ICML-96 Workshop "Learning in context-sensitive domains" Bari, Italy.

Dynamically Adjusting Concepts to Accommodate Changing Contexts

Mark Devaney
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
markd@cc.gatech.edu

Ashwin Ram
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
ashwin@cc.gatech.edu

Abstract

In concept learning, objects in a domain are grouped together based on similarity as determined by the attributes used to describe them. Existing concept learners require that this set of attributes be known in advance and presented in entirety before learning begins. Additionally, most systems do not possess mechanisms for altering the attribute set after concepts have been learned. Consequently, a veridical attribute set relevant to the task for which the concepts are to be used must be supplied at the onset of learning, and in turn, the usefulness of the concepts is limited to the task for which the attributes were originally selected. In order to efficiently accommodate changing contexts, a concept learner must be able to alter the set of descriptors without discarding its prior knowledge of the domain. We introduce the notion of attribute-incrementation, the dynamic modification of the attribute set used to describe instances in a problem domain. We have implemented the capability in a concept learning system that has been evaluated along several dimensions using an existing concept formation system for comparison.

1 Introduction

Concept formation or concept learning is the process by which information is generalized into categories for efficient storage and communication and as a basis for further exploration in the environment (Gennari, 1990; Michalski, 1983; Shavlik & Dietterich, 1990). A concept learner takes one or more *instances* described along a number of *attributes* and produces a set of concepts or generalizations of the input. Each attribute has a corresponding *value*, and instances are typically viewed simply as a set of attribute-value pairs.

While the representation of concepts may vary from system to system, all concept learners form concepts for a common purpose: to facilitate the prediction of attributes or behavior of unseen instances in the environment (Anderson & Matessa, 1990, 1992; Fisher, 1987). This prediction usually occurs in pursuit of some goal or task; thus in order to produce a set of concepts which is useful in this *context*, the instances classified must be described by attributes which are *relevant* to the problem at hand. Furthermore, the context of a concept formation problem involves not only the goals and tasks of the learner but its background knowledge and domain theories as well as the external environment in which it operates. Context is an essential aspect of concept learning, for it determines which attributes to use for a given problem out of the infinitely many available, and consequently, as the context of a problem changes, there is a corresponding change in the set of attributes which are considered relevant. A concept learner that is to respond efficiently to changes in context must therefore be capable of easily altering the descriptors used in forming concepts but also of retaining and making use of information that it has already learned.

Existing concept learners are incapable of incorporating new attributes into their concepts and, instead, must completely reconstruct new concepts *ab initio*, without making use of their prior knowledge. For example, a concept formation system might produce the classes "pet" and "non-pet" based on training observations described by attributes such as "type of animal" and "tameness" but not "size". In order to address a slightly different context and produce the concept "small pet", this learner would have to construct the concept from scratch by reprocessing all of the training instances, this time including the descriptor "size". In response to this problem, we introduce the notion of *attribute-incrementation*, the dynamic

restructuring of an existing set of concepts as a result of a change in the attribute set used to describe the instances from which the concepts were formed. An attribute-incremental concept learner modifies its existing knowledge in response to a change in context instead of treating the new context as an entirely novel problem. In the above example, an attribute-incremental concept learner simply adds the new attribute of size to its existing concepts which are then restructured to form the concept of a small pet.

We present an implementation of an attribute-incremental concept formation system called AICC (Attribute-Incremental Concept Creator), based on Fisher's (1987) COBWEB. AICC is capable of both adding and removing attributes from an existing concept hierarchy and restructuring it accordingly. An evaluation of the system shows that the attribute-incremental approach achieves predictive accuracy and classification efficiency comparable to COBWEB and, in addition, in response to a change in the attribute set, AICC is able to produce a new set of concepts by modifying an existing one in substantially less time than it takes COBWEB to create a new set of concepts from scratch. While the benefits of the attribute-incremental approach are exclusively related to speed-up, the time requirements of the algorithm are significantly better than the application of standard concept learning algorithms such as COBWEB in many situations. Given sufficiently complex data sets, this speedup can reduce the time needed to create a set of concepts from several days to several hours, and as more complex problems are addressed and more complex algorithms introduced, these performance improvements become non-trivial.

2 Changing context

Human concept learners continuously face changes in context. Consequently, as concept formation systems are placed in situations more typical of those facing human concept learners, the issue of changing contexts becomes increasingly important. Concept learning is not a static process carried out in a rigidly-defined and unchanging environment by an unmotivated agent, but rather, a goal-driven, context-dependent process (Hadzikadic & Yun, 1988; Ram & Leake, 1995). The problem of concept formation must be viewed not only as a search for the correct set of concepts, but as a search for the correct description as well (Rendell, 1990). This description is precisely the set of attributes selected to describe the instances being classified.

Concepts are not formed in a vacuum — they are used in pursuit of some higher-level goal or task. As these goals and tasks change, different attributes of the objects become relevant Seifert (1989). Similarly, it has been shown that when constructing plans, people frequently add attributes to the set they are considering as the plan progresses as well as create *ad hoc* categories in response to novel goals that may arise (Barsalou, 1991). The domain theory of the agent also affects the set of attributes considered relevant (Pazzani, 1991; Wisniewski & Medin, 1991), and domain theories change as experience and knowledge are gained through interaction and experimentation in the environment. This is often what distinguishes expert and novice — the expert knows what "matters" while the poor domain theory of the novice does not rule out irrelevant attributes or fails to take into account attributes that are in fact useful (e.g., Beach, 1988; Boster & Johnson, 1989).

Context may be affected by changes in the external environment as well. Objects change over time, causing new properties to emerge and existing ones to disappear, and concepts that were created in the past may no longer be useful. A similar situation occurs when the environment itself does not change, but the perceptual capabilities of the agent do. This can occur as new techniques for observation or measurement are invented, or new insights are made into the causal relationships in the domain. For example, the invention of the microscope provided a wealth of "new" attributes with which to describe the world.

Context can also change as a result of the learning algorithms themselves. One such algorithm is the technique of *constructive induction* (Michalski, 1983), in which the learning system itself formulates new descriptors for use in describing its input. The introduction of new descriptors in this manner is traditionally thought of as a bias shift rather than a change in context, which is typically manifested as a change in attribute relevance rather than the introduction of entirely new attributes. However, in a broader view, context can be thought of as influencing or defining the set of descriptors used in a task rather than just the relative importance of these descriptors. Additionally, the introduction of new descriptors can be viewed as altering their relevance from zero to some non-zero value. For example, the FRINGE algorithm (Pagallo & Haussler, 1989) learns concepts by first constructing a decision tree for a set of training examples. Then, new features are generated through Boolean combinations of attributes that occur near the "fringe" of the tree. These features are added to the training examples and the decision tree is re-created from scratch. A more recent example of constructive induction is a system described by Hirsch & Japkowicz (1994) for predicting promotors in DNA which employs feature-generation operators to create new features; these are then added to the original descriptions in order to improve the quality of learning. This process occurs in

stages and requires a complete re-creation of the concept structure at each stage.

Existing inductive concept learners may be described as either *incremental* or *non-incremental* based on the way in which they process instances being classified. Incremental learners incorporate instances one-at-a-time and refine their concepts dynamically (Fisher & Langley, 1991), whereas non-incremental learners require that all instances to be classified be supplied at once before any concepts are formed. The incremental approach allows a system to efficiently incorporate new information from the environment and adjust its knowledge accordingly. However, instances are represented as collections of attributes, and while incremental learners can incorporate new instances they cannot handle changes in the composition of the attribute set used to describe these instances. These systems are more accurately described as *object-incremental*, to distinguish them from *attribute-incremental* concept learners which incorporate attributes rather than instances incrementally. The ability to alter instance *descriptions* dynamically allows attribute-incremental concept learners to respond efficiently to a wider variety of changes in context than their object-incremental counterparts. As concept learners are employed in tasks more typical of natural domains and "real-world" problems, this ability to respond to changing contexts becomes increasingly important.

3 Technical Details

AICC is an unsupervised concept learning system capable of adding and removing attributes from an existing concept hierarchy and restructuring it accordingly. AICC has been implemented as an extension to an existing concept learner, Fisher's (1987) COBWEB. COBWEB was chosen because its representation of concepts and the method used to construct them facilitated the addition of attribute-incremental capability. The use of an existing system as a foundation also reduced implementation-specific details, allowing the performance of each system to be more accurately evaluated and compared.

3.1 COBWEB

COBWEB is an inductive concept formation system which forms a concept hierarchy partially ordered by generality through incremental incorporation of training instances from the environment. Concepts within the hierarchy are represented probabilistically as the probability of the occurrence of each attribute value present in that concept. The root node of the hierarchy summarizes all instances classified by the hierarchy, and concepts become more specific as the tree is descended, with leaf nodes describing a single instance. The concept hierarchy is created dynamically as instances are incorporated through the use of simple operators and an evaluation function called *category utility* (Gluck & Corter, 1985). The major components of COBWEB are the category utility metric and the operators used to create and modify the concept hierarchy, described below.[1]

Table 1: AICC algorithm

FUNC AICC (Object, Root of a classification hierarchy)
1) Update instance descriptions
2) FOR each partition P beginning at the children of root
  A) DO (splits)
    i) let CU_0 = category utility of P
    ii) FOR each node n_i in P, let CU_i be the category utility of the partition resulting from splitting n_i
    iii) let CU_m be the maximum of all CU_i
    iv) if $CU_m > CU_0$, then split node n_m and let P = this new partition
  WHILE $CU_m > CU_0$

  B) DO (merges)
    i) let CU_0 = category utility of P
    ii) FOR each node n_i and n_j, $(i \neq j)$, in P, let CU_{ij} be the category utility of the partition resulting from merging nodes n_i and n_j.
    iii) Let CU_m be the maximum of all CU_{ij}
    iv) if $CU_m > CU_0$, then merge nodes n_i and n_j and let P = this new partition.
  WHILE $CU_m > CU_0$

3.2 AICC

AICC is an inductive concept formation system that incorporates *attributes* incrementally rather than *instances*. The input to AICC consists of a concept hierarchy produced by COBWEB and new descriptions of the instances used to create that hierarchy. These descriptions consist of new attributes to add to the original descriptions or existing attributes to remove from the original descriptions. In either case the input is in the same form as the original instance descriptions (attribute-value pairs). The algorithm for restructuring an existing hierarchy is shown in table ??.

3.3 Updating instance descriptions

The first stage in constructing a new concept hierarchy by modifying an existing one involves updating the descriptions of the instances used to construct the initial hierarchy. This consists of removing the counts of at-

[1]For a more extensive treatment of COBWEB, see Fisher (1987) or McKusick & Thompson (1990).

tributes that are no longer relevant and adding counts for the new attributes which are being added to the descriptions. The modifications are made in a single pass through the concept hierarchy in which each node is visited once. As each node is visited, the counts of attributes to be removed from the instance descriptions are simply eliminated. Adding new attributes to the instance descriptions is more complex: the appropriate attribute value counts to add depends on the instances classified at that node. To simplify this process, a minor modification was made to COBWEB so that each node in the concept hierarchy contains a list of identifiers of the instances classified at that node as well as attribute-value counts. The new instance descriptions contain these identifiers as well, allowing attributes to be added appropriately by matching the identifiers present in the node to those of the new descriptions. After each node has been visited and its attribute-value counts updated, the information in the concept hierarchy corresponds to the new attribute set used to describe the instances, but its structure must be changed to reflect this new state.

3.4 Hierarchy reorganization

After the alteration of the attribute set, it is unlikely that the original concept hierarchy is one that achieves maximal category utility. The reorganization of the hierarchy is a search for this new optimal point in the search space, beginning at the original hierarchy. In this portion of the process, AICC makes a single pass through the tree starting at the first partition (set of siblings) and continuing until all partitions have been visited. At each partition, the COBWEB operators *merge* and *split* are used to reorganize the nodes (concepts) in the partition. This process is performed in two stages, one for each operator.

The first stage of hierarchy reorganization employs the split operator, which replaces a node with its children, to determine whether any of the concepts in the partition should be specialized. The category utility of the partition resulting from splitting each node one-at-a-time while leaving the rest of the partition intact is calculated and compared to the initial category utility.[2] If splitting any of the nodes results in an increase in the category utility score of the partition, the one that results in the largest increase is made to the tree. Then, the process is repeated using the newly modified partition and its category utility score for comparison.

When no further improvements to the category utility can be made by splitting nodes, AICC generalizes concepts in the partition by applying the COBWEB merge operator, which combines two nodes in a partition. Each possible pair of nodes is considered for merging by comparing the resulting category utility with that of the partition produced by the previous stage. If any of these merges improves the category utility score, the one that results in the largest increase is actually applied. This process is repeated until the category utility of the partition can no longer be improved. Once each partition in the tree has been visited, the restructuring is complete and the resulting concept hierarchy is a conceptualization of the original instances described along the modified attribute set.

4 Evaluation

There are three major dimensions along which concept formation systems are typically evaluated. Since the primary purpose of concept learning is to facilitate prediction, the most important of these dimensions is the *predictive accuracy* of the concepts produced by the system. The second evaluative component is the efficiency of the system in constructing its concepts, or its *concept creation efficiency*. The third component, the *classification efficiency*, is the efficiency with which instances may be classified using a given set of concepts produced by the system. We have evaluated AICC along each of these dimensions and compared its performance to COBWEB. These results show that:

- AICC is able to construct its concepts in significantly less time than COBWEB.
- The concepts produced by AICC are equally good at prediction as those produced by COBWEB.
- Prediction using a concept hierarchy produced by AICC is as efficient as prediction using a COBWEB hierarchy.

The systems were evaluated using two data sets from the UCI Machine Learning Database (Merz & Murphy, 1996), the soybean small database originally used in Fisher's (1987) evaluation of COBWEB, and the lymphography database[3]. The soybean data consists of 47 instances described by 35 attributes having four classes based on diagnostic condition. The lymphography data consists of 148 instances described by 19 attributes having four classes based on diagnostic condition as well.

Our experimental methodology differs somewhat from that typically used in evaluating inductive concept learners. Traditionally, results are depicted as the performance of the dependent variable (e.g., predictive accuracy) as a function of the number of training *instances* used to construct the set of concepts. We

[2] The initial category utility is calculated based on the new attribute set; it is not the category utility of the partition before attributes were added and/or removed.

[3] Obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data.

are measuring the effectiveness of the AICC algorithm on adding additional *attributes* to an existing concept structure and not additional *instances*; therefore, the independent variable is the number of attributes used to construct the initial concept hierarchy and the number of training instances is constant. The total number of attributes available for each data set was randomly divided into 10 separate groups of (roughly) equal size and AICC was trained and evaluated a total of nine times, each time using an an additional group of attributes to construct the initial hierarchy.

In order to compensate for the effects of training set selection as well as the effect of initial attribute selection, two cross-validations were performed. For each data set, the training and testing sets were randomly selected five times and in each of these trials a subordinate ten-fold cross validation was conducted relative to attribute selection. The attribute validations were conducted by conducting 10 runs for each of the 9 values of initial training set size. For each run, 10 combinations of M subsets were randomly chosen ($1 \leq M \leq 9$) out of the 10 available subsets; AICC was trained on these subsets and the remaining ones were added using attribute-incrementation, allowing the effectiveness of the algorithm to be measured with increasingly smaller changes in descriptor size. The results for each of the 9 runs on increasingly larger initial descriptor sizes were obtained by averaging the performance of the algorithm over each of the 5 training and testing set variations and the subordinate attribute cross-validation, a total of 50 trials for each data point. For purposes of comparison, COBWEB was run using the 5 different training and testing sets but using all attributes and these results were averaged to supply a baseline for comparison.

4.1 Concept creation efficiency

The primary claim made of the attribute-incremental approach is that by utilizing existing knowledge in the form of a concept hierarchy, new information can be more efficiently incorporated. Thus, a learner is able to efficiently accommodate changes in the context of a problem, and the degree of this efficiency is determined by the amount of change which the learner faces. In order to quantify the amount of speedup that can be obtained using the attribute incremental approach, the amount of time (in seconds) required to construct a concept hierarchy was measured.[4] For AICC this measure consisted of two components, the time to construct the initial hierarchy using the reduced attribute set and the time to add the remaining attributes to this hierarchy in order to produce a set of concepts described by all available attributes.

[4]Experiments were run on a SUN Sparc10 workstation in single-user mode using local storage to eliminate load variances

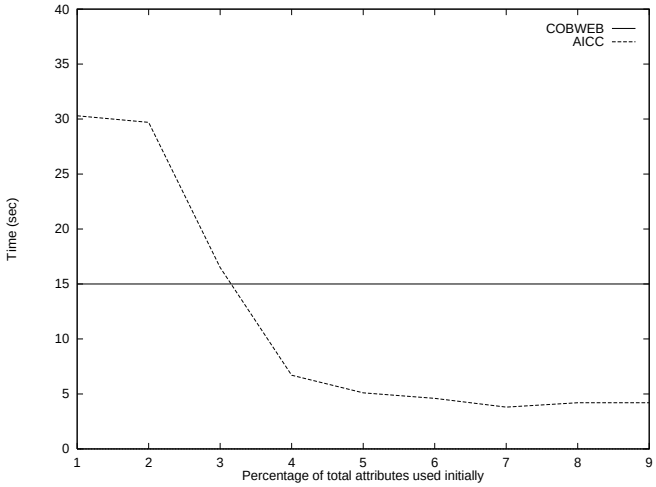

Figure 1: Hierarchy reorganization time - Soybean

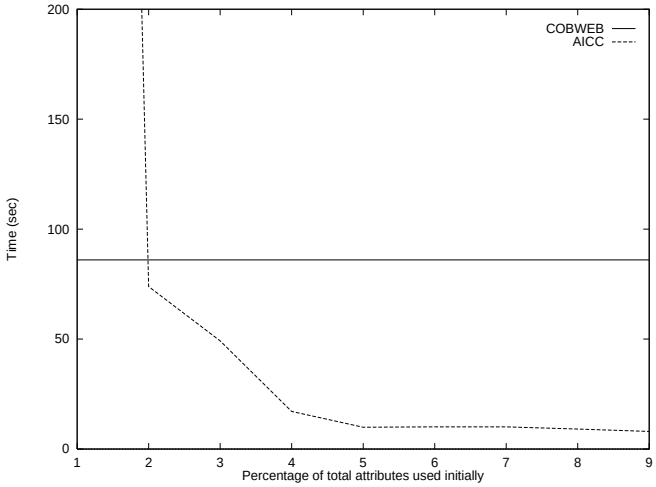

Figure 2: Hierarchy reorganization time - Lymphography

The true cost of incrementally adding attributes is the time to reorganize a a concept hierarchy given a new descriptor set as the cost of initially constructing the tree occurs only once and is thus amortized over time. Figures ?? and ?? depict the cost of reorganization alone along with the time required by COBWEB to construct an equivalent concept hierarchy. These results illustrate that the time to construct categories using the attribute incremental approach *decreases* as the size of the initial descriptor set increases whereas the time required by a traditional learning algorithm *increases* as descriptor size increases because it does not employ prior knowledge.

Figures ?? and ?? illustrate the total time for

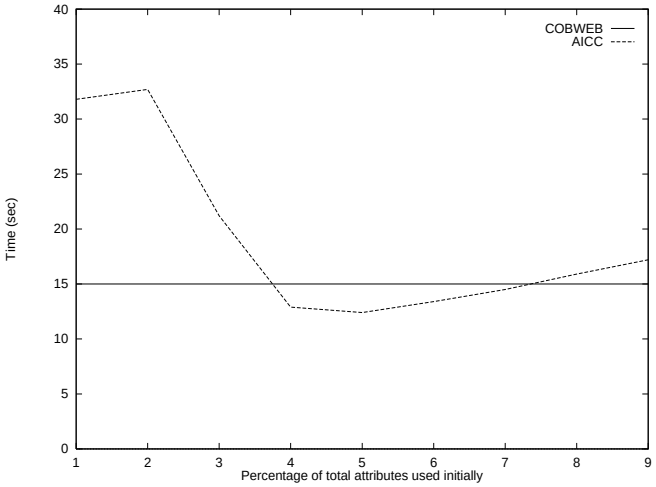

Figure 3: Total hierarchy creation time - Soybean

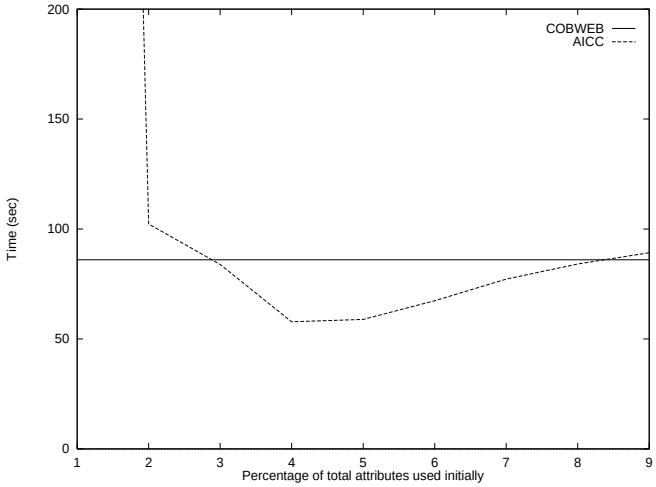

Figure 4: Total hierarchy creation time - Lymphography

each of the algorithms to construct a complete concept hierarchy for the two data sets.[5] These figures provide an indication of the cost of the attribute-incremental approach when the initial concept hierarchy must be constructed before the additional descriptors can be included. While this is not a scenario in which the attribute-incremental approach would be a good choice, these results show that even in this situation the performance of the algorithm is not that different than that of COBWEB, except when only relatively few attributes are used for the initial hierarchy and a large amount of reorganization must be performed.

4.2 Predictive Accuracy

The primary means of measuring the effectiveness of concepts produced by a learning algorithm is to employ them in the task of predicting the class membership or values of attributes of previously unseen instances. We measure predictive accuracy here by using a set of concepts to predict the values of all attributes for a given testing set. In each of the runs described above, after the concept hierarchy was constructed from the training instances, each of the testing instances was incorporated through the hierarchy to a leaf node without actually modifying the concept structure. This was repeated a times, where a is the number of attributes used to describe the instance. During each incorporation the attribute to be predicted was "masked off" so that it was not used during classification and the prediction was the corresponding attribute value at the leaf node to which the instance was classified. The accuracies of all a repetitions were averaged to supply a single predictive accuracy measure for each testing instance and these measures were averaged over all testing instances to supply a single score for each run. Again, the results of all 50 trials for each run combined to arrive at a single result for each of the initial attribute set sizes. Both COBWEB and AICC achieved nearly identical predictive accuracies for both data sets regardless of the number of attributes which had to be incorporated into an existing hierarchy. Table ?? shows the details and illustrates the ability of AICC to construct a useful concept hierarchy even when a great deal of concept restructuring must occur.

Algorithm	% init. atts.	*Accuracy*	
		Soy	Lymph
COBWEB	100	.9	.67
AICC	10	.9	.67
AICC	20	.9	.70
AICC	30	.9	.70
AICC	40	.9	.71
AICC	50	.9	.70
AICC	60	.9	.71
AICC	70	.9	.68
AICC	80	.9	.69
AICC	90	.9	.69

Table 2: Predictive Accuracy

4.3 Classification Efficiency

Classification efficiency refers to the ease with which a given concept hierarchy may be used to predict the features of new instances and is a rough indicator of the

[5] The AICC time is the sum of the time to construct the initial hierarchy and the time to incorporate the additional attributes.

Algorithm	% init. atts.	*Incorporations*	
		Soy	Lymph
COBWEB	100	8.09	11.69
AICC	10	8.61	11.61
AICC	20	8.72	11.62
AICC	30	8.6	11.36
AICC	40	8.5	11.77
AICC	50	8.4	11.70
AICC	60	8.47	11.83
AICC	70	8.35	11.12
AICC	80	8.44	11.53
AICC	90	8.37	11.93

Table 3: Classification efficiency

% init att	*Soy*		*Lymph*	
	Splits	Merges	Splits	Merges
10	8.5	27.10	15.2	108.5
20	11.8	22	34.3	71.2
30	11.5	20.7	34.8	55.6
40	8	15.6	35.4	42.4
50	6	12.9	28.8	32.9
60	5.5	11.9	24.7	31.5
70	4.2	10.2	24.5	33
80	2.8	10.2	18.1	25.7
90	2.4	10.7	14.4	23.8

Table 4: Amount of hierarchy reorganization - AICC

"quality" of a concept hierarchy. As each of the testing instances was incorporated through the hierarchy, the number of attempted incorporations of the node into the existing concepts was counted. This number is the sum of the size of all partitions visited from the root to the leaf node into which the testing instance was eventually incorporated. Again, the scores of the algorithms are nearly identical, suggesting again that they are producing concept hierarchies of similar structure regardless of the amount of reorganization which must be performed, as shown in table **??**.

Finally, in order to rule out the possibility that AICC is not performing much if any restructuring of the initial concept hierarchy and the evaluation metrics are simply largely insensitive to concept structure the number of splits and merges made to the initial hierarchy were measured and are shown in table **??**.

5 Conclusions & Related Work

Attribute-incrementation is an extension of the ability of concept learners to respond efficiently to changes in goals, tasks, background knowledge, and environment. Dynamically altering the attribute set used for a given problem has been addressed to some degree by Gennari's (1989) CLASSIT-2 which processes attributes incrementally in order of salience, eventually ignoring those that are not relevant to a given performance task. Similarly, the technique of constructive induction (Michalski, 1983) creates new attributes during the concept formation process by combining existing attributes in certain ways. AICC differs from these approaches in that it can incorporate entirely new attributes into an existing hierarchy as well as remove attributes to accommodate changes in the context of a problem while retaining its prior knowledge of the domain. Another area of machine learning research which greatly benefits from the ability to add or remove attributes efficiently is *feature selection* (Doak, 1992; Domingos, 1995), in which a learner explores the space of possible representations to find the one that most improves performance.

The ability to incrementally incorporate attributes from the environment is also important when there is a cost associated with the acquisition of information (Tan, 1993). This type of system illustrates the trend toward creating systems which must face real-world tasks and their corresponding constraints in the pursuit of a goal. Attribute-incremental concept formation provides such systems with a means of addressing these issues and allows them to respond to environmental constraints in a more dynamic and efficient manner. While the technique does not in itself *detect* and respond to changes in context, it allows a system that does to efficiently alter its conceptual representations to accommodate these changes.

The use of existing knowledge to facilitate performance in a new task has been a fairly popular area of research and has proven quite useful in improving the efficiency and accuracy of concept learners. Martin & Billman (1991) use prior knowledge of the variability of attributes to improve the performance of a concept learner, and Martin (1991) demonstrates the efficiency of transferring existing knowledge to a new domain. Thompson, Langley, & Iba (1991) have implemented a concept learner which uses knowledge of conjunctive sets of attributes, called components, to improve the creation of a concept hierarchy. AICC employs its current knowledge of a domain to adapt to a change in the context of a given problem rather than treating the new context as a novel problem. This use of prior knowledge has been shown to greatly increase the efficiency of the learning process without detracting from the ultimate performance of the learner.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments.

References

Anderson, J. R., & Matessa, M. (1990). A rational analysis of categorization. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 76-84). Austin, TX: Morgan Kaufmann.

Anderson, J. R., & Matessa, M. (1992) Explorations of an incremental, Bayesian algorithm for categorization. *Machine Learning*, 1, 275-308.

Barsalou, L. W. (1991). Deriving categories to achieve goals. In G. H. Bower (Ed.), *The psychology of learning and motivation: Advances in research and theory*, (Vol. 27). New York: Academic Press.

Beach, K. (1988). The role of external mnemonic symbols in acquiring an occupation. In M. M. Gruneberg, P. E. Morris, & R. N. Sykes (Eds.), *Practical aspects of memory: Current research and issues*, (Vol. 1). New York: Wiley.

Boster, J. S., & Johnson, J. C. (1989). Form or function: a comparison of expert and novice judgements of similarity among fish. *American Anthropologist*, 91, 866-889.

Doak, J. (1992). An evaluation of feature selection methods and their application to computer security (Tech. Rep. CSE-92-18). Davis: University of California.

Domingos, P. (1996). Context-sensitive feature selection or lazy learners. To appear in *Artificial Intelligence Review*.

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

Fisher, D. H., & Langley, P. (1991). The structure and formation of natural categories. In G. H. Bower (Ed.), *The psychology of learning and motivation: Advances in research and theory*, (Vol. 27). New York: Academic Press.

Gennari, J. H. (1989). Focused concept formation. *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 379-382). Ithaca, NY: Morgan Kaufmann.

Gennari, J. H. (1990). An experimental study of concept formation (Tech. Rep. 90-06). Irvine: University of California, Department of Information and Computer Science.

Gluck, M. A., & 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 Associates.

Hadzikadic, M. & Yun, D. Y. Y. (1988). Concept formation by goal-driven, context-dependent classification. *Methodologies for intelligent systems*, 3, 322-333.

Hirsh, H. & Japkowicz, N. (1994). Bootstrapping training-data representations for inductive learning: a case study in molecular biology. *Proceedings of the Twelfth International Conference on Artificial Intelligence* (pp. 639-644). Seattle, WA: MIT Press.

Martin, J. D. & Billman, D. O. (1991). Variability bias and category learning. *Proceedings of the Eighth International Machine Learning Workshop* (pp. 90-94). Evanston, IL: Morgan Kaufmann.

Martin, J. D. (1991). Transfer of predictive structure: Improving concept formation. Doctoral Dissertation, Department of Information and Computer Science, Georgia Institute of Technology, Atlanta.

McKusick, K. & Thompson, K. (1990). COBWEB/3: A portable implementation (Tech. Rep. FIA-90-6-18-2). Moffet Field, CA: NASA Ames Research Center.

Merz, C.J., & Murphy, P.M. (1996). UCI Repository of machine learning databases [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Michalski, R. (1983). A theory and methodology of inductive learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning: An artificial intelligence approach*. Palo Alto, CA: Morgan Kaufmann.

Pagallo, G., & Haussler, D. (1989). Two algorithms that learn DNF by discovering relevant features. *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 119-123). Ithaca, NY: Morgan Kaufmann.

Pazzani, M. J. (1991). Influence of prior knowledge on concept acquisition: Experimental and computational results. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 17, 416-432.

Ram, A. & Leak, D. (1995). Learning, goals, and learning-goals. In A. Ram & D. B. Leake (Eds.), *Goal-driven learning*. Cambridge, MA: MIT Press/Bradford Books.

Rendel, L. (1990). Some issues in concept learning. In J. W. Shavlik & T. G. Dietterich (Eds.), *Machine learning : An artificial intelligence approach* (Vol. 3). San Mateo, CA: Morgan Kaufmann.

Seifert, C. M. (1989). A retrieval model using feature selection. *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 52-54). Ithaca, NY: Morgan Kaufmann.

Shavlik, J. W., & Dietterich, T. G. (1990). *Readings in machine learning.* San Mateo, CA: Morgan Kaufmann.

Tan, M. (1993). Cost-sensitive learning of classification knowledge and its applications in robotics. *Machine Learning*, 13, 7-33.

Thompson, K., Langley, P., & Iba, W. (1991). Using background knowledge in concept formation. *Proceedings of the Eighth International Machine Learning Workshop* (pp. 554-558). Evanston, IL: Morgan Kaufmann.

Wisniewski, E. J. & Medin, D. L. (1991). Harpoons and long sticks: The interaction of theory and similarity in rule induction. In D. H. Fisher, M. J. Pazzani, & P. Langley (Eds.), *Concept formation: Knowledge and experience in unsupervised learning.* San Mateo, CA: Morgan Kaufmann.