Noise-Tolerant Conceptual Clustering*

Douglas H. Fisher
Department of Computer Science
Box 67, Station B
Vanderbilt University
Nashville, TN 37235
dfisher@vuse.vanderbilt.edu

Abstract

Fisher (1987a,b) introduced a performance task for conceptual clustering: *flexible* prediction of arbitrary attribute values, not simply the prediction of a single 'class' attribute. This paper extends earlier analysis by considering the effects of *noise* and other environmental factors. The degradation in flexible prediction accuracy that results from noise is mitigated by 'preferred' prediction points for individual attributes. Methods that identify these prediction points are inspired by *pruning* in learning from examples. We extend these noise-tolerant techniques to untutored learning. In addition, prediction point preferences shed light on relationships between conceptual clustering, case-based, and default reasoning.

1 Introduction

Machine concept learning has traditionally been concerned with *learning from examples* (e.g., Quinlan, 1986), which assumes that observations are identified as members of *a priori* known classes (e.g., diseases); the learner must characterize the observations of each class. In contrast, *conceptual clustering* methods (Michalski & Stepp, 1983; Fisher, 1987b; Cheeseman, Kelly, Self, Stutz, Taylor, *k* Freeman, 1988) discover, as well as characterize meaningful classes.

In principle, learning should improve an organism's performance at some task(s). In learning from examples a performance task is apparent: improve prediction of class membership (e.g., diagnose the illness of a patient). On the otherhand, conceptual clustering is not traditionally associated with a performance task (Michalski *k* Stepp, 1983; Cheeseman, Kelly, Self, Stutz, Taylor, *k* Freeman, 1988). Fisher (1987a,b) proposes that a performance task for conceptual clustering is *flexible* prediction: the prediction of multiple attributes, not simply a single class 'attribute'. For example, a learning from examples system may attempt to optimize correct prediction of a congressperson's political party (i.e., class) from information about their congressional voting record (i.e., 'attributes' like their vote on farm *aid* or the *MX missile).* In contrast, flexible prediction is concerned with simultaneously improving prediction along all dimensions (e.g., party, farm aid, and *MX missile).* While the performance task of learning from examples has implications for expert system construction (Quinlan, 1986; Bareiss *k* Porter, 1987), flexible prediction permeates common-sense reasoning. Despite the importance of flexible prediction, few systems have been concerned with it (Kolodner, 1983; Lebowitz, 1982), much less systematically characterized with respect to it.

This paper explores the impact of noise on flexible prediction accuracy using Fisher's (1987b) COBWEB system. Weaknesses in the face of noise motivate two extensions to COBWEB that are inspired by research on *pruning* in learning from examples. Our extensions are applicable to other untutored learning systems and shed light on relationships between conceptual clustering, case-based, and default reasoning.

2 COBWEB and Flexible Prediction

COBWEB incrementally builds classification trees from objects that are described by nominal attribute - value pairs. For example, consider the tree of Figure 1 over voting records of U.S. Senators (Fisher, 1988). Stored at each node are the value distributions of each attribute over the objects classified under the node. Consider node N1, which classifies many senators voting 'yes' on budget cuts [P(Budget-cuts = yes \N]) = 0.88]. More generally, probabilities at a node are conditioned on membership in the node's parent, so that the probabilities of N1 are (trivially) conditioned on classification at the root, while probabilities at N2 assume classification at N1. Each node is a. *probabilistic concept* (Smith *k* Medin, 1981); the classification tree is a *probabilistic concept tree.*

Each tree level contains sibling classes that collec-

*This work was supported by a grant from the Vanderbilt University Research Council.

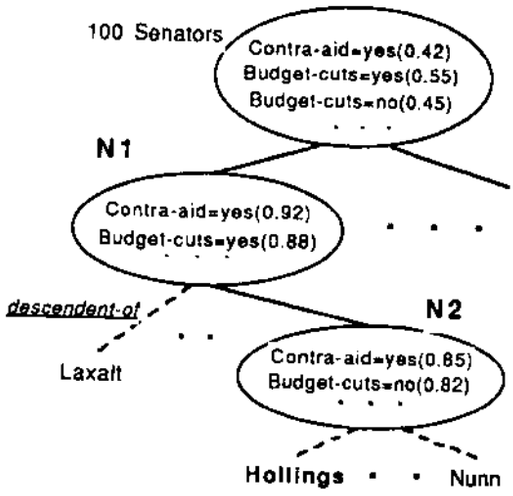

Figure 1: A partial concept tree over congressional voting records.

tively partition the observed objects. COBWEB incorporates a new object into the class that best matches the object according to *category utility* (Gluck & Corter, 1985), a measure that rewards the formation of object classes that maximize 'prediction ability'. More formally, the category utility of a class, C_k, is a function of the *expected number* of attribute values that can be correctly predicted about members of the class, $E(\#CorrectPredictions|C_k)$. If $P(A_i = V_{ij}|C_k)$ is the probability that a particular value, V_{ij}, is true of C_k members and the value will be predicted with the same probability then this expectation can be further formalized in terms of conditional probabilities that are stored at tree nodes: $E(\#CorrectPredictions|C_k) = \sum_i \sum_j P(A_i = V_{ij}|C_k)^2$. Category utility has some additional complexities (Gluck & Corter, 1985; Fisher, 1987b), but for our purposes we assume

$$P(C_k) \sum_i \sum_j P(A_i = V_{ij}|C_k)^2,$$

where $P(C_k)$ is the proportion of the observations to which the expectation applies; $P(C_k)$ is the probability that the benefits (expectations) of a class will be realized.

COBWEB incrementally filters objects into appropriate classes based on category utility. A new object is evaluated with respect to a class by tentatively placing the object in the class; each class's attribute-value distributions are tentatively updated to reflect the values of the new object. These distributions are actually stored as integer counts: $P(A_i = V_{ij}|C_k) = (\#$ of C_k objects with $V_{ij})/(\#$ of C_k objects). Probabilities are computed from the newly (tentatively) updated distributions and the category utility of the class is computed. The class that maximizes category utility after adding the new object is chosen to classify the object and appropriate distributions are updated permanently. This process is recursively applied to the subtrees rooted at the selected child until a leaf is reached. A leaf is a singleton class that represents a previously observed object. While objects are predominantly incorporated with respect to existing classes, operators also exist for new node (class) creation, node combination (merging), and node division (splitting). A more complete description of COBWEB can be found in Fisher (1987b).

Object incorporation is easily adapted to allow object classification and flexible prediction: category utility guides an object along a path of nodes to a 'best[1] matching leaf. If any value(s) are missing from the new observation, they may be predicted from the known values of the leaf. While COBWEB trees are reminiscent of decision trees, probabilistic concepts are *polythetic* in that multiple attributes guide classification. If an object has missing attribute values then category utility acts as a partial-matching function with summation limited to probabilities of known attributes.

In our experiments a classification tree is constructed from a *training* set of objects. Each object of a distinct *testing* set is repeatedly classified with respect to the tree; in each repetition a different attribute is removed from the object description and must be predicted. Average prediction accuracy is computed over the test set for each attribute. In addition, prediction accuracy is tested for different size training sets. The result is a 'learning curve' for each attribute. Figure 2 shows the learning curve for three of the 37 attributes in a domain of soybean case histories (Stepp, 1984). As the curves illustrate, learning difficulty may vary considerably across attributes; some attributes (e.g., condition) are quickly and effectively learned (i.e., simply guessing the most frequent condition value would yield about 35% accuracy), some are invariant to training (e.g., occurrence-of-hail), but most are somewhere in between (e.g., damage-severity). COBWEB learns to predict all attributes (with variable success) using a single probabilistic concept tree. In contrast, ID3 and other learning from examples systems would have to be applied separately for each attribute. Fisher (1987a) compares the performance of one COBWEB tree with multiple, special-purpose ID3 trees.

This paper investigates the impact of two environmental factors on flexible prediction. Foremost among these is *noise:* the incorrect reporting of an attribute's value. Noise alters attribute correlations that all inductive systems require for effective learning. For example, if we randomly replace attribute values in the soybean domain with a probability of 25% (i.e., artificially introduce 25% noise) then prediction of severity-of-damage degrades to roughly 60%. A second influence on prediction accuracy is the extent of training, which also effects perceived statistical relationships between attributes. Our investigation motivates two strategies for noise-tolerant concep-

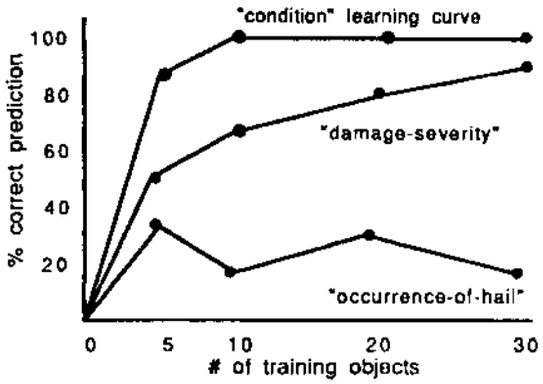

Figure 2: Learning curves for three attributes.

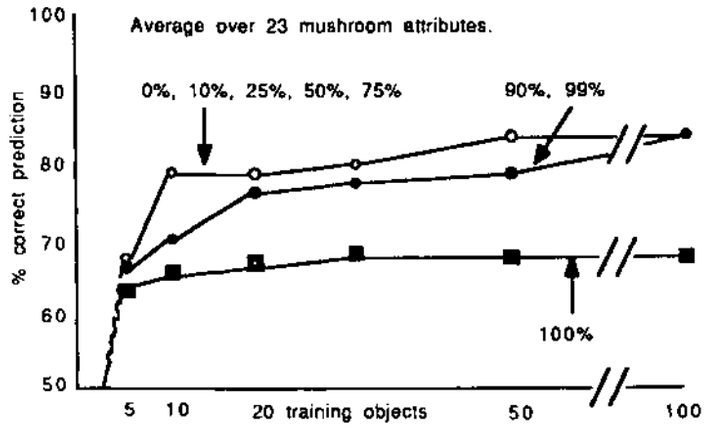

Figure 3: Learning curves (no noise).

tual clustering.

3 Chi-Square Preferences

Quinlan (1986) has extensively investigated the detrimental impact of noise on prediction in 1D3. Original versions of ID3 decomposed the training set to the point where all observations classified under a decision tree node were members of the same class. However, Quinlan (1986) demonstrates that this strategy tends to 'overfit/ the data in noisy domains: classification to a leaf may be guided by spurious, unjustified rules that do not benefit and actually detract from prediction accuracy. Several authors (Quinlan, 1986, in press; Breiman, Friedman, Olshen, & Stone, 1984) have explored methods for *pruning* unjustified rules so as to mitigate the effects of noise. A well-known technique is *chi-square pruning* (Quinlan, 1986). A decision tree node is decomposed only if it will lead to class distributions (at the new children) that are significantly different than the class distribution at the node to be divided. If the distributions do not differ significantly then the subtree is pruned; deeper classification will not benefit prediction.

The benefits of pruning in learning from examples are well-documented in noisy domains and motivate an exploration of the possible benefits to flexible prediction. COBWEB was modified to use a chi-square test of statistical significance for individual attributes. In particular, the distribution of an attribute's values at each node, N, are compared to the distribution of the node's children. If the distributions do not differ significantly by a user-specified confidence threshold (e.g., 90%) then N is taken as the prediction point for the attribute. Note that in flexible prediction we cannot simply prune a subtree based on the significance test for any one attribute, since this may not be an appropriate pruning point for other attributes. Rather, a strategy of less finality is to identify *preference* points based on the chi-square test that are maintained for each attribute. Attribute prediction requires that object classification proceed to an appropriate preference point, but no deeper. The most common attribute value at the preference point is the predicted value.

Experiments with the chi-square heuristic were run in three natural domains: the soybean disease case histories, a domain of poisonous and edible mushroom descriptions over 23 attributes, and voting records of 435 U.S. Representatives defined over 17 votes each. Prediction accuracy over a separate test set was checked at regular training intervals, for varying noise levels, and for 8 different chi-square confidence thresholds. Thresholds included 0% and 100%. In the of case 0% confidence any distribution difference is significant and using this threshold results in classification to a leaf, which is identical to previous COBWEB implementations. In contrast, 100% confidence can never be achieved and disallows classification beyond the root; the prediction is the most common value over the entire training set.

Figure 3 plots the 'averaged' attribute learning curves of the mushroom domain with *no* noise, but with different confidence thresholds. Low confidence appears to be the best strategy early in training since predictive patterns have been exposed, but have not reached statistical significance to the degree required by higher confidences. Chi-square preferences at high thresholds converge on equivalent accuracy (i.e., no significant differences in accuracy levels) later in learning (also see Fisher & Schlimmer, 1988).

The graph of Figure 4 examines the effect of noise. This graph shows averaged attribute accuracy levels after significant training at noise levels of 0%, 25%, and 50%). In general, the optimal threshold tends to increase with noise. Over all attributes there is a significant positive correlation between noise level and optimal confidence threshold (Pearson coefficient = 0.375, sample = 23 x 3 = 69). As noise increases, the deeper the classification (i.e., beginning at 0% confidence) the greater the overfitting. In noisy domains, higher chi-square confidences significantly increase accuracy.

A general trend in the data is that optimal confidence thresholds increase with noise and training, asymptoting at very high confidence thresholds (e.g., 99%). Early

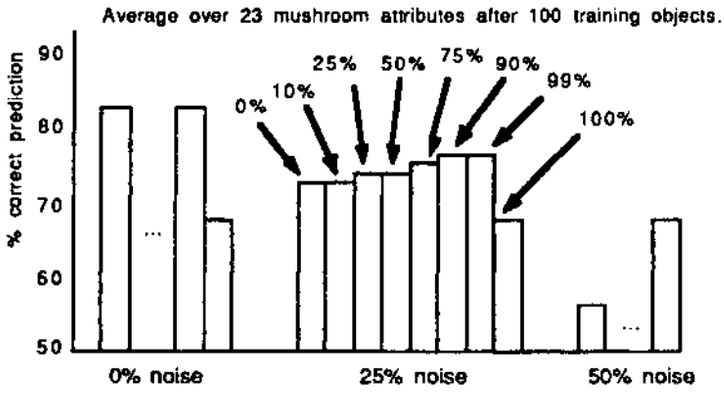

Figure 4: Asymptotic accuracy under noisy conditions.

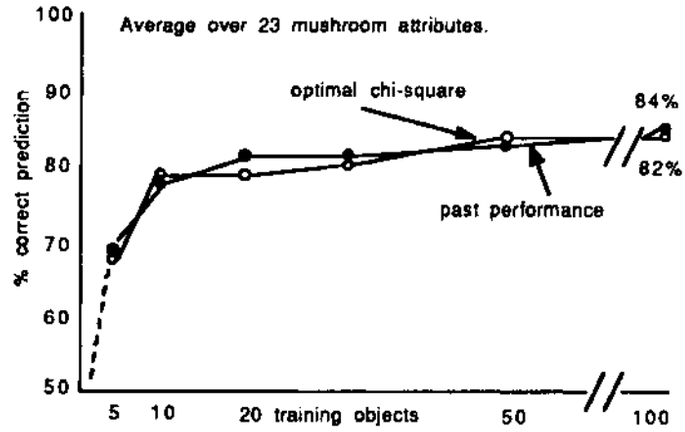

Figure 5: Learning curves (no noise).

in training and under noiseless conditions classification to a leaf is the optimal strategy. No chi-square confidence threshold will maximize performance in all (or most) learning scenarios. Thus, unless we can make *a priori* assumptions about the amount of noise and available training data, inconsistent performance will be a weakness of any constanti-threshold method. Section 4 introduces a simple, but effective means of preference identification that makes no assumptions about noise, training, or their interaction.

4 Past-Performance Preferences

A straight-for ward heuristic is that prediction of a missing attribute should occur at a node that historically has facilitated the greatest number of correct predictions. As a new training object is recursively classified, each of its attribute values is compared against the corresponding attribute values of the node; if the object's value equals that of the node's most common value, then the attribute's value would have been correctly predicted at the node. For each attribute and node, a count is maintained of the number of times the attribute was correctly predicted at the node (i.e., correct-at-node counts) during training. A count is also kept of the number of times that the attribute was correctly predicted at one of the node's descendents (i.e., correct-at-descendent). This latter count is updated as the recursive classification procedure unwinds; the correct prediction at a node is remembered and used to update the counts of its ancestors. When an object is added as a leaf we assume that it correctly predicts its own attribute values and there are no descendants (that correctly predict them). By convention, correct-at-node counts are initialized to 1 and correct-at-descendent counts are initialized to 0.

To predict the value of a missing attribute, classification proceeds until a node is reached that has historically outperformed its descendents in terms of predicting the missing attribute. At this point, the most common attribute value is forwarded as the correct answer. This method is similar in intent to Breiman. Friedman, Olshen, & Stone's (1984) *cost-complexity* pruning strategy and Quinlan's (in press) *reduced error* pruning. In these latter methods, a decision tree is fully constructed with a training set. The tree is then used to classify a separate test set. As each test item is classified a determination is made as to whether it would be correctly classified at each node (by the most common class at the node) on the path to a leaf. The tree is pruned at those nodes that maximized prediction over the test set. A separate test set is required because the original decision tree was engineered to fit the training set; cross-validation is necessary. In contrast, past performance is applicable to flexible prediction. A separate test set is not required, because the tradeoffs required to simultaneously improve prediction along many attributes introduces tradeoffs similar to those of training/cross-validation in ID3.

Experiments identical to those with chi-square were run. Figures 5 and 6 indicate that past-performance preferences roughly match the *optimal* chi-square thresholds averaged over various training and noise levels. Importantly, the optimal chi-square threshold differs between training and noise levels. Over each training/noise-level combination in the three (mushroom, soybean, congressional) domains (i.e., a total of 51 situations), past performance's mean accuracy is greater in significantly (x^2, a = 0.005) more situations than any single chi-square confidence level (i.e., ranging from 41/51 when compared to confidence level 0.00 to 49/51 for confidence level 0.50). These averaged results hide fluctuations among attributes, but conveniently and accurately reflect past performance's advantage.

The simplicity and effectiveness of past-performance preferences underscores an important principle: overt performance is the best (and simplest) 'model' of the complex interactions between noise levels, training, and other factors. Past performance makes no *a priori* assumptions about individual attribute characteristics, noise levels, or extent of training. Rather, attribute preferences are determined by the accumulated correct prediction counts for each node and its descendents.

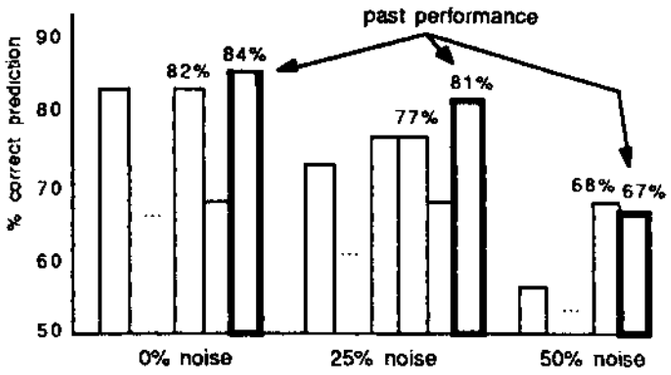

Figure 6: Asymptotic accuracy under noisy conditions.

5 Related and Future Research

Our studies of flexible prediction suggest further research in conceptual clustering and related areas.

5.1 Default Reasoning

Default values are often used to support uncertain reasoning, but Brachman (1985) points out that the automated reasoning literature makes no mention of prescriptive means for assigning default values. Not coincidentally, there has been little (if any) work on default value maintenance during learning. Our research provides a partial prescription for default identification.

Kolodner's (1983) CYRUS system offers a probabilistic interpretation of default values: *normative* values are true of a user-specified percentage (e.g., 67%) of class members. Normative (default) values at a top-level node (e.g., 'conservative' congressmen vote 'yes' on budget cuts) can be *shown otherwise* by classification to deeper levels (e.g., 'southern democrats' vote 'no' on budget cuts). Unfortunately, normative values require a user-supplied parameter and only delimit when a value is likely to be true; they fail to capture the more subtle notion of when it is best to predict that a value is true. Past-performance and chi-square preferences address both issues by demarcating locations where the *most common* value *should be* predicted. Nonetheless, important issues remain unaddressed.

Attribute preferences are *absolute* - based on the presumption that a certain amount (most) of 'evidence' is known. An important goal of future research is a process of default selection that is sensitive to the amount of available evidence (i.e., observed features). One idea is to cease classification when the difference in category utility scores between the best node and alternatives becomes sufficiently small: evidence is not sufficient to distinguish classification paths. This value would be the default value unless (and until) further evidence could be obtained. This classification procedure is more in line with traditional efforts in automated and default reasoning. Future work will seek to flesh out the relationship between incremental concept formation and prescriptive mechanisms for default identification and exploitation.

5.2 Case-Based Reasoning

Case-based reasoning has emerged as a subfield of automated reasoning and learning. The subfield is distinguished by its reliance on cases or object-level descriptions as a source of problem-solving information. The emergence of this subfield has the unfortunate effect of segregating research efforts that share fundamental specification and design principles, but differ (perhaps) only in implementation. Initial versions of COBWEB generated predictions from best matching leaves (previously observed objects). Thus, COBWEB is an efficient *implementation* of case-based reasoning, since generalized concepts help identify appropriate cases in logarithmic time versus linear time in the number of cases.

In addition to efficiency concerns, this paper illustrates the utility of case-based *and* abstraction-based inferencing. Reasoning at the case-level is most productive when few training observations are available and noise is not present (Ashley & Rissland, 1988), but this strategy overfits the data as training and noise increase. Past performance has the emergent effect of using cases very early in training - a case (observation) is initially added to a concept tree and is viewed as correctly predicting each of its attribute values. When very few observations have been seen these individual 'successes' will tend to out way the formative classes above. Gradually, solidifying attribute correlations at higher-level nodes become more reliable classifiers. Past performance's emergent strategy can be viewed as a probabilistic and conservative *specific to general* search for the optimal prediction level of each attribute.

An alternative to abstract ion-based reasoning is to retain carefully-selected cases only. Presumably, selective retention overcomes both problems of efficiency and accuracy that might otherwise hamper a nonselective case-based reasoner. Selective retention is employed by Bareiss *k* Porter (1987) and Aha *k* Kibler (1988) in their respective case-based learning from examples systems. Nonetheless, appropriate retention is difficult without the global guidance that abstracted knowledge can provide: Aha *k* Kibler report difficulties in dealing with irrelevant attributes and Bareiss *k* Porter use implicit sources of abstracted knowledge. Finally, while strict adherence to a case-based strategy is feasible in a learning from examples context, it is difficult to imagine an efficient implementation for flexible prediction, in which many prediction dimensions must be simultaneously coordinated.

6 Concluding Remarks

Experiments with COBWEB uncovered data overfitting that was mitigated by adapting pruning techniques for flexible prediction. However, the use of a constant-threshold method like chi-square often leads to non-optimal performance; we cannot predict *a 'priori* the amount of noise, training, or their interaction.

Limitations of chi-square preferences appear to be reduced by past performance: classification ceases at a point that has historically outperformed its descendents. Thus, we do not model (or impose a model on) noise, training, and other statistical interdependences beyond that which COBWEB is constructing through clustering. Rather, whatever these interdependencies, their effect is evident through prediction performance; nothing beyond past performance is used to guide classification. This principle is also implicit in COBWEB's use of category utility, a measure of the expectation of correct prediction afforded by classes. Class quality is tied directly to the task that will benefit from clustering. The explicit consideration of a performance task that improves with learning distinguishes COBWEB from almost all other clustering work and frees the system from user-supplied parameters or distributional assumptions of any kind.

Acknowledgements

I thank Jungsoon Yoo, Hua Yang, Ray Bareiss, and Gautam Biswas for informative discussions. Comments on an earlier draft by Dennis Kibler and John Gennari improved discussion and style considerably. Reviewers helped improve the clarity and correctness of the paper.

References

Aha, D. *k* Kibler, D. (1988). Detecting and removing noisy instances from concept descriptions. Technical Report 88-12, University of California, Irvine, CA.

Ashley, K. *k* Rissland, E. (1988). Waiting on weighting: a symbolic least commitment approach. *Proceedings of the Seventh National Conference on Artificial Intelligence* (pp. 239-244). St. Paul, MN: Morgan Kaufmann.

Bareiss, R., *k* Porter, B. (1987). PROTOS: An exemplar-based learning apprentice. *Proceedings of the Fourth International Machine Learning Workshop* (pp. 12-23). Irvine, CA: Morgan Kaufmann.

Brachman, R. J. (1985). I lied about the trees. *AI Magazine, 6,* 80-93.

Breiman, L., Friedman, J., Olshen, R., *k* Stone, C. (1984). *Classification and Regression Trees.* Belmont, CA: Wadsworth.

Cheeseman, P., Kelly, J., Self, M., Stutz, J., Taylor, W., *k* Freeman, D. (1988). AutoClass: A bayesian classification system. *Proceedings of the Fifth International Machine Learning Conference* (pp. 54-64). Ann Arbor, MI: Morgan Kaufmann.

Fisher, D. H. (1987a). Conceptual clustering, learning from examples, and inference. *Proceedings of the Fourth International Workshop on Machine Learning* (pp. 38 49). Irvine, CA: Morgan Kaufmann.

Fisher, D. H. (1987b). Knowledge acquisition via incremental conceptual clustering. *Machine Learning, 2,* 139-172.

Fisher, D. II. (1988). A computational account of basic level and typicality effects. *Proceedings of the Seventh National Conference on Artificial Intelligence* (pp. 233-238). St, Paul, MN: Morgan Kaufmann.

Fisher, D. H. & Schlimmer, J. (1988). Concept simplification and prediction accuracy. *Proceedings of the Fifth International Conference on Machine Learning* (pp. 22 28). Ann Arbor, MI: Morgan Kaufmann.

Gluck, M. A., *k* Corter, J. E. (1985). Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 283 287). Irvine, CA: Lawrence Erlbaum.

Kolodner, J. L. (1983). Reconstructive memory: A computer model. *Cognitive Science,* 7, 281 328.

Lebowitz, M. (1982). Correcting erroneous generalizations. *Cognition and Brain Theory, 5,* 367 381.

Michalski, R. S., *k* Stepp, R. (1983). Learning from observation: conceptual clustering. In R. S. Michalski, J. G. Carbonell, *k* T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach.* Los Altos, CA: Morgan Kaufmann.

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

Quinlan, J. R. (in press). Simplifying decision trees. *International Journal of Man-Machine Studies.*

Smith, E. E., *k* Medin, D. L. (1981). *Categories and concepts.* Cambridge, MA: Harvard University Press.

Stepp, R. (1984). *Conjunctive Conceptual Clustering: A Methodology and Experimentation.* Doctoral dissertation. University of Illinois, Urbana-Champaign, IL.