Machine Learning: Proceedings of the Fourteenth International Conference, Nashville, TN, July 1997 (to appear).

Efficient Feature Selection in Conceptual Clustering

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

Feature selection has proven to be a valuable technique in supervised learning for improving predictive accuracy while reducing the number of attributes considered in a task. We investigate the potential for similar benefits in an unsupervised learning task, conceptual clustering. The issues raised in feature selection by the absence of class labels are discussed and an implementation of a sequential feature selection algorithm based on an existing conceptual clustering system is described. Additionally, we present a second implementation which employs a technique for improving the efficiency of the search for an optimal description and compare the performance of both algorithms.

1 Introduction

The choice of which attributes to use in describing a given input has crucial impact on the classes induced by a learner. For this reason, the majority of real-world data sets used in inductive learning research have been constructed by domain experts and contain only those attributes which are expected to be relevant to the classification problem. However, there are areas (e.g., the cloud data described in Aha & Bankert, 1995, 1994) in which sufficient domain knowledge to select only relevant attributes does not exist, and in such cases the data must be described by all attributes that are considered potentially relevant.

Unfortunately, the presence of more information in the form of additional descriptors does not usually generate a corresponding increase in performance of the classifier. Because an inductive learner must treat all attributes as equally important, the presence of irrelevant or misleading information will tend to *decrease* the effectiveness of the learning algorithms. *Feature selection* algorithms address these concerns by reducing the number of attributes used to describe the training population in order to improve the quality of the concepts produced. These algorithms search the space of possible input descriptions and select the one that performs best according to some evaluation criteria. There have recently been several applications of feature selection to supervised concept learning systems which have produced promising results on complex real-world data sets, e.g., Aha & Bankert (1995, 1994), Doak (1992).

We describe a preliminary investigation into the viability of feature selection techniques in the paradigm of *unsupervised* concept learning, where the absence of class labels compounds the difficulty of the problem, but the added complexity of the learning algorithms makes it an ideal candidate for reducing the size of feature sets used in learning. Our first implementation is based on Fisher's (1987) COBWEB, an unsupervised concept learning system. Additionally, we present an implementation which employs a technique we call *attribute-incrementation* to improve the efficiency of the feature selection process. We evaluate both systems in terms of the potential to improve predictive accuracy by reducing descriptor size and the attribute-incremental system by its potential to reduce the time required to arrive at the reduced description without sacrificing improvement in predictive accuracy.

2 Feature selection

Feature selection can be viewed as a search through the space of potential descriptions of the training data. The process involves an algorithm to control the exploration of the space of potential subsets, an evaluation function to judge the quality of each of these subsets according to some metric, and the ultimate performance function with which the learner is evaluated (Aha & Bankert, 1995). Since the space of all feature subsets of a attributes has size 2^a, feature selection algorithms typically use some form of non-exhaustive search. One of the more popular techniques is a hill-climbing search known as *sequential selection* which may be either *forward (FSS)* or *backward (BSS)*.

In forward sequential selection, the learner begins with an empty descriptor set and evaluates the effect of adding each

attribute one at a time. The attribute that results in the best performance as measured by the evaluation function is added to the description and the process repeats until no further improvement can be made. Similarly, backward sequential selection begins with the full descriptor set and repeatedly removes a single attribute until no improvements in performance can be made. The complexity of sequential algorithms is O(a^2) and in data sets with large attribute sets even sequential selection algorithms can require a great deal of processing time (Doak, 1992).

The search through description space is guided by an evaluation function which measures the "quality" of each attribute subset. John, Kohavi, and Pfleger (1994) distinguish two types of evaluation functions, "filter" models and "wrapper" models. Wrapper models employ feedback from the performance function (Kohavi & John, 1997), typically the classifier itself; measuring the performance of a feature subset by its classification accuracy on an internal "testing" set or by cross-validation on the training data. Filter models employ some measure of an *intrinsic* property of the data (Doak, 1992) that is presumed to affect the ultimate performance of the classifier but is not a direct measure of this performance. In supervised learning, wrapper models typically measure subset quality by evaluating predictive accuracy of the class labels. Similarly, while filter models do not explicitly measure accuracy of predicting the class label they evaluate subsets on their ability to "determine" the class label. The algorithms FOCUS (Almuallim & Dietterich, 1991) and Relief (Kira & Rendell, 1992) are examples of such methods.

The ultimate performance function in supervised concept learning is typically the accuracy of the learner in predicting the class labels of a previously-unseen set of testing instances. Other traditional evaluative metrics measure the efficiency with which the concepts are learned or some structural property of the concepts, for example.

3 Unsupervised feature selection

There is some evidence from supervised feature selection research that wrapper models outperform filter models (e.g., John, Kohavi, & Pfleger 1994; Aha, & Bankert, 1995), presumably because the induction algorithm itself is being used to evaluate feature subsets and thus the same bias exists in both the evaluation function and the performance function. However, because class labels are not present in unsupervised learning (although they usually exist in the data and are often used to evaluate the final output of the system), it is not possible to use any of the typical wrapper or feature model approaches.

In the absence of class labels, predictive accuracy in conceptual clustering is usually measured by the average accuracy of predicting the values of each of the descriptors present in the testing data. One approach to feature selection in a conceptual clustering system, then, is to employ a wrapper approach with average predictive accuracy over all attributes replacing the predictive accuracy of class labels.

However, another evaluation function is suggested by the techniques used in the clustering algorithms themselves. Because unsupervised concept learning systems construct the classes as well as the rules to assign instances to these classes, they already employ an evaluation function to guide the process of creating concepts. These functions supply a quality measure given a set of concepts using only intrinsic properties of the data and are thus well-suited for use as an evaluation function to guide the feature-selection selection search.

We have taken the latter approach, employing an evaluation function called *category utility*, based on research by Gluck and Corter (1985) in human category formation. An interesting aspect of this evaluation function is that it blurs the traditional wrapper/filter model distinction - it is like a wrapper model in that the underlying learning algorithm is being used to guide the descriptor search but it is like a filter model in that the evaluation function measures an intrinsic property of the data rather than some type of predictive accuracy. Category utility has been used by a number of conceptual clustering systems; we have employed one such system, Fisher's (1987) COBWEB, as our underlying concept learner in a feature selection task.

4 Category utility and COBWEB

COBWEB is a conceptual clustering system which represents concepts probabilistically in a hierarchical tree structure. Each set of siblings in the hierarchy is referred to as a *partition*, and the category utility metric for a partition is calculated as:

$$\frac{\sum_{k=1}^{n} P(C_k)[\sum_i \sum_j P(A_i = V_{ij}|C_k)^2 - \sum_i \sum_j P(A_i = V_{ij})^2]}{n}$$

The C_k terms are the concepts in the partition, A_i is each attribute, and V_{ij} is each of the possible values for attribute A_i. This equation yields a measure of the increase in the number of attribute values that can be predicted given a set of concepts, $C_1 \ldots C_k$, over the number of attribute values that could be predicted without using any concepts. The term $\sum_i \sum_j P(A_i = V_{ij})$ is the probability of each attribute value independent of class membership and is obtained from the parent of the partition. The $P(C_k)$ term weights the values for each concept according to its size, and the division by *n*, the number of concepts in the partition, allows comparison of partitions of different sizes.

This evaluation function is only useful for symbolic attributes, and in order to evaluate our algorithm on a data set containing continuous variables we use Gennari's (1990) CLASSIT algorithm and replace the innermost summations

of the above category utility equation with:

$$\frac{\sum_k^n P(C_k)\frac{1}{\sigma_{ik}} - \frac{1}{\sigma_{pi}}}{n}$$

where K is the number of classes in the partition, σ_{ij} the standard deviation of attribute *i* in class *k* and σ_{pi} the standard deviation of attribute *i* at the parent node.

COBWEB constructs its concept hierarchy incrementally through the use of four operators. As each training instance is incorporated, it is recursively processed through the tree and one or more of the operators is applied as determined by the category utility metric. Briefly, at each partition in the hierarchy, the COBWEB algorithm considers adding the instance to an existing concept using the *incorporate* operator or applying the *create* operator to construct a new concept containing the instance at that partition. The choice is determined by which action results in a better category utility score for the partition and the process repeats until a leaf node is reached or the *create* operator is applied. Before recursing, however, the algorithm compensates for ordering effects in the input by considering the application of one of two additional operators, *merge* and *split*. The *merge* operator attempts to combine the two nodes which were identified as the best host for the new training instance into a single concept, effectively generalizing the concepts in the partition. The *split* operator performs the inverse operation of attempting to replace the best host with its children, specializing that concept. If either of these operations would result in an increased category utility score for the partition, the one that best does so is applied and recursion continues at the next partition.

Our implementation of an unsupervised feature selection algorithm generates a set of feature subsets in either the forward or backward direction from the training data, runs COBWEB using each of these subsets and measures the category utility of the first partition (children of the root) of the resulting concept hierarchy, retaining the highest score and the description which produced it. The process is repeated by generating the next larger (or smaller) subset using the current best description and terminates when no higher category utility score can be obtained.

5 Improving search efficiency

Feature selection has the potential for very high computational complexity because of the large number of attribute subsets that may be evaluated. This is particularly true in COBWEB due to the expense of repeatedly computing the category utility metric. One immediate observation about the feature selection process described above is that after an attribute is selected for addition or removal, the concept structure must be reconstructed from scratch using the new descriptor set. That is, COBWEB generates a set of concepts using *n+1* (or *n-1*) descriptors without making use of the existing *n*-descriptor hierarchy.

In an attempt to improve the efficiency of the feature subset search, we employ a technique we refer to as *attribute-incremental* concept formation. An attribute-incremental learner adds (or removes) *attributes* to an existing concept structure rather than *instances*. Our previous research (Devaney & Ram, 1996) has suggested that it is usually much more efficient to retain and modify an existing concept structure when making relatively minor modifications to the data that comprise it rather than throwing it away and beginning anew from scratch as COBWEB is forced to do. The sequential feature selection problem is an ideal application of this technique, as a large number of small changes in descriptor sets are made. We have implemented an attribute-incremental concept learner based on COBWEB, referred to as AICC (Attribute-Incremental Concept Creator) and have created an additional feature selection implementation using AICC as the underlying concept learner.

The AICC algorithm takes as input a concept hierarchy and a new *attribute* to add or remove, along with the values of this attribute for each of the training instances. In a single pass through the hierarchy, each partition is visited recursively and modified in light of the change in descriptors of the training data. This modification is performed in several stages. In the first stage, the set of descriptors of each concept node at the current partition is modified to reflect the changed attribute set. Then, the category utility score is recalculated to reflect this change.

At this point, the algorithm tries restructuring the hierarchy in an attempt to improve its category utility score. This is done through the use of the operators *merge* and *split* used by COBWEB, but in a different manner. First, the split operator is repeatedly applied in the given partition by attempting to split each of the nodes in the partition. If any of these splits results in a better category utility score it is performed and the process repeats on the new partition until no further improvement can be made. Then, the algorithm measures the effect of merging each possible pair of nodes in the partition and, again, performs the best one and repeats until no further improvement in the category utility of the partition can be made. At this point, the algorithm continues at the next partition and terminates when all partitions have been visited. The AICC algorithm is shown in table 1.

Essentially, AICC is performing the same sort of hill-climbing search through the space of possible concept representations as COBWEB, but instead of beginning from scratch each time the descriptor set changes, AICC begins at the previous point in the space. When small changes are made in the representation, as occurs in a feature selection task, the new set of concepts lies close to the prior one and AICC is able to arrive at this new point much more quickly by traveling less far through the search space.

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 $CU_m > CU_0$

6 Evaluation

The two goals of this initial investigation were to verify the hypotheses that the predictive accuracy of concepts induced by COBWEB could be improved by employing a feature selection process to restrict the set of descriptors considered and that significant performance gains in terms of search time could be made by employing the attribute-incremental approach without sacrificing this improved predictive accuracy. To this end we compare three concept formation systems: COBWEB without feature selection (i.e., using the entire attribute set), COBWEB with feature selection, and AICC with feature selection. As an additional interesting baseline we also compare with COBWEB (no feature selection) but using only the attributes defined as relevant.

Evaluations were conducted using one real and one artificial data set, both from the UCI Machine Learning Database (Merz & Murphy, 1996). The real data was the Cleveland Clinic Heart-Disease database[1] and the artificial data was based on the LED data set generator.

The heart disease data contains a total of 76 attributes including one class label indicating the presence of heart disease on an integer scale of 0 (none) to four. The data set is interesting in the context of feature selection research because most of the attributes are considered irrelevant and the majority of published experiments use only a subset of fourteen attributes (Aha, 1988). The LED data contains 24 attributes, seven (binary) attributes which predict one of ten classes, and seventeen attributes which have random values.

[1]Donated by V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

Algorithm	Subset size	Accuracy
COBWEB (relevant)	13 (.00)	.755 (.146)
COBWEB (all)	75 (.00)	.567 (.184)
COBWEB-FS	18.00 (8.48)	.677 (.123)
AICC-FS	16.33 (10.61)	.733 (.159)
COBWEB-BS	20.67 (9.09)	.813 (.161)
AICC-BS	19.33 (2.94)	.784 (.135)

Table 2: Subset size and accuracy : Heart Disease

Algorithm	Subset size	Accuracy
COBWEB (relevant)	7 (.00)	1.00 (.00)
COBWEB (all)	24 (.00)	.77 (.157)
COBWEB-FS	6.67 (.816)	.93 (.179)
AICC-FS	7 (.00)	1.00 (.00)
COBWEB-BS	12 (14.35)	.88 (.170)
AICC-BS	8 (.00)	.89 (.043)

Table 3: Subset size and accuracy : LED

For each data set, a three-way cross-validation was performed by randomly selecting three training sets of 100 instances and three testing sets of 50 instances from the total number of available instances for the heart disease data, and by generating three training sets of 100 instances and three testing sets of 50 instances using different random seeds for the LED data. Each of the algorithms was run on the training data with the class label attribute "masked out" so that it was not used during concept learning, only for final evaluation.

As a baseline for comparison, the standard (non feature selection) version of COBWEB was run on the training sets, once using all available attributes and once using only the relevant attributes. The two feature selection algorithms, referred to as COBWEB-FS and AICC-FS were evaluated in both a forward and backward feature selection context.

After training, the concepts produced by each of the algorithms were evaluated by measuring the accuracy of predicting the class label of the previously-unseen testing instances. As in evaluations conducted with the heart disease database by other researchers (Aha, 1988), accuracy was measured by treating the output of the classifier as a binary indicator of the presence of heart disease (values of 1-4 were treated as identical). The performance task in the LED domain was to predict the class value (1-10). As shown in tables 2 and 3, all of the feature selection algorithms arrived at significantly reduced descriptor sets and a corresponding increase in predictive accuracy.

Another focus of this research is concerned with improving the computational efficiency of the feature selection

Algorithm	Subsets tried	Time (seconds)
COBWEB (relevant)	1 (.00)	203.33 (15.79)
COBWEB (all)	1 (.00)	923.33 (120.27)
COBWEB-FS	577.33 (251.48)	31560.67 (16048.84)
AICC-FS	659.33 (276.11)	5902.67 (4051.12)
COBWEB-BS	1064 (206.12)	280890 (67092.07)
AICC-BS	1128.33 (28.57)	53521 (44376.88)

Table 4: Subsets evaluated and total time : Heart disease

Algorithm	Subsets tried	Time (seconds)
COBWEB (relevant)	1 (.00)	66.33 (5.71)
COBWEB (all)	1 (.00)	224 (11.31)
COBWEB-FS	158.33 (13.88)	11451 (1936)
AICC-FS	164.00 (.00)	5322 (1256)
COBWEB-BS	204.67 (191.32)	50482 (8942)
AICC-BS	280.00 (.00)	14812 (10995)

Table 5: Subsets evaluated and total time : LED

process. The attribute-incremental approach of the AICC algorithm which efficiently adds and removes attributes is particularly useful in the task of feature selection. As seen above, the algorithm provides performance similar to COBWEB in both forward and backward feature selection. In order to measure the efficiency which can be gained by using this approach we also measured the amount of time in CPU seconds[2] each of the feature selection algorithms required to arrive at a final attribute subset as well as the number of subsets that were evaluated during the search. These results are shown in tables 4 and 5 and illustrate the dramatic speedup that can be obtained by using the attribute-incremental approach in sequential feature selection.

7 Conclusions and future research

The research described here is a preliminary investigation into the applicability of feature selection techniques in unsupervised concept learning. These techniques have proven valuable in supervised concept learning and although the domain of unsupervised concept learning presents a number of additional complexities, similar positive results have been demonstrated, suggesting that further research is in order.

In particular, the use of feature selection greatly reduced descriptor size while improving performance with respect to classification accuracy. This ability is crucial in domains with a large number of available attributes that are not all necessarily relevant to a particular classification task. It has also been shown that focusing on relevant attributes also increases the efficiency of a classifier by reducing the number of attributes considered during classification (Gennari, 1991). We intend to further research these claims using more real and artificial data sets which exhibit the traits feature selection is designed to take advantage of.

Another area explored in this research is the use of the paradigm of attribute-incremental concept formation to improve the performance of the feature selection process. Experimental evidence suggests that this technique greatly reduces the time required to search the space of potential descriptors without sacrificing the ultimate performance of the concepts. As more complex data is explored, techniques for improving search performance such as attribute-incrementation or caching (Caruana & Freitag, 1994) become increasingly important.

Beyond further empirical evaluations, areas of future research include exploring non-sequential search algorithms incorporating some combination of AICC and COBWEB to optimize the efficiency of performing the search while improving the accuracy of the concepts obtained, as well as investigating the potential for applying the attribute-incremental approach to other conceptual clustering and supervised concept learning algorithms.

Acknowledgements

The authors wish to thank Doug Fisher for initially suggesting the topic, David Aha for suggesting the heart disease database, and the anonymous reviewers for their helpful feedback.

References

Aha, D. (1988). Text file "heart-disease.names", location: `ftp.ics.uci.edu/pub/machine-learning-databases/heart-disease/`.

Aha, D., & Bankert, R. (1994). Feature selection for case-based classification of cloud types: An empirical comparison. In D.W. Aha (Ed.) *Case-based reasoning: Papers from the 1994 Workshop*. Menlo Park, CA: AAAI Press.

Aha, D., & Bankert, R. (1995). A comparative evaluation of sequential feature selection algorithms. *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, Ft. Lauderdale, FL: Unpublished.

Almuallim, H., & Dietterich, T.G. (1991). Learning with many irrelevant features. *Ninth National Conference on Artificial Intelligence* (pp. 547-552). MIT Press.

Caruana, R., & Freitag, D. (1994). Greedy attribute selection. *Machine Learning : Proceedings of the Eleventh International Conference*, San Francisco, CA: Morgan Kaufmann.

Devaney, M. & Ram, A. (1996). Dynamically adjusting

[2]Running in single-user mode on a SUN Sparc10 workstation

concepts to accommodate changing contexts. In M. Kubat & G. Widmer, Eds., *Proceedings of the Workshop on Learning in Context-Sensitive Domains*.

Doak, J. (1992). An Evaluation of Feature Selection Methods and Their Application to Computer Security (Tech. Rep. CSE-92-18). Davis: University of California.

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

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

Gennari, J. H. (1991). Concept formation and attention. *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 724-728). Irvine, CA: Lawrence Erlbaum Associates.

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.

John, G.H., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. *Machine Learning : Proceedings of the Eleventh International Conference*, San Francisco, CA: Morgan Kaufmann.

Kira, K., & Rendell, L. (1992). The feature selection problem: Traditional methods and a new algorithm. *Tenth National Conference on Artificial Intelligence* (pp. 129-134). MIT Press.

Kohavi, R. & John, G. H. (1997), Wrappers for Feature Subset Selection. *Artificial Intelligence Journal*. Forthcoming.

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.