Selection of Learning Methods Using an Adaptive Model of Knowledge Utility

Lawrence B. Holder
University of Texas at Arlington
Department of Computer Science Engineering

Abstract

Experiments measuring knowledge utility during the course of learning reveal a common behavior among several learning methods. As the amount of learned knowledge increases, the utility (performance) of the knowledge initially increases, but eventually degrades. A multistrategy learning system based on a model of this common behavior selects a learning method according to the model's prediction of achievable performance. The model-based adaptive control (MBAC) approach provides this capability by maintaining models for each learning method. MBAC also predicts the amount of knowledge to be acquired by the method in order to avoid the degradation of performance. MBAC unifies multiple learning strategies based on the common behavior of the utility of their knowledge during the course of learning.

Key words: utility problem, overfit, performance response, adaptive control

1. Introduction

Machine learning research has developed several empirical and analytical learning methods that demonstrate performance improvements due to learned knowledge. Unfortunately, more in-depth experimentation with these methods reveals that the performance improvement is only temporary. As the methods generate more and more knowledge, the performance for which they were designed to improve, eventually degrades. This phenomenon relates to the overfit problem in empirical learning and the utililty problem in analytical learning.

In order to avoid the overfit/utility problem, the learning method must determine the correct subset of the learnable knowledge that maximizes performance. Trying all possible subsets is computationally infeasible; therefore, most learning methods generate knowledge from specific to general or general to specific. Given that the learning method acquires knowledge in order of generality (specificity), avoiding the overfit/utility problem reduces to generating the correct *amount* of learned knowledge.

In view of the large number of alternative methods available for improving a given performance dimension (e.g., classification accuracy or problem-solving speed), a more general utility problem exists in determining not only the correct amount of learned knowledge, but also the correct method for learning this knowledge. Overcoming this *general utility problem* requires a new con-

trol mechanism for determining the correct learning method and amount of learned knowledge.

As Section 2 will demonstrate, a common behavior exists among several learning methods due to the general utility problem. As depicted in Figure 1, performance initially increases, but eventually degrades. Section 3 describes he model-based adaptive control (MBAC) approach, which maintains a model of this common behavior for each learning method. The control mechanism uses the model to determine the amount of learned knowledge necessary to achieve a desired level of performance, and selects appropriate learning methods according to the shape and certainty of their associated models.

Using one model to describe the behavior of multiple learning methods simplifies the integration of these methods. Instead of integrating on the basis of a common knowledge representation, a multistrategy learning system can integrate on the basis of the performance/knowledge relationship while maintaining individual knowledge representations for each method.

2. General Utility Problem

The *general utility problem* in machine learning is the degradation of performance due to increasing amounts of learned knowledge (Holder, 1990). The term derives from the *utility problem* used by Minton (1988) to describe this phenomenon in analytical learning, but generalizes to other machine learning strategies.

Other researchers have observed the ubiquity of the utility problem in machine learning paradigms. Carlson et al. (1990) compare the utility problem in deductive learning to the problems of noise and overfit in inductive learning. Etzioni (1988) alludes to the general utility problem as he proposes a hypothesis filter for all learning methods.

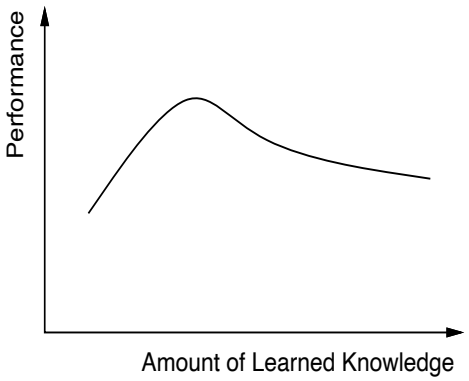

Figure 1: Performance response of a learning method that suffers from the general utility problem.

2.1 Performance response

A useful tool for analyzing the general utility problem in machine learning is the *performance response*, which measures the performance of the learned knowledge during the course of learning. Figure 1 shows the performance response of a learning method that suffers from the general utility problem.

The horizontal axis measures the amount of learned knowledge. The units along this axis represent the change in learned knowledge made by a knowledge transformation. A knowledge transformation is a decomposition of the learning program into less complex operations affecting the learned knowledge. For example, one decomposition of a splitting algorithm is a single split, and one decomposition of a neural network learning algorithm is a single cycle.

The vertical axis measures the performance of the learned knowledge after each transformation. The method for measuring performance depends on the learning algorithm. For empirical learning the performance response curve plots the classification accuracy of the knowledge after each transformation. For analytical learning the performance response curve plots the inverse of the CPU time needed by the knowledge to

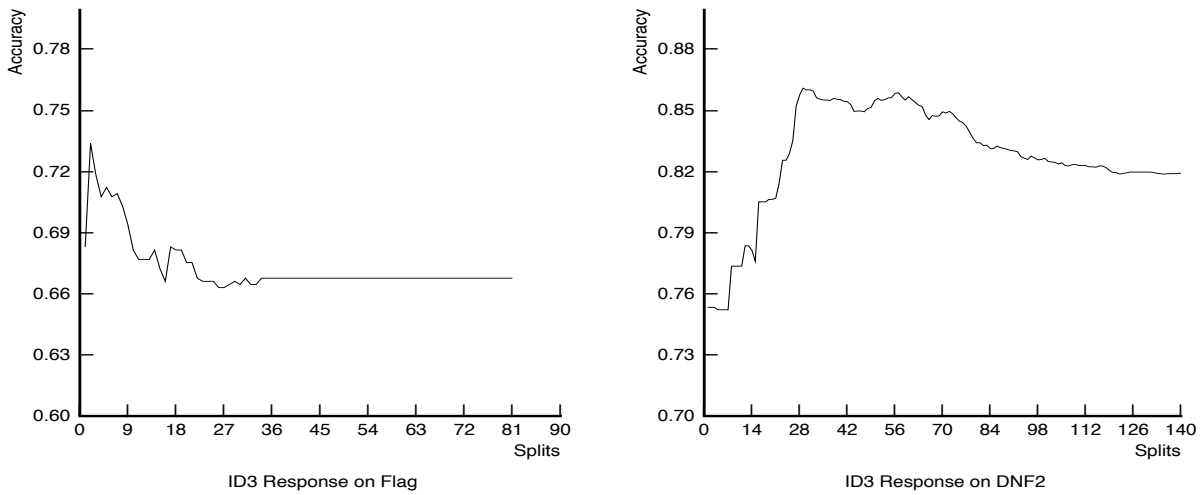

Figure 2: Performance response of ID3.

solve a set of test problems.

2.2 Empirical learning

The ID3 program (Quinlan, 1986) induces decision trees by recursively splitting the given set of training instances. The performance response for ID3 plots the accuracy of the decision tree on a separate set of testing instances after each split. Splits are performed in a breadth-first order so that overfit is deferred to the later splits. Figure 2 shows the performance response of ID3 on the Flag and DNF2 domains (see Section 2.4) when splitting to pure nodes (contain instances of only one class). As the figure illustrates, the performance responses follow the trend of Figure 1. In order to combat overfit in ID3, Quinlan developed chi-square pre-pruning (Quinlan, 1986) and reduced-error post-pruning (Quinlan, 1987). Experimentation by Holder (1991) shows that these techniques alleviate some overfitting, but the general utility problem trend remains. Performance response curves for other splitting methods, PLS1 (Rendell, 1983) and CART (Breiman et al., 1984), follow a similar pattern (Holder, 1991).

Experimentation by Michalski (1989) on AQ15 and Holte et al. (1989) on CN2 reveal the general utility problem in set-covering methods as the method learns an increasing number of disjuncts. Plots of the performance response for AQ15 on several medical domains reveal the general utility problem trend of Figure 1 (Holder, 1991).

Error back-propagation (Rumelhart et al., 1986) is a connectionist learning method that also suffers from the general utility problem. As the number of cycles increases, the network overfits the training instances. Holder (1991) plots the performance of a network after each cycle, revealing the same general utility problem trend.

2.3 Analytical learning

As an analytical learning system acquires increasing amounts of macro-rules, the cost of retaining the rules eventually exceeds their benefit. Minton called this phenomenon the *utility problem* and offered the Prodigy system as a solution (Minton, 1988). The system empirically estimates the utility of each rule and discards a rule when the utility becomes negative. Experimentation on the Soar system has uncovered similar results (Tambe and Newell, 1988).

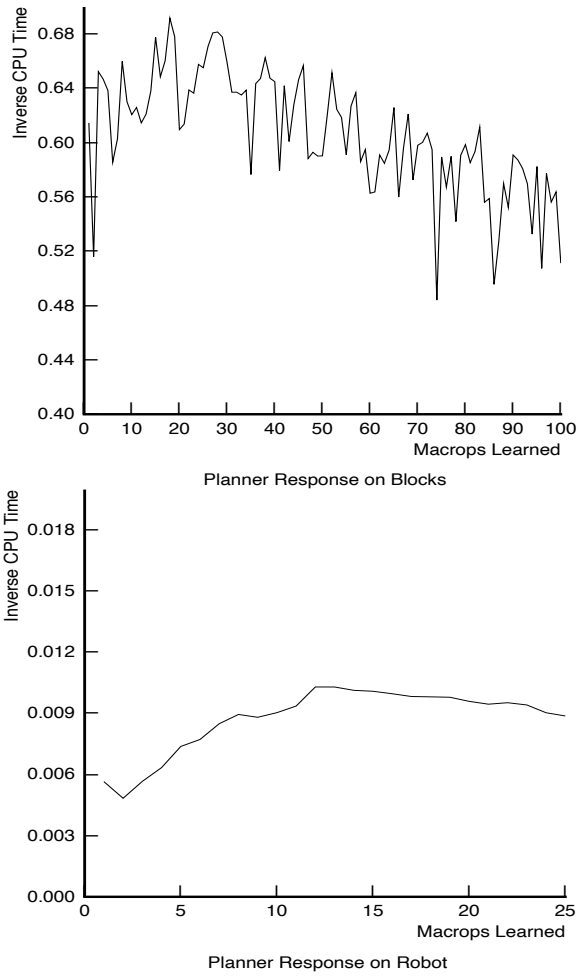

Figure 3: Planner performance response.

Although experimentation with Prodigy and Soar confirms the existence of the utility problem, the experiments typically do not show the performance response of the system. Figure 3 plots the performance response of a simple analytical learning method. The method consists of a forward-chaining planner and a Strips plan generalizer (Fikes et al., 1972). Two domains are used in the experimentation: blocks and robot (see Section 2.4).

The experiments proceed by solving a training problem in the domain, generalizing the resulting plan, adding the generalized plan (macrop) to the set of available operators, and then measuring the amount of CPU time needed to solve a separate set of test problems. The x-axis of the performance response is the number of learned macrops. The y-axis measures the inverse of the CPU time needed to solve the set of test problems. Although erratic in the blocks domain, the response curves in Figure 3 follow the general utility problem trend.

2.4 Trends

The previous sections verify the existence of the general utility problem in several machine learning methods. The performance responses of these methods follow the general trend illustrated in Figure 1. Adopting this trend as a model of the performance response permits the control of the general utility problem by constraining the amount of learned knowledge to reside at the point corresponding to the peak performance.

Tables 1 and 2 quantify the possible performance gains by using this model-based control of the amount of learned knowledge. Each entry in the tables is the final performance as a percentage of peak performance averaged over ten performance response curves. Table 1 lists entries for several of the previously described empirical learning methods on five different domains. ID3 Chi stands for ID3 with chi-squared pre-pruning with confidence of 99.9%. ID3 RE stands for ID3 with reduced-error post-pruning. BP4 stands for error back-propagation with four hidden units. Table 2 lists entries for the Planner analytical learner on two domains. Note that the entries in Table 2 can be arbitrarily deflated by allowing the analytical learner to acquire more macrops.

Table 1: Final performance as a percentage of peak performance for empirical learners.

	Domain				
Method	BC	Flag	Flare	Vote	dnf2
ID3	91.2	88.2	95.0	97.6	93.6
ID3 Chi	90.8	89.9	96.1	97.0	97.2
ID3 RE	98.6	95.4	98.7	99.7	100.3
PLS1	87.9	96.3	97.7	98.1	92.8
BP4	82.8	89.8	88.2	92.6	91.1

Table 2: Final performance as a percentage of peak performance for analytical learner.

	Domain	
Method	Blocks	Robot
Planner	67.4	76.1

For the empirical learners, the Breast Cancer (BC), Flag, Flare and Voting (Vote) domains come from the UC Irvine machine learning databases. The DNF2 domain is defined by a DNF concept over forty binary-valued features that appears in (Pagallo and Haussler, 1990). The Planner analytical learning method uses two domains: blocks and robot. The blocks domain consists of four operators for moving blocks in the blocks-world. The robot domain consists of eight operators using a robot to move boxes within a layout of connected rooms. See (Holder, 1991) for a complete description of

these domains.

As shown in Tables 1 and 2, the final performance is less than the peak performance for all but one case. Thus, the ability to constrain the amount of learned knowledge to the point corresponding to peak performance will improve the performance of the learner. Although individual methods exist for alleviating the general utility problem in each particular learning method, the performance response model offers a general method for avoiding the general utility problem in many machine learning methods.

3. MBAC Approach

The model-based adaptive control (MBAC) approach uses the trend identified in Section 2 as a model to control the amount of learned knowledge in order to maintain utility. The model describes the performance response for a particular task domain, performance dimension and knowledge transformation (learning method). MBAC instantiates the model as a parameterized curve and fits the curve according to previously observed samples from actual performance responses. Using this model of the performance response curve, MBAC determines the point on the curve having the desired level of performance and recommends learning the amount of knowledge corresponding to this point.

The proposed MBAC approach resembles an adaptive control loop. First, the performance element uses the knowledge to perform some task. MBAC compares the performance on the task to the user-defined performance objectives. This performance comparison serves as feedback to improve the model's estimate of the true performance response. Using the updated model, MBAC decides how to transform the knowledge in order to achieve and maintain the desired performance objectives.

```
proc Select-Transformation (models, thr)
begin
  foreach m in models do
    decision = Parabolic-Decision(m, thr)
    model-move(m) = move(decision)
    model-error(m) = |thr − perf(decision)|
  models = sort(models, error, <, certainty, >)
  return(first(models))
end
```

Figure 4: Transformation selection.

3.1 Transformation selection

The MBAC approach uses a parabolic model of the performance response. This model assumes that most performance objectives have thresholds near the peak of the performance response. In this case, a model of the peak is sufficient for controlling the amount of learned knowledge near the peak. Figure 4 describes the Select-Transformation procedure, which takes a performance threshold *thr* and a set of models pertaining to the performance dimension of the threshold. The procedure returns the best model corresponding to the best transformation for achieving the threshold. The *move* field of the model contains the recommended move according to the parabolic model.

Select-Transformation computes the decision based on the parabolic estimate of the data for each model (the Parabolic-Decision procedure). A decision consists of a move (number of transformations) along the amount of learned knowledge axis and the predicted performance after performing the move. The model error is the absolute value of the difference between the threshold and the predicted performance. Next, Select-Transformation sorts the selected models in ascending order of model error. Models with the same error are further sorted in descending order of model certainty (standard devi-

ation). The best model is the first model in the set of sorted models.

3.2 Experimentation

MBAC selects transformations according to the corresponding model's ability to achieve the performance threshold. This experiment compares the parabolic model's predicted performance to the actual performance obtained by performing the recommended number of transformations. The results indicate that the selected transformation achieves predicted performance and outperforms the unselected transformations.

First, the experiment uses randomly-selected training and testing sets to produce ten response curves for each combination of task domain and transformation. These curves provide the data points for fitting the parabolic models. Next, the same process generates ten more response curves to be used for testing the recommendations of the models. The performance threshold is set at a point higher than that achievable by any learning method; therefore, the models predict their achievable peak performance. Table 3 shows the model's predicted performance, standard deviation, and the actual performance achieved by performing the recommended number of transformations on the average of the ten testing response curves.

The results indicate that the model predictions are accurate. Since the transformation selection procedure in Figure 4 sorts the models based on their predictions, the procedure will perform well at choosing the best transformation. The results also show that the actual performance achieved by the transformations is within one standard deviation of the predicted performance for all but BP2 and BP4 on the Flag task domain. The results show that the highest predicted performance correlates with the highest actual performance in all but the Flare domain (where the difference between the first and second best actual performance is small). Therefore, the transformation selection procedure, which picks the highest predicted performance, consistently chooses the best (or near best) transformation. Further experimentation in (Holder, 1991) demonstrates MBAC's ability to accurately model the performance response, build models starting with no samples from the performance response curve, and transfer knowledge about one domain to another new domain.

4. Conclusions

Successful application of learning methods requires the selection of the appropriate learning method and the acquisition of the appropriate amount of knowledge by the method. Model-based adaptive control (MBAC) addresses these two control dimensions of the general utility problem by using a model of the performance response trend common among several learning methods.

Several systems relate to the adaptive approach in MBAC. The MetaLEX system (Keller, 1987) transforms search control knowledge according to two performance objectives. MetaLEX uses a qualitative model of the relationship between knowledge and performance that is restricted to the two performance objectives. MBAC's model applies to multiple learning strategies and multiple performance objectives. The VBMS system (Rendell et al., 1987) and the AIMS system (Tcheng et al., 1991) learn models for selecting an appropriate empirical learning method. Both systems use a general method for learning the relationship between performance objectives and learning method. MBAC advocates a more constrained model describing less-complex learning transformations. The MBAC system will benefit most

Table 3: Parabolic model's predicted and actual performance on empirical learning methods. The standard deviation of the model is shown in parentheses.

	Breast Cancer		Flag		Flare		Voting	
	Predicted	Act.	Predicted	Act.	Predicted	Act.	Predicted	Act.
ID3	0.705(0.042)	0.697	0.762(0.063)	0.771	0.814(0.017)	0.815	0.959(0.076)	0.947
PLS1	0.712(0.037)	0.706	0.776(0.041)	0.795	0.825(0.017)	0.813	1.003(0.058)	0.960
BP2	0.353(0.117)	0.262	0.590(0.107)	0.472	0.393(0.257)	0.441	0.912(0.032)	0.903
BP4	0.403(0.138)	0.466	0.593(0.085)	0.477	0.407(0.189)	0.415	0.917(0.028)	0.911
BP8	0.501(0.077)	0.503	0.612(0.096)	0.526	0.404(0.163)	0.379	0.928(0.035)	0.927
BP16	0.450(0.112)	0.447	0.549(0.093)	0.591	0.412(0.148)	0.447	0.927(0.020)	0.908

from the development of formal models to replace the current empirical models. Holder (1991) describes some possible formal models for several learning methods.

Refinement of the model-based adaptive control approach through further experimentation and incorporation of formal models will evolve the approach into a general methodology for maintaining the utility of learned knowledge. At the heart of the methodology will be the model of the performance response whose common shape serves to integrate multiple learning strategies under one framework.

Acknowledgements

I would like to thank Larry Rendell and the members of the Inductive Learning Group at University of Illinois for their helpful suggestions. Thanks to Ray Mooney, Jude Shavlik and Carl Kadie for their implementations of the learning methods. This research was partially funded by the National Science Foundation under grant IRI 8822031.

References

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

B. Carlson, J. Weinberg, and D. Fisher. Search control, utility, and concept induction. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 85–92, 1990.

O. Etzioni. Hypothesis filtering: A practical approach to reliable learning. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 416–429, 1988.

R. E. Fikes, P. E. Hart, and N. J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 4(3):189–208, 1972.

L. B. Holder. The general utility problem in machine learning. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 402–410, 1990.

L. B. Holder. *Maintaining the Utility of Learned Knowledge Using Model-Based Adaptive Control*. PhD thesis, Department of Computer Science, University of Illinois at Urbana–Champaign, October 1991.

R. C. Holte, L. E. Acker, and B. W. Porter. Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh IJCAI*, pages 813–818, 1989.

R. M. Keller. *The Role of Contextual Knowledge in Learning Concepts to Improve Performance*. PhD thesis, Department of Com-

puter Science, Rutgers University, January 1987.

R. S. Michalski. How to learn imprecise concepts: A method based on two-tiered representation and the AQ15 program. In Y. Kodratoff and R. S. Michalski, editors, *Machine Learning: An Artificial Intelligence Approach III*. Morgan Kaufmann, 1989.

S. Minton. *Learning Search Control Knowledge: An Explanation-Based Approach*. Kluwer Academic Publishers, 1988.

G. Pagallo and D. Haussler. Boolean feature discovery in empirical learning. *Machine Learning*, 5(1):71–100, 1990.

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

J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234, 1987.

L. Rendell, R. Seshu, and D. Tcheng. Layered concept learning and dynamically-variable bias management. In *Proceedings of the Tenth IJCAI*, pages 308–314, 1987.

L. A. Rendell. A new basis for state-space learning systems and a successful implementation. *Artificial Intelligence*, 20(4):369–392, 1983.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing, Volume 1*, chapter 8, pages 318–362. MIT Press, 1986.

M. Tambe and A. Newell. Some chunks are expensive. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 451–458, 1988.

D. K. Tcheng, B. L. Lambert, S. C. Lu, and L. A. Rendell. AIMS: An adaptive interactive modeling system for supporting engineering decision making. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 645–649, 1991.