the effects of overtraining seen in psychological experiments. The learning methods employed in STAGGER are far from a complete solution to the problems of learning in complex, reactive environments. So far, it is limited to learning Boolean combinations of attribute values and cannot acquire relational descriptions of structured objects. STAGGER also requires feedback, as all concept attainment systems do, and is therefore unable to conceptually cluster its inputs.

Acknowledgements

This research was supported in part by the Office of Naval Research under grants N00014-84-K-0391 and N00014-85-K-0854, the National Science Foundation under grants IST-81-20685 and IST-85-12419, the Army Research Institute under grant MDA903-85-C-0324, and by the Naval Ocean Systems Center under contract N66001-83-C-0255. We would like to thank Michal Young who was involved in the early formulation of these ideas, Ross Quinlan for suggesting a natural extension to the matching process, and the entire machine learning group at Irvine for their vigorous discussions and consistent encouragement.

References

Bruner, J. S., Goodnow, J. J., & Austin, G. A. (1956). *A study of thinking*. New York: John Wiley & Sons, Inc.

Duda, R., Gaschnig, J., & Hart, P. (1979). Model design in the Prospector consultant system for mineral exploration. In D. Michie (Ed.), *Expert systems in the micro electronic age*, Edinburgh: Edinburgh University Press.

Hampson, S., & Kibler, D. (1983). A Boolean complete neural model of adaptive behavior. *Biological Cybernetics, 49*, 9-19.

Langley, P. (in press). A general theory of discrimination learning. In D. Klahr, P. Langley, & R. Neches (Eds.), *Production system models of learning and development*.

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence, 18*, 203-226.

Mitchell, T. M., Utgoff, P. E., & Banerji, R. (1983). Learning by experimentation: Acquiring and refining problem-solving heuristics. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning: An artificial intelligence approach*. Palo Alto, California: Tioga Publishing Company.

Quinlan, J. R. (1986). The effect of noise on concept learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach, volume II*. Los Altos, California: Morgan Kaufmann Publishers, Inc.

Rescorla, R. A. (1968). Probability of shock in the presence and absence of CS in fear conditioning. *Journal of Comparative and Physiological Psychology, 66*, 1-5.

Schlimmer, J. C. (1986). *A note on correlational measures* (Technical report #86-13). Irvine, California: The University of California, Department of Information and Computer Science.

Schlimmer, J. C., & Granger, R. H., Jr. (in press). Incremental learning from noisy data. *Machine Learning*.

Siegel, S., & Domjan, M. (1971). Backward conditioning as an inhibitory procedure. *Learning and Motivation, 2*, 1-11.