Inductive Policy

Foster John Provost and **Bruce G. Buchanan**
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
foster@cs.pitt.edu

Abstract

The concept of *inductive bias* can be broken down into the underlying *assumptions* of the domain, the particular *implementation* choices that restrict or order the space of hypotheses considered by the learning program (the *bias choices*), and the *inductive policy* that links the two. We define inductive policy as the strategy used to make bias choices based on the underlying assumptions. Inductive policy decisions involve addressing tradeoffs with respect to different bias choices. Without addressing these tradeoffs, bias choices will be made arbitrarily. From the standpoint of inductive policy, we discuss two issues not addressed much in the machine learning literature. First we discuss batch learning with a strict time constraint, and present an initial study with respect to trading off predictive accuracy for speed of learning. Next we discuss the issue of learning in a domain where different types of errors have different associated costs (risks). We show that by using different inductive policies accuracy can be traded off for safety. We also show how the value for the latter tradeoff can be represented explicitly in a system that adjusts bias choices with respect to a particular inductive policy.

Introduction

In order for an inductive learning program to define concepts that generalize beyond its training examples, it must incorporate *inductive bias* – assumptions that lead it to prefer certain inductive steps over others. Mitchell [Mitchell 1980] defined bias as "any basis for choosing one generalization over another other than strict consistency with the observed training instances." Systems bias their learning in many ways, including using restricted description languages, heuristics to search the hypothesis space, and domain knowledge to guide the search. The *strength* of a bias has been defined as the fraction of hypotheses considered by the learner within that bias relative to all possible hypotheses [Utgoff 1984].

More recently, machine learning researchers have worked to formalize the notion of inductive bias in terms of specific restrictions of or orderings to the hypothesis space [Rendell 1986], [Haussler 1988], [Dietterich 1991]. Such formalisms help to shed light on the problem of inductive bias, but something from Mitchell's original definition was lost in the process. We are gaining insight into the various ways of restricting and ordering hypothesis spaces. However, the difference between the bias choice and the strategy for making a bias choice often goes undiscussed. For example, a preference for simplicity may *lead to* the choice of a certain ordering (*e.g.*, preferring the shorter expression in some language), but they are not equivalent.

In this paper we define *inductive policy* as the strategy used to make inductive bias choices, based on underlying assumptions (and preferences) in the domain. Inductive policy decisions address tradeoffs with respect to different bias choices. Without addressing these tradeoffs, bias choices will be made arbitrarily. We present initial studies of tradeoffs not addressed much in the machine learning literature: trading off predictive accuracy for speed of learning, and trading off accuracy for safety in a domain where different errors have different costs. We also show how the value for the latter tradeoff can be represented explicitly.

Inductive Policy

To avoid confusion in the subsequent discussion let us call a choice of implementation restricting or ordering the space of possible hypotheses (possibly dynamically) a *bias choice*. A particular inductive policy is based on underlying assumptions of the learning domain and task and addresses the tradeoffs with respect to the different bias choices. As with bias choices the granularity of inductive policies can vary, and determining the exact boundaries between assumptions, policy, and bias choice may provide for interesting discussions. However, there is a qualitative difference between bias choices and the strategies for making them.

To illustrate the difference, let us consider a learning task where an underlying assumption is that the set of features given may not be adequate for learning the concept with a simple description language. Two somewhat different inductive policies come to mind immediately: (i) use a learning system with a more complex description language, or (ii) surround the first- order system with techniques for constructive induction (in effect, increasing the complexity of the description language). Each of these policies leads to different possible sets of bias choices (for example, the choice of a particular constructive induction implementation). The sets of bias choices are reduced by further, finer grained, policy decisions. Another example of underlying assumptions in a domain concern the total number of examples available for learning and the amount of processing power. With many examples and little computing power, the machine learning researcher immediately wants to consider different strategies than if the opposites were true, *e.g.*, (i) use a relatively inexpensive algorithm, (ii) use a technique for selecting a subset of examples (intelligent or random), or (iii) use an incremental learning method.

Decisions at the inductive policy level are made with respect to the underlying assumptions of a domain (or the preferences of the investigator) and the tradeoffs among them. Different inductive policies, and their associated bias choices, can lead to different learning performance. In order to make decisions at the policy level we need to understand the tradeoffs.

Explicitly addressing inductive policy is not new. Samuel, in his classic checkers program, decided to use a linear combination of terms from the domain realizing that this choice would prohibit him from expressing interactions among the terms, but would provide him with a simpler representation [Samuel 1963]. Later he decided to move toward the other end of the tradeoff spectrum, choosing the more complicated signature tables that would allow him to represent interactions among terms. Although previous work in machine learning has implicitly addressed such tradeoffs, we feel that it is useful to make the distinction between bias choice and policy explicit.

McCarthy argued [McCarthy 1958] that representing knowledge declaratively in a reasoning program was a prerequisite for a learning program to augment or edit it. Similarly, insofar as the possible bias choices of a learning program are explicitly defined, it is conceivable that a second-order system can tune them so that the first-order system can learn better or faster [Buchanan *et al.*, 1978]. Different inductive policies apply in different domains, leading to different bias choices. Whether these choices are made by hand or by a second-order system, their explicit representation facilitates their change.

Different Inductive Policies

Decisions are made at the inductive policy level that tie the bias choices to the underlying preferences and assumptions. Different bias choices may involve different learning programs, or, as far as the bias of the learning program has been made explicit, different bias choices within a given program. In this section we will investigate how different inductive policy decisions, reflecting different assumptions about tradeoffs in the learning task, can lead to differences in the performance of a learning program. More specifically, we show how different policies dealing with assumptions about the time available for learning can lead to differences in the predictive accuracies of the resultant concept descriptions. Then we illustrate how differences in inductive policy with respect to different prediction errors can lead to differences in the overall cost of using a learned concept description. Finally, we show how by using a function that represents explicitly the weight given to the costs of prediction errors, a system can learn a concept description that better satisfies the underlying cost assumptions.

The learning program used in this section is MC-RL, a multiclass version of the RL4 learning system [Clearwater & Provost 1990]. MC-RL uses an explicit bias representation, its partial domain model (PDM), to enable easy changes in bias choices based on decisions at the inductive policy level. In section 3.3 we use the ClimBS system [Provost 1992], which automatically adjusts MC-RL's bias based on explicitly represented bias adjustment operators. The focus of this paper is on the inductive policies used, rather than the particular systems. Space constraints prohibit more detailed description.

Tradeoff: accuracy vs. time to learn

One policy decision that must be made is based on assumptions about the amount of time available for learning. The bias choices made when a large amount of time is available will be different from those made when there is a time limit. In the literature, one sees analyses of learning programs' efficiencies, both analytically (usually with respect to asymptotic computational complexity) and empirically (usually on UCI database or artificial domains). Such comparisons are usually used for explicit or implied comparisons of learning programs. These comparisons are the beginning of a body of data on which inductive policy decisions can be based.

Little research addresses explicit limits on the time available for learning (other research in AI addresses resource bounded reasoning, see *e.g.*, [Bratman *et al.* 1988]). The tradeoff of time spent learning versus prediction quality is discussed in [desJardins 1991]; desJardins describes a method for using probabilistic background knowledge to select maximally relevant features to describe concepts (for incremental learning). Also with respect to incremental learning, [Pazzani & Sarrett 1990] discusses predictive accuracy as a function of the number of examples for conjunctive learning algorithms. [Clearwater *et al.* 1989] discusses incremental batch learning as a way of presenting intermediate results when a time limit is reached, and saving the "best rules so far" in case the time limit is less than the time to process a single batch. [Holder 1990] shows how the PEAK system can address the problem of a time limit on the use of a learned concept (and the use of EBL to speed it up). We discuss trading off predictive accuracy for short learning time for a batch learning system as a preliminary study. Our goal is to be able to learn a good concept description within a specified time limit.

If a basic assumption of a given learning task is that there will only be x time units for learning, the inductive policy followed will be different from that followed when there is no such limit. One inductive policy that might be followed in such a situation is: use a heuristic search that is guaranteed to terminate in x time units and return the best concept description the system could learn in that time. We have begun to investigate the relationship between the beam width of MC-RL's search of the space of syntactically defined rules, the time taken to learn, and the accuracy of the resultant concept description.

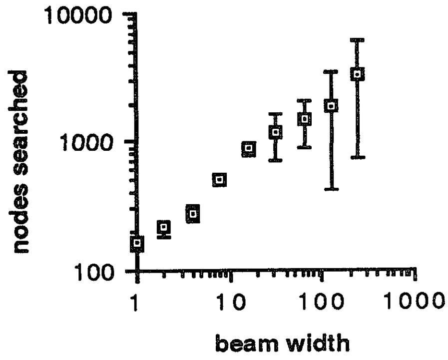

Figure 1: Increased search with increased beam width in mushroom domain.

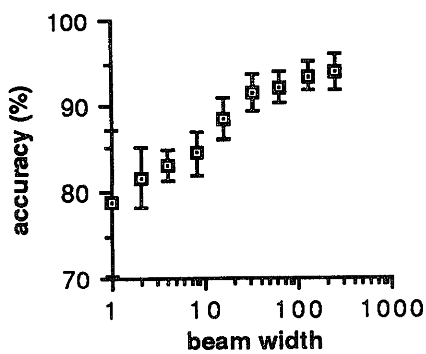

Figure 2: Increased accuracy with increased beam width in mushroom domain.

Analytic results predict that the search time for the MC-RL system will grow linearly with the beam width. Figure 1 shows the actual relationships found in the mushroom domain from the UCI repository (points show mean values over 10 runs with randomly selected training sets, error bars show 95% confidence interval using Student's t). These relationships are sublinear (note the logarithmic scales); the analysis could not take into account the contribution of some of MC-RL's heuristics for pruning the search space.

Figure 2 shows the corresponding classification accuracies of the resultant rule sets (points, as above; tested on separate test sets). Similar results were found in the automobile domain. The graceful degradation of the classification performance with smaller beam width indicates that it may indeed be profitable to determine the maximum beam width such that the search is guaranteed to terminate within the given x time units. Such a calculation would involve a determination of the time to search a single node on a given machine with a given PDM and number of examples, and a curve that is guaranteed to bound (from above) the actual run times (the analytically derived curve may suffice, with the determination of the constants).

Once the performance of the system within given time limits has been characterized, it can be compared with that of other approaches and higher level tradeoffs can be studied. For example, even if enough space is available for batch learning it might be found that it is better to use an incremental learning system for a certain class of time constrained learning tasks–*i.e.*, the gain in speed by using a policy of incremental learning outweighs the gain by using a quick heuristic search in a batch algorithm, to yield a concept description with the same accuracy (or perhaps a combination of the two policies gives the best results).

Other inductive policies take advantage of the relative speed with which a large portion of the target concept description can be learned. For example, part of an inductive policy might be: use a quick heuristic search, then switch to a more time consuming search using the knowledge gained with the quick search to guide and restrict subsequent learning. In addition, a quick search is useful for exploratory work in making other bias choices.

Better Safe than Sorry

Another set of assumptions that affect inductive policy decisions addresses the costs of making incorrect predictions or of failing to make a prediction (errors of commission or omission). Machine learning work usually treats these costs as equal and concentrates solely on predictive accuracy. One exception to this is the CRL system [Tcheng *et al.* 1989], which allows the user to specify domain dependent error metrics, which can then be used to guide the learning (results as to the method's efficacy are not presented). Another exception is the work of [Etzioni 1991], which studies the introduction of decision analytic techniques (which take into account costs, benefits and likelihoods) into an agent's control policy (and the use of learning to aid estimation). The decision to learn with sensitivity to prediction cost is an inductive policy decision that affects inductive bias choices (similar to the decision to be sensitive to the cost of measuring features, as in [Tan and Schlimmer 1990]).

We have investigated different inductive policies using MC-RL in the mushroom domain. MC-RL learns a set of rules, some of which predict that a mushroom matching the antecedent is poisonous, others predict that the mushroom is edible. This rule set is used as part of the knowledge base for an inference engine that gathers evidence from the rules that fire on an example, and combines the evidence to make its prediction. The inference engine simply finds all the rules that match the example and uses an evidence gathering function supplied for the domain when more than a single concept is predicted by the fired rules. The default evidence gathering function has the rules vote to determine the concept to predict.

In the mushroom domain the cost of making a mistake is lopsided. Under normal circumstances, no harm is done when an edible mushroom is classified as poisonous. In contrast, classifying a poisonous mushroom as edible is dangerous. In this domain, the assumption that a certain prediction is more costly than another should lead to a different inductive policy than that taken when one can assume that all mistakes can be weighted equally. Obviously, a completely safe policy would be not to even use a learning program; instead use a concept description that always predicts a mushroom is poisonous– dangerous predictions would never be made. However, this approach would never allow any mushroom to be eaten. The policy used by mushroom experts varies from expert to expert. A very conservative policy requires considerably more evidence, for example, than a less conservative one [Spear 1992].

Table 1 lists the results of several experiments in this domain with different inductive policies and associated bias choices. The experiments are described in detail below. The table lists the number of rules learned for a given experiment, the predictive accuracy (on a test set), and the percent of predictions that classify a poisonous mushroom as edible (% dangerous) when using the voting strategy and a "better safe than sorry" strategy in the inference engine. Better safe than sorry (BSTS) simply predicts that a mushroom is poisonous if any rule in the rule set classifies it so. A representative set of 1015 examples (every eighth) was chosen from the database for efficiency reasons; this was split randomly into training sets of size 100 and test sets of size 915. The results in Table 1 are averages over 10 runs; 95% confidence intervals are given.

Experiment 1 used a bias chosen empirically to give a high predictive accuracy. With the voting

Exp	rules (ave)	Voting		Better Safe Than Sorry	
		Accuracy	risk	Accuracy	risk
1	25	94.4±1.4	2.6 ±1.6	92.6±2.6	1.1±0.62
2	25	94.3±2.0	1.1±0.54	91.7±2.2	0.91±0.51
3	15	86.4±3.6	0.25±0.26	86.1±3.4	0.23±0.26
4	11	75.5±12.6	0.22±0.23	75.4±12.5	0.22±0.23
5	818	88.8±1.6	0.48±0.52	85.7±2.6	0.20±0.26
6	807	87.0±3.1	0.22±0.20	84.5±4.1	0.04±0.14

Table 1: Experiments comparing (%) accuracy (correct predictions/total * 100) and (%) risk (dangerous predictions/total * 100) associated with rules learned with different policies and biases in the mushroom domain. Experiments 1-6 are described in the text (below).

strategy approximately half of the incorrect predictions made were of the dangerous type–over 2 percent of the test examples. With BSTS this fraction is reduced to approximately 1 percent. The rest of the experiments tested different inductive policies (and the associated bias choices) designed to lower the fraction of predictions that were dangerous. In Experiment 2, MC-RL was run as in Exp. 1 for the rules predicting mushrooms to be poisonous; however, the rules for edibility were restricted not to cover any negative training examples and to be simpler in form (only 3 conjuncts allowed instead of 5) to avoid possible fitting of the data. The result was that the fraction of dangerous predictions was reduced slightly, without a significant decrease in the predictive accuracy. Experiment 3 was identical to Experiment 2, except that each edibility rule was forced to cover a larger fraction of the positive examples (40% instead of 20%). The fraction of dangerous predictions was reduced again, accompanied by a decrease in predictive accuracy. Forcing each edibility rule to cover an even larger fraction of positive examples (Exp. 4–60%) did not decrease the dangerous predictions, but did decrease the predictive accuracy.

Experiments 2 through 4 took the policy that the poisonous classification accuracy should be high, but that only rules that are in some sense "safe" should be learned for the edibility class. Experiments 5 and 6 combine this with a different policy. While edibility rules are learned as in Exp. 2, when searching for rules for the poisonous class MC-RL is instructed to learn a large, highly redundant set by turning off its heuristic for selecting a small "good" rule set. The hope is that many alternative descriptions of poisonous mushrooms will do a better job of catching the few examples that had previously slipped by. This is in fact seen to be the case, especially when used in the BSTS inference engine. The final experiment combines the best of the two inductive policies–learning only "good" rules for edibility (by setting the positive threshold for edibility rules to 0.4) while learning a highly redundant set of rules for the poisonous class. When used in the BSTS inference engine, the fraction of dangerous predictions is reduced to 0.04 percent (in all but 1 of 10 runs it was zero) and the predictive accuracy was still 84.5 percent.

These experiments show how different policies can lead to different tradeoffs of accuracy for safety of predictions. They show particular policies that are useful in the mushroom domain; their utility in other domains has yet to be shown. Specifying *a priori* the specific bias choices that will perform best given a high level specification of a policy is a matter for further investigation. The method used above was to determine empirically a good set of bias choices, guided by the inductive policy. The next section shows how the tradeoff of accuracy for safety can be represented explicitly, and a system for bias adjustment can determine the bias choices that yield a good concept description.

Explicit Specification of Inductive Policy for a Bias Adjustment System

The ClimBS system [Provost 1992] performs a hill climbing search in MC-RL's bias space, incrementally constructing a concept description across biases (using a greedy rule set pruning heuristic, similar to that described in [Quinlan 1987]) and using the learned knowledge to guide and restrict its bias space search. ClimBS is provided with a starting bias, bias transformation operators, and a rule set evaluation function. It uses the transformation operators to create tentative candidate biases, learns rules with each, combines these with the previously learned rule set, and chooses the best of the tentative sets with respect to the evaluation function. The current bias is set to the bias with which the best rule set was learned; and the process iterates until one of several stopping criteria is met.

ClimBS is a second-order system for automatically adjusting inductive bias of a first-order system (MC-RL). ClimBS allows for an (partial) explicit specification of bias choices corresponding

Experiment	No. rules	Accuracy	dangerous
7	8±2	96.3±1.3	2.7±2.0
8	9±1	95.2±0.9	0.87±0.48
9	8±2	91.5±5.2	0.74±0.45
10	6±2	82.2±14.8	0.16±0.28

Table 2: Experiments comparing the accuracy and risk of rules learned with different inductive policies represented as bias evaluation functions in the ClimBS bias adjustment system. Experiments 7-10 are described in the text (below). Accuracy and risk are defined as in Table 1.

to bias adjustment policies. The particular set of bias transformation operators, combined with ClimBS' hill climbing search and the bias evaluation function, correspond to a policy for changing first-order bias choices based on an evaluation of the results of learning. The set of operators used to generate the results below operated on the positive threshold for individual rule performance, the negative performance threshold, the complexity of the description language (the maximum number of conjuncts allowed in a rule), and the beam width of the heuristic search. The initial bias was very restrictive (the parameters listed above were set at 0.9, 0, 1, and 1, respectively), and the operators weakened the bias along the several dimensions. This corresponds to a policy of trying to learn "good" rules first, and subsequently reducing the standards to complete the concept description.

Table 2 shows the results of several experiments in which ClimBS was used to learn rules for the mushroom domain. The bias evaluation function was varied across the experiments to reflect different assumptions about the tradeoff of accuracy vs. the cost of making dangerous errors. The table lists the number of rules learned, the predictive accuracy of the rule set, and the percent of the predictions that are dangerous. As above, the results are averaged over 10 trials, 95% confidence intervals are given; 100 (randomly selected) examples were used for learning, 400 for bias evaluation, and 515 for testing.

In Experiment 7, the default evaluation function (only consider predictive accuracy, use voting strategy) was used. As in Experiment 1, although the classification accuracy is impressive, over two percent of the examples were dangerously classified as edible, when they were actually poisonous. In Experiment 8 the evaluation function compared biases based on predictive accuracy, using the BSTS evidence gathering function. The accuracy of the resultant descriptions are not degraded much, while the fraction of dangerous predictions is reduced to one-third its previous value. Recall that switching to the BSTS engine provided a significant reduction in dangerous predictions.

Experiments 9 and 10 use a linear combination of predictive accuracy and number of dangerous predictions to evaluate the rule sets learned with different biases. In particular, the function used was: f = number of correct predictions - w * number of dangerous predictions. In Experiment 9, w = 10; in Experiment 10, w = 50. One can see the tradeoff of classification accuracy for safe predictions that is manifest in the rule sets learned by ClimBS. The 0.16 percent dangerous prediction rate obtained in Experiment 10 is still four times that of Experiment 6. However, ClimBS was not equipped with an operator that would turn off the heuristic for selecting a small "good" rule set (as was done in Section 3.2); this heuristic was always on. ClimBS performance is compatible with that of Experiment 3. Note that the variance of the classification accuracy is larger when the more complicated evaluation functions are used. This may be a side effect of the convergence criterion used. (If over the last 5000 (MC-RL) nodes searched the concept description had not improved, ClimBS terminated its search).

ClimBS allows for the specification of the relative value of different rule sets with respect to their performance. This explicit representation of how the system should deal with the accuracy/safety tradeoff allows the system to adjust the first-order bias to find a concept description that gives a good score with respect to this function, and thereby performs well with respect to the tradeoff. The problem of specifying the bias evaluation function (in light of a particular inductive policy) remains. However the function from Experiment 10 was the first chosen. We assert that specifying this function and letting ClimBS automatically adjust the bias is easier than the manual bias adjustment, because its specification is more directly related to the assumption in question (*i.e.*, the relative importance of the different prediction errors and predictive accuracy).

Conclusions

We believe that the results showing that predictive accuracy degrades gracefully with shorter times taken to learn are not tied to the particular bias choices used, but would be shown with different heuristic searches or different evaluation functions in the beam search used. With respect to the accuracy/safety tradeoff, we believe that the increases in safety shown with manual bias adjustment in MC-RL are due to the policies (learn only "good" rules for edibility, learn highly redundant sets of rules for poisonous) used to guide the selection of bias choices, more than to the bias choices themselves. With ClimBS, we believe the power comes from the policy of using an evaluation function that takes the specific tradeoff into account when making bias choices, rather than from the particular evaluation function used (the bias choice). Substantiating or refuting these claims is future work.

These claims of the importance of concentrating on inductive policy in addition to concentrating on bias choices must be qualified by repeating the theme that we have stressed throughout the paper. Concentrating on inductive policy provides focus for studies of inductive bias choices. In turn, the tradeoffs with respect to the different possible bias choices help to guide the selection of policy for a given task. Without studies of how different bias choices affect learning systems' performances with respect to these tradeoffs, bias choices will be made arbitrarily–without an inductive policy.

Acknowledgments

This work benefited from discussions with Scott Clearwater, Randy Jones, Tom Mitchell, Jeff Schlimmer, Kurt VanLehn, and the Pitt machine learning discussion group. This work was supported by NLM grant 1-R01-LM05104 and an IBM Graduate Fellowship.

References

Bratman, M.; Israel, D.; and Pollack, M. 1988. Plans and Resource- Bounded Practical Reasoning. *Computational Intelligence* 4(4), 349- 355.

Buchanan, B.; Johnson, C.; Mitchell, T.; and Smith, R. 1978. Models of Learning Systems. In J. Belzer (ed.), *Encyclopedia of Computer Science and Technology* 11, 24-51.

Clearwater, S.; Cheng, T.; Hirsh, H.; and Buchanan, B. 1989. Incremental Batch Learning. In Proc. of ML-89, 366-370. Morgan Kaufmann.

Clearwater, S. and Provost, F. 1990. RL4: A Tool for Knowledge-Based Induction. In Proc. of the 2nd Int. IEEE Conf. on Tools for AI, 24-30. IEEE Computer Society Press.

desJardins, M. 1991. Probabilistic Evaluation of Bias for Learning Systems. Proc. of ML-91, 495-499. Morgan Kaufmann.

Dietterich, T. 1991. Machine Learning: Issues, Answers, and Quandaries. Keynote Lecture, AAAI-91.

Etzioni, O. 1991. Embedding Decision-analytic Control in a Learning Architecture. *Artificial Intelligence* 49 (1991), 129-159.

Haussler, D. 1988. Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Learning Framework. *Artificial Intelligence* 36, 177-221.

Holder, L. 1990. The General Utility Problem in Machine Learning. In Proc. of ML-90, 402-410. Morgan Kaufmann

McCarthy, J. 1958. Programs with Common Sense. Reprinted in R. Brachman and H. Levesque (Eds.), *Readings in Knowledge Representation*, 299-308. Morgan Kaufmann, 1985.

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

Pazzani, M. and Sarrett, W. 1990. Average Case Analysis of Conjunctive Learning Algorithms. In Proc. of ML-90, 339-347. Morgan Kaufmann.

Provost, F. 1992. Searching the Bias Space: Policies for Inductive Bias Adjustment. Ph.D. thesis, forthcoming.

Quinlan, J. 1987. Generating Production Rules from Decision Trees. Proc. of IJCAI-87.

Rendell, L. 1986. A General Framework for Induction and a Study of Selective Induction. *Machine Learning* 1 (1986), 177-226.

Samuel, A. 1963. Some Studies in Machine Learning Using the Game of Checkers. In E. Feigenbaum and J. Feldman (Eds.), *Computers and Thought* 71-105. McGraw-Hill.

Spear, M. 1992. Vice President of Research, Sylvan Spawn Laboratories. Private Communication.

Tan, M. and Schlimmer, J. 1990. Two Case Studies in Cost-Sensitive Concept Aquisition. In Proc. of AAAI-90, 854-860. Morgan Kaufmann.

Tcheng, D.; Lambert, B.; Lu, S.; and Rendell, L. 1989. Building Robust Learning Systems by Combining Induction and Optimization." In Proc. of IJCAI-89, 806-812. Morgan Kaufmann.

Utgoff, P. 1984. Shift of Bias for Inductive Concept Learning. Ph.D. thesis, Rutgers University.