In: Proc. 2nd European Machine Learning Conference (EWSL-87), pp11-30,
Eds: I. Bratko and N. Lavrac. Wilmslow,UK:Sigma (1987)
http://www.cs.utexas.edu/users/pclark/papers

Induction in Noisy Domains

Peter Clark
Tim Niblett

The Turing Institute
Glasgow, G1 2AD

ABSTRACT

This paper examines the induction of classification rules from examples using real-world data. Real-world data is almost always characterized by two features, which are important for the design of an induction algorithm. Firstly, there is often noise present, for example, due to imperfect measuring equipment used to collect the data. Secondly the description language is often incomplete, such that examples with identical descriptions in the language will not always be members of the same class.

Many induction systems make the 'noiseless domain' assumption that the examples do not contain errors and the description language is complete, and consequently constrain their search for rules to those for which no counter-examples exist in the data used for induction. However, in real-world domains correlations between attributes and classes in a data set are rarely without exceptions. To locate such correlations and induce rules describing them it is also necessary to consider rules which may not classify all the training examples correctly.

This paper firstly discusses some of the problems presented by noise and proposes a top-down induction algorithm for induction in real-world domains. Secondly, an experimental comparison of this algorithm with other induction systems is presented using three sets of real-world medical data.

October 20, 1994

Induction in Noisy Domains

Peter Clark
Tim Niblett

The Turing Institute
Glasgow, G1 2AD

1. Introduction

Automatic rule induction systems for inducing classification rules have already proved valuable as tools for assisting in the task of knowledge acquisition for expert systems. In particular, two families of systems based on the ID3 and AQ algorithms have been especially successful. ID3 has been successfully applied to the classification of chess endgames (for example [1], [2], [3]) which were intractable to human experts because of the volume of data, and C4 (an ID3 descendant) to the diagnosis of thyroid diseases [4]. Systems based on the AQ algorithm, such as AQ11 [5] and GEM [6], have been successful in the fields of soya bean diagnosis ([6], [7]) and chess [6].

We consider the principle task of a real-world induction system to be assisting the expert in expressing his or her expertise. Consequently, we require that the induced rules are highly predictive and are easily comprehensible to the expert. In particular, we wish the induction algorithm to be tolerant of noise, which is almost always present in real-world problems.

Many induction systems make a 'noiseless domain' assumption that any genuine regularity in the data will be perfect (ie. without counter-examples). Consequently, during the search for useful generalizations, such systems explore only the space of rules which are completely consistent with the training examples. By searching for the most general, consistent rules, these systems locate regularities which involve large numbers of training examples and hence are statistically unlikely to be due to chance choice of the training examples. Instead, they reflect genuine correlations between attributes and classes in the domain and consequently perform well.

However, in most real-world domains major correlations are rarely perfect and instead often have a few counter-examples in the training data. Such counter-examples arise due to the presence of noise and an inadequate description language for representing the domain. Searching only the space of consistent rules does not find such regularities and instead more specific rules based on only a few training examples tend to be selected. Although these rules perform perfectly on the training examples, their predictive accuracy on future test examples is often lower because rules formed on the basis of small numbers of examples are susceptible to noise. The small trends they reflect are more likely to be due to chance choice of training set compared with other rules found reflecting major correlations, but rejected due to the presence of counter-examples in the data. As a consequence, the rule set is both large and not of the highest predictive power.

For an induction system to induce highly predictive rules in domains containing noise the search space must be expanded to include rules for which counter-examples exist and the evaluation of rules appropriately modified to enable the most predictive rules to be located. This paper

describes an algorithm in which these changes are implemented. The principle mechanism for avoiding the selection of rules, which are specific, completely consistent but poorly predicting, is the use of a statistical significance test. Rules which are found insignificant are deemed likely to be due to chance choice of the training examples and are, consequently, discarded.

Firstly, the problems presented by noise are discussed and the algorithm CN2 for inducing production-like rules is described. Secondly, in sections 3 and 4, we present experimental comparison of this system's performance in 3 medical domains with 3 other induction systems.

2. Induction in Noisy Domains

2.1. The Problem of Noise

In any set of training data serving as input to a learning algorithm there may be correlations between the attributes used to describe examples and the classes into which the examples are catagorized. The strength of these correlations will vary depending on the numbers of examples supporting and opposing the observed relationship. Some of these correlations will reflect some genuine property of the domain in which there is either a causal relationship between the attributes and class (ie. the attributes cause the class or vice versa) or an associative relationship (ie. the attributes and class are due to the same common cause). Others will be due to chance, caused by the particular selection of training examples presented to the system, and have no predictive power. Strong correlations usually involve larger numbers of training examples and are, thus, less likely to be due to chance as the effects of noise are reduced with increasing sample size.

In an ideal domain with an ideal description language, all regularities which are caused by genuine correlations between features of the domain will be observed as 'perfect' (ie. without counter-examples). However, in most real-world domains major regularities are rarely perfect and instead often have a few counter-examples in the training data. Such counter-examples arise from two causes :

- **Errors due to transcription**

  Whenever an example situation is presented to a learning algorithm it must be described in some manner. The process of recording and transcribing the attributes of an example is prone to error from many causes (for example imperfect measuring equipment, mistaken classification by an expert, typing errors etc.).

- **Errors due to an insufficient description language**

  It is possible that no rule or set of rules, within the space or rules defined by the description language, can completely and correctly classify all possible situations. This is quite common in medical domains, for example, where there may be several possible disease catagories for a given set of observations and where further observations (ie. a more complete description) are not possible for reasons of time or other costs.

If either or both of these features are present, then there is an important implication. This is that rules induced by an induction system for which there are counter-examples in the training data cannot be necessarily dismissed as not reflecting genuine correlations between attributes and classes in the domain, and hence, as having no predictive power. Indeed these rules may be *more* predictive than other more specific rules for which there are no counter-examples but are formed on the basis of fewer training examples, and hence, being more susceptible to noise.

2.2. Coping with Noise

These problems of noise and description language have long been recognized and many of the successful induction systems include methods designed to reduce their effect. Some of these methods are now reviewed.

2.2.1. Removal of Erroneous Training Examples

One such technique is to perform induction using only *representative* training examples, as selected by the expert or automatically, as was done by the ESEL system [8] for AQ11's application to the task of soya bean diagnosis.

2.2.2. Flexible Rule Application

A second technique is to use a 'flexible matching' procedure for applying the induced rules, whereby their interpretation involves the use of weights and probabilities instead of solely boolean values, thus exploiting to the maximum information contained in the training data. This technique also used by AQ11 and is on of the important features of the AQ15 algorithm, see [9], [10]. Misclassification by an erroneous rule may be overridden by other rules, whose conditions are nearly met and which have higher weight attached.

2.2.3. Rule Truncation

A third method is, after induction, to remove the rules which represent the weakest correlations found between attributes and classes, thus passing the classification task to other more reliable rules. This technique needs to be combined with flexible matching in order to allow examples which do not satisfy any condition of any rule to still be classified. AQ15 uses this technique, a truncation procedure being employed to remove the least reliable rules induced by the system.

2.2.4. The Consideration of Rules for which Counter-examples exist

A fourth method is to alter the rule generation procedure, often involving the relaxing of the constraint that the induced rules should be completely (or to the maximum extent, if completeness is impossible) consistent with the training data. This allows induction to be halted in regions of the search space where there is little training data to guide the system and where further search is as often damaging as beneficial. Such learning algorithms use top-down, hypothesis driven methods for forming rules, for example PLAGE [11] and CALM [12]. The pruning of decision trees is also an example of this technique, as is done by Assistant [13] and C4 [14]. Quinlan [15] presents a detailed empirical study of the effect of tree pruning in noisy domains.

2.2.5. The Use of Domain Knowledge

Additional domain knowledge can be used to reduce problems of description language and noise. Explanation-based generalization, (for example [16], [17]), constrains generalization to be performed only where the generalization's validity can be proved. DISCIPLE [18] uses explanations to guide generalization. AQ15 [9] allows the user to provide background knowledge to assist in induction.

This paper presents a description and empirical evaluation of a new induction system based on the 4th of these techniques, involving the relaxing of the requirement of complete consistency of rules with the training data during their generation. This system, CN2, has been designed with the aim of inducing short, simple, comprehensible rules in domains where problems of poor description language and/or noise may be present. Induced rules are in a form similar to production rules, with the condition being a conjunct of tests on an example's attributes and the conclusion being a class prediction. Thus, this paper can be viewed as an investigation of applying a method similar to tree pruning to the generation of 'production rule'-like expressions. These rules are interpreted in a logical manner (not involving weights etc.) in order to maintain their comprehensibility and keep inference well-defined for their later use.

2.3. The CN2 Algorithm

We now describe CN2, the algorithm proposed in this paper.

Rule Form

CN2 induces rules in a form similar to those generated by the AQ family of induction systems. A *selector* relates an attribute to an attribute value or disjunct of values, for example

[Cloudy = yes]

[Weather = wet v stormy]

[Temp > 60]

Attributes are typed, the type for each attribute being declared by the user. The 3 types used are binary, set or numeric, corresponding respectively to the 3 selectors above.

A selector or conjunction of selectors forms a *complex*. In CN2, a complex defines the condition part of a *decision rule* for identifying a particular class.

A *rule set* is an ordered list of rules for identifying classes† (there may be zero or more rules per class). For example, a rule set might be

[Advice = do_not_use_umbrella] <= [Weather = sunny]
[Advice = use_umbrella] <= [Weather = wet v stormy] & [Indoors = no]
[Advice = do_not_use_umbrella] <= [Indoors = yes]

Rule Interpretation

To use the rule set to classify new examples, CN2 applies a 'strict match' interpretation by which each rule is tried in order until one is found whose conditions are satisfied by the attributes of the example to classify. The prediction of this rule is then assigned as the class of that example. In this way, the ordering of the rule set is important. The rules could of course be converted to an order-independent form by adding negated selectors to the condition parts.

In the case of no rules being satisfied, a final 'default rule' assigns the class which occurred most frequently in the training examples to the new example to be classified.

Search Technique

hhhhhhhhhhhhhhhh

† It is also possible for CN2 to induce a rule set for a *single* class by altering the evaluation function guiding the rule search, for example by replacing CN2's entropy function (described later) by AQR's evaluation function (also described later).

During induction, the rule set is gradually built up by generating the rules one by one. Each rule is generated using the same following generation technique, though with a different set of training examples. The training example set provided for generating each new rule represents those examples *not* satisfying the conditions of any rule in the rule set generated so far.

ALGORITHM CN2

1. Search for a satisfactory rule
2. **IF** a satisfactory rule found

THEN g add it to the end of the rule set so far
g remove from the training set of examples those which satisfy this rule's <conditions>
g go to 1

ELSE stop.

Initially, the rule set is empty.

The system searches for predictive rules by performing a pruned general-to-specific search. The search is for a complex which is satisfied by a large number of examples of any single class, and few or none of other classes. The best complex found becomes the condition part of the new rule and the most common class of examples it covers becomes the class prediction of the rule.

At each stage in the search, a (size-limited) set or *star* S of 'best complexes found so far' is kept, and specializations of this set examined. Thus the system performs a beam search of the space. The evaluation functions used to bias and prune this search are discussed after the rule generation algorithm is described.

RULE GENERATION ALGORITHM

1. Form the set S of all *most general* complexes, ie. those containing only one selector.

   Set 'best rule' to be nil.

2. Evaluate each complex in S. Three types of evaluation are made :
   - what is its quality?
   - is it statistically significant?
   - are any specializations of it statistically significant?

   For each complex,
   - if not significant AND cannot become significant by specialization, discard it from S
   - if significant AND better than 'best rule', replace 'best rule' with this complex.

3. **IF** set S is not empty
   AND elements of S can be specialized further
   THEN specialize each complex in S in all ways possible and goto 2
   ELSE stop.

If the rule generation algorithm is provided with an empty set of training examples, it immediately fails to find a rule.

Specialization of a complex is performed by either extending it with a new conjunctive term or removing a disjunctive element a selector contains. Each complex can be specialized in several ways, and all specializations are generated and evaluated. Such general-to-specific search has a high branching ratio, and in order to constrain the search the star S of 'best complexes found so far' is limited to a user-defined maximum number of elements *maxstar*. Trimming of the star is performed after completion of step 2 by removing its lowest ranking elements as evaluated by an evaluation function.

One implementation of this specialization process is to repeatedly *intersect*† the set of most general complexes with itself, and after each intersection remove all self-contradictory and unchanged elements in the set.

Numeric attributes are dealt with in a similar manner to Assistant, where the range of values of an attribute is quantized into discrete regions. The complete range of values and size of each region is provided by the user. Consequently, selectors are of the form

<attribute> <comparator> <region boundary>

where <comparator> is either '≥' or '<'. Specialization occurs by shrinking the region covered by the selector, either by moving the region boundary or adding a second conjunctive selector to

hhhhhhhhhhhhhhhh

† The intersection of set A with set B is formed by making the conjunct of each element of A with each element of B in turn for all possible pairs, then removing duplicates. for example {a^b, a^c, b^d} intersected with {a, b, c, d} is {a^b, a^b^c, a^b^d, a^c, a^c^d, b^d, b^c^d}. If we now remove unchanged elements in this set we obtain {a^b^c, a^b^d, a^c^d, b^c^d}.

delineate both upper *and* lower bounds.

Following Quinlan's discussion of techniques for dealing with unknown attribute values in [14], we use the simple and effective method of replacing unknown values with the most commonly occurring value (or range, in the case of numeric attributes) for that attribute in the training data.

A Prolog implementation of this algorithm is given in the appendix.

Heuristics

The above algorithm performs three evaluations using two evaluation functions. Firstly is the function for assessing rule quality, determining if a new complex should replace the 'best rule' so far, and which complexes in the star S should be discarded if its maximum size is exceeded. To evaluate a complex, the set E of examples which it *identifies* (ie. which satisfy all its selectors) is found and the probability distribution **P** = (p_1,... p_n) of examples in E amongst classes calculated (where n = number of classes represented in the training data). For example, a given complex may identify 4 examples of class C_1, 2 of C_2, 1 of C_3 and none of C_4, hence **P** = (0.57, 0.29, 0.14, 0). Because the search is for complexes identifying a large number of examples of any single class and few of other classes, a function f = $p_{\max} - \sum p_{(\neq max)}$ (where $p_{\max}$ is the largest element of **P**) would be appropriate. However, it was found that using the information-theoretic measure entropy

$$E = -\sum p \log_2(p)$$

proved a better function to use (this function to be minimized). The behaviour of E and f are roughly comparable. However, entropy will distinguish cases such as **P** = (0.7, 0.1, 0.1, 0.1) and **P** = (0.7, 0.3, 0, 0) in favour of the latter, a desirable feature as there exist more ways of specializing the latter to a complex identifying only one class (for example if the examples of the majority class are excluded by specialization, the distributions become **P** = (0, 0.33, 0.33, 0.33) and **P** = (0, 1, 0, 0) respectively). In addition, entropy tends to direct the search in the direction of more significant rules ; empirically, rules of high entropy tend to also have high significance.

The second evaluation function used is to test whether a complex is *significant*. A complex identifying only examples of one class is an expression of a regularity found in the training data. By 'significant complex' we refer to one which expresses a regularity unlikely to have occurred by chance, ie. reflects a genuine correlation between attribute values and classes in the domain. To assess significance we compare the *observed* distribution amongst classes of examples satisfying the complex with the *expected* distribution under the null hypothesis that the complex is selecting examples randomly. Some differences in these distributions are to be anticipated owing to random variation. The question we ask is whether the observed distributions are too great to be accounted for purely by chance, ie. that selecting examples at random from the training set would rarely yield the distribution produced by the complex. If so, we consider that the complex is likely to reflect a genuine correlation between attributes and classes.

To test significance we use the likelihood ratio statistic [19], given by

$$2\sum_{i=1}^{n} f_i \log(f_i/e_i)$$

where the distribution **F** = $(f_1, \cdots, f_n)$ is the observed frequency distribution of examples amongst classes satisfying the complex in question and **E** = $(e_1, \cdots, e_n)$ is the expected frequency distribution of the same number of examples under the null hypothesis that the complex is selecting examples randomly. This statistic provides an information-theoretic measure of the (non-commutative) distance between the two distributions (we assume that **F** is continuous with respect to **E**, ie. that the f are zero when the e are zero). If **P** = $(p_1, \cdots, p_n)$ is the observed *probability* distribution of examples amongst classes satisfying the complex and **Q** = $(q_1, \cdots, q_n)$ is the probability distribution of examples in the whole training set, then this measure becomes

$$2N \sum_{i=1}^{n} p_i \log(p_i / q_i)$$

where $N = \sum f$ is the total number of examples satisfying the complex, since $e_i = q_i N$. Under suitable assumptions it can be shown that this statistic is distributed approximately as χ^2 with $n-1$ degrees of freedom. This measure indicates significance - the lower it is, the more likely it is that the regularity is due to chance choice of training examples.

In our situation **P** and **Q** are found from the observed distributions. We choose a significance threshold α and compute or look up the corresponding significance level for α with the number of degrees of freedom available for classification. Rules are rejected as being insignificant if the likelihood ratio statistic produces a value less than this significance level. In this case we are judging that selecting the same number of examples at random could have led to the observed distribution.

Thirdly, a check is made in the algorithm to examine whether specializations of complexes in the star *could* (but not necessarily will be) be significant. It does this by considering the probability distribution **P**, rather than actually generating such specializations. This check behaves as a fast look-ahead, and is included purely for efficiency. If there are no significant specializations possible, this complex can be discarded.

CN2 is implemented in Quintus Prolog and contains about 300 lines of code, taking approximately 5 minutes run-time to induce a rule set in the lymphography domain (see section 3) on a 4 megabyte SUN-3 using a value of maxstar = 15.

2.4. Related Work

We have adopted the technique of a top-down rule generation procedure coupled with significance thresholding to constrain the specialization process. In noisy domains, top-down approaches generally work well because the early stages of rule formation spread their dependence on training examples over a large proportion of the training set. Such top-down searches have been used in other induction systems, such as CALM [12], its derivative SEQUOIA [20], PLAGE [11] and by King [21]. These systems use different techniques for halting specialization, based on classificational accuracy on training examples and rule coverage. CALM, for example, includes a thresholding test for rules identifying positive examples at a rate greater than T_μ and negative examples at a rate lower than T (T_μ and T being user-supplied 'completeness' and 'contradiction' ratios). Similarly, the top-down ID3 algorithm can use such thresholding or 'pruning' techniques, such as was done in Assistant [13], C4 [14] and by Niblett and Bratko [22].

CN2 is characterized by its use of entropy to guide rule specialization, and the use of a significance test to halt specialization. These measures prove to be useful for controlling the search, and avoid the (sometimes elaborate) manipulation of tunable parameters required of the user until the algorithm induces well. In addition, tests based on percentage accuracies and coverage can miss small but statistically significant features of the domain. Consider a rule identifying three examples of the same class - the probability that the rule performed this well by chance is much larger if this class is common than if this class was very rare. Percentage accuracies and coverages cannot discern between these two cases, whereas a significance test can identify small but significant trends. Indeed, the authors of both SEQUOIA and CALM note that a χ^2 significance measure could be used as an alternative method for thresholding (see [20] and [23]).

Generally, simple bottom-up data-driven approaches cope poorly with noise, for example the AQR system described later. However, more sophisticated data-driven approaches have been successful, such as AQ11 [5] and AQ15 [9]. These systems use techniques to ensure that reliance on those examples which produce poor and specific rules is removed through various selection and pruning methods, reviewed earlier in section 2.2.

3. Experimental Comparison

CN2 was compared with four other algorithms in three medical domains. Firstly, we give a brief description of the algorithms used for comparison. Secondly, details of the medical domain are given and evaluation criteria presented.

3.1. Comparative Algorithms

3.1.1. Assistant

Assistant [13] is a descendant of ID3 [24] and CLS [25]. Assistant induces rules in the form of decision trees. The entropy measure is used to guide the growth of the decision tree, as described in [1]. In addition, Assistant can apply a tree pruning method based on a technique of maximal classification precision. This technique detects the node at which additional branching would cause more errors than if the tree building process was stopped at this particular node. The effect of tree pruning, as illustrated later in table 1, is that the decision tree is smaller whilst retaining the same predictive accuracy as the unpruned tree. One of the functions of the tree pruning mechanism is to reduce problems with noisy data - the pruning mechanism halts growth of nodes when the training examples used to guide growth become small in number. However, even in relatively noiseless domains the tree can be pruned without loss of predictive accuracy.

3.1.2. A Bayesian Classifier

In addition, the performance of a simple Bayesian classifier was examined and compared to the other algorithms. It should be noted that more sophisticated applications of the Bayes rule also exist in which the attribute tests are ordered [26]. The classifier used represents its ‘decision rule’ as a matrix of probabilities $p(v_j \mid C_k)$ of the occurrence of each attribute value given each class in turn. To classify a new example, Bayes’ theorem

$$p(C \mid \Lambda v_j) = \frac{p(\Lambda v_j \mid C_k)p(C_k)}{\sum_k p(\Lambda v_j \mid C_k)p(C_k)}$$

is applied where the summation is over the n classes and $p(C \mid \Lambda v_j)$ denotes the probability that the example is of class C given its attributes v_j (we denote the conjunct of the example's attributes by Λv_j). This probability is calculated for every class and the predicted class is then chosen as that of highest probability. $p(C_k)$ is estimated from the distribution of examples amongst classes in the training examples. If independence of attributes is assumed, $p(\Lambda v_j \mid C_k)$ can be calculated using the probability matrix.

3.1.3. AQR

AQR is an induction system using the basic AQ algorithm to generate classification rules. The AQ algorithm is used in a variety of ways by many systems, for example AQ11 [5] and GEM [6]. Many such systems use this algorithm in a more sophisticated manner than AQR to improve predictive accuracy and rule simplicity (for example AQ11 uses a more complex rule interpretation method involving degrees of confirmation), hence AQR represents a relatively simple AQ-based system. The AQ rule generation algorithm is described well in the literature (for example [3], [7], [8]) and only a brief overview will be presented here for comparison with CN2. Unlike CN2, AQR generates a decision rule for each class in turn. Having chosen a class to generate a rule for, a disjunct of complexes ('cover') forming the condition of the rule is generated in stages, each stage generating a single complex. After generating a complex, the examples covered by it are removed from the training set and another complex sought for. Complexes are generated until all the examples of the class are covered.

The search for a complex proceeds as follows. First, an example of the class to generate a rule for is chosen (the 'seed'). Next, an example of a different class (a 'negative example') is chosen, and the most general complexes satisfied by the seed but not by the negative example are generated. This set (or 'partial star') of complexes is then repeatedly specialized to exclude more and more negative examples whilst covering as many positive examples as possible, until all the negative examples are excluded. The best complex is then chosen and returned from the generation algorithm. To make the search tractable, the size of the partial star is limited to a (user-defined) maximum called *maxstar*, the worst elements being discarded should this be exceeded.

3.1.4. The Default Rule

In addition, the performance of a default rule was tested, which simply assigned the most commonly occurring class in the training data to all new examples to be classified, independent of their attributes.

3.2. Test Domains

The above algorithms were tested on three sets of medical data from the domains lymphography, prognosis of breast cancer recurrence and location of primary tumor. The data was obtained from the Institute of Oncology of the University Medical Center in Ljubljana, Yugoslavia [13]. This data is identical to that used to test AQ15 in [9] and [10], and are now described.

3.2.1. Lymphography

Examples in this data set used 18 attributes, with four possible final diagnostic classes. 148 examples were available. The data was consistent, ie. examples of any two classes were always different. All the tested algorithms produced fairly simple and accurate rules. Unlike the other two domains, this data set was not submitted to a detailed checking after it was originally compiled by the medical center, and thus may contain errors in attribute values.

3.2.2. Prognosis of Breast Cancer Recurrence

For about 30% of patients that undergo a breast cancer operation, the illness reappears after five years - thus prognosis of recurrence is important. This domain contained 9 attributes with two possible classes of 'no recurrence' and 'recurrence'. Data for 286 patients were known (201 who did not have recurrence after five years, 85 who did). This data was verified after collecting, and thus is likely to be relatively error-free. The set of attributes was found to be relatively incomplete, ie. not sufficient to induce high quality rules.

3.2.3. Location of Primary Tumor

Physicians distinguish among 22 possible locations of primary tumor, thus there were 22 classes in this domain. 17 attributes were used describe the patients' diagnostic data. The data was inconsistent, ie. examples of different classes existed with identical attribute values. This data was verified after collecting, and thus is likely to be relatively error-free. The set of attributes was found to be relatively incomplete, ie. not sufficient to induce high quality rules.

3.3. Evaluation Criteria

Ultimately, the evaluation of the performance of these systems as producing meaningful rules must be done by an expert in the application domain. However, it is difficult to quantify such an assessment. It has been shown on several occasions (for example [4], [9], [14]), that the more quantifiable measures of classificational accuracy and rule simplicity are good general indicators of a rule's use to an expert. This empirical result is not surprising as we expect an expert to be able to classify examples himself or herself accurately also. We use these two measures to evaluate the induction systems described, thus also making this assumption that they are good indicators of a rule's utility. Direct examination of the produced rules by an expert has not yet been carried out.

It should be also noted that there are other methods of assessment. One is to measure how noise resistant the rules are, by introducing small variations to the training data and examining the stability of the complexes. In addition, the performance of the algorithms in terms of computer time and memory required to generate and apply their rules can be measured. These properties are not reviewed here, apart from making the observation that the systems were all able to induce rule sets in the domains tested in an 'acceptable' time (less than 30 minutes run-time on a SUN-3). O'Rorke [3] and Jackson [27] provide detailed comparisons of time and memory requirements of ID3 and AQ11P in generating rules from examples using large chess endgame databases.

3.3.1. Classificational Accuracy

Classificational accuracy is assessed by presenting the algorithms with examples not used at induction time for inducing the rules, and measuring the percentage of times that the example's class is correctly predicted by the system. Cross-algorithm comparisons of rule complexity are difficult, due to the large differences in rule languages used by the systems and the degree of subjectivity involved in assessment of a rule's complexity. For Assistant's decision trees we measure complexity by the number of nodes (including leaves) in the tree. For CN2 and AQR we measure complexity by the number of selectors in the final rule set. These measures reveal the gross features of the induced decision rules. More detailed measures of rule complexity have been done by O'Rorke [3] but are not used here.

3.3.2. Rule Complexity

Assessment of the complexity of a Bayesian rule is more difficult. One measure would be to count the number of elements in the $p(V_j \mid C_k)$ matrix. However, such a measure is independent of the training examples and also does not take into account features of the matrix which may make it more comprehensible (for example a few elements may be very large, the rest small). In addition, the complexity of applying the rule is not taken into account. Due to these difficulties and the subjectivity involved, we do not give a firm estimate of the complexity of Bayes rules but only provide the size of the matrix as a rough guide.

A complexity of 1 is assigned to the default rule as it is equivalent to a decision tree with a single leaf node.

3.3.3. Combining Accuracy and Complexity

We do not attempt to combine measures of accuracy and complexity, for example using a function $f = (accuracy - \alpha\ complexity)$, as such a combining function is dependent on the purpose for which the rules are to be used. For example, a 1% fall in predictive accuracy may or may not justify a 10% decrease in rule complexity, depending on the application.

4. Results and Discussion

The five algorithms were tested on each of the domains. Assistant was tested with and without tree-pruning applied and CN2 tested at three levels of significance threshold. The results of testing were averaged over five trials for each algorithm in each domain. They are shown in table 1. In each test the training examples were selected at random from the entire data set, and the remaining used for testing. The algorithms were all then run and their rules tested using the same training and testing examples for each algorithm. In each case 70% of the total data was used as training examples and the remaining 30% for testing.

From these results some interesting observations can be made. Most importantly, the algorithms designed to reduce problems caused by noisy data (tree pruning Assistant and CN2) achieve a lower rule complexity without damaging their predictive accuracy. In the lymphography domain, for example, CN2 was able to achieve the same classificational accuracy as the

hhhhhhhhhhhhhhhh

† Complexity not supplied for Bayes classifier, as discussed earlier. The size of the probability matrix for lymphography was 240 elements, for breast cancer 540 elements and for primary tumor 465 elements.

† AQR tries to attain 100% accuracy on the training data as far as is possible, but is unable to achieve this

Table 1. Measurements of rule accuracy and complexity of the tested algorithms (cmpxs = number of complexes, sels = number of selectors per complex).

Domain	Algorithm	Accuracy	Complexity	
Lymphography	Default rule	56%	1	(always choose most common class)
	Assistant :			
	(no pruning)	79%	41	(41 nodes, inc. 24 leaves)
	(pruning)	78%	36	(36 nodes, inc. 21 leaves)
	Bayes	83%	†	
	AQR	76%	76	(20 cmpxs, av. 3.8 sels)
	CN2 :			
	(90% threshold)	78%	24	(18 cmpxs, av. 1.3 sels)
	(95% threshold)	81%	22	(15 cmpxs, av. 1.5 sels)
	(99% threshold)	82%	12	(8 cmpxs, av. 1.6 sels)
Breast Cancer	Default rule	71%	1	(always choose most common class)
	Assistant :			
	(no pruning)	62%	112	(112 nodes, inc. 59 leaves)
	(pruning)	68%	44	(44 nodes, inc. 24 leaves)
	Bayes	65%	†	
	AQR	72%	208	(47 cmpxs, av. 4.4 sels)
	CN2 :			
	(90% threshold)	70%	28	(12 cmpxs, av. 2.5 sels)
	(95% threshold)	70%	20	(8 cmpxs, av. 2.2 sels)
	(99% threshold)	71%	4	(3 cmpxs, av. 1.3 sels)
Primary Tumour	Default rule	26%	1	(always choose most common class)
	Assistant :			
	(no pruning)	40%	178	(178 nodes, inc. 93 leaves)
	(pruning)	42%	52	(52 nodes, inc. 27 leaves)
	Bayes	39%	†	
	AQR	35%	562	(105 cmpxs, av. 5.4 sels)
	CN2 :			
	(90% threshold)	37%	33	(8 cmpxs, av. 3.6 sels)
	(95% threshold)	36%	42	(10 cmpxs, av. 4.3 sels)
	(99% threshold)	36%	19	(5 cmpxs, av. 3.4 sels)

Table 2. Accuracy of the different algorithms on training and testing data.

Domain	Algorithm	Accuracy Of Decision Rules On Training Data	Accuracy Of Decision Rules On Testing Data
	Default rule	54%	56%
	Assistant (no pruning)	100%	79%
	Assistant (pruning)	98%	78%
	Bayes	89%	83%

Lymphography	AQR	100%	76%
	CN2 (90% threshold)	100%	78%
	CN2 (95% threshold)	99%	81%
	CN2 (99% threshold)	91%	82%
	Default rule	70%	71%
	Assistant (no pruning)	92%	62%
	Assistant (pruning)	85%	68%
Breast	Bayes	70%	65%
Cancer	AQR	100%	72%
	CN2 (90% threshold)	76%	70%
	CN2 (95% threshold)	74%	70%
	CN2 (99% threshold)	72%	71%
	Default rule	23%	26%
	Assistant (no pruning)	73%	40%
	Assistant (pruning)	53%	42%
Primary	Bayes	48%	39%
Tumour	AQR	75%†	35%
	CN2 (90% threshold)	40%	37%
	CN2 (95% threshold)	45%	36%
	CN2 (99% threshold)	37%	36%

other algorithms by inducing on average only 8 short rules at the highest threshold value tested.

It is perhaps surprising that different methods of halting the rule specialization process, besides having the desirable effect of reducing rule complexity, do not greatly affect predictive accuracy. This effect has been reported in a number of papers, (for example [9], [10], [13], [22]). Indeed it perhaps may be the case that any technique will have this effect, providing a certain maximum level of pruning is not exceeded. If this is the case then an algorithm should be preferred if it most closely estimates this maximum level.

Secondly, in our experiments in the breast cancer domain none of the algorithms were able to induce rule sets which classified more accurately than the default rule ("always choose the most common class in the training data"). Indeed, there seems to be little if any observable correlation between attribute values and final class. This is reflected in CN2's inability to locate significant complexes during induction. As the threshold on CN2 is increased, the significance test for assessing a rule's performance becomes more strict - at a cutoff of 99% CN2 was virtually unable to find any significant rules in this domain at all (on average four complexes containing 1.3 selectors in each). This contrasts with CN2's performance in the two other domains at the same threshold level, where a greater number of rules were located. Thus significance thresholding has the desirable effect that if there are few or no correlations in the data, it will induce few or no rules to supplement the default rule of chosing the most common class, rather than induce regardless of the correlations in the data.

Thirdly, in the primary tumour domain Assistant's decision trees perform slightly better than the rules induced in an production rule form by AQR and CN2. This result was also found in our earlier tests with these algorithms, and is shown in the results presented here. This third

domain has a large number of different classes (21 populated classes) unlike the lymphography and breast cancer domains (4 and 2 classes respectively), which could perhaps be responsible for this effect.

Fourthly, it should be noted that the Bayesian classifier tested also produced accurately classifying decision rules and that the superiority in terms of classificational accuracy of other induction algorithms over the Bayesian classifier was not demonstrated in our tests. As stated earlier, the classifier tested used a relatively simple application technique of the Bayes rule - more complex statistical classifiers could well perform even better.

Both tree pruning Assistant and CN2 apply a similar technique to reducing rule complexity, namely sometimes halting specialization of branches/rules *before* they classify training examples perfectly. Thus CN2 can be partly viewed as applying a kind of 'tree pruning' technique to decision rules expressed in terms of complexes rather than trees. By performing such pruning these 'noisy' algorithms are applying the principle that a sacrifice of classificational accuracy on the *training* data may achieve increased rule simplicity whilst not damaging the predictive accuracy on new test data. This can be seen in table 2. The performance of the pruned assistant trees on the *training* data compared with that of the unpruned trees is lower, as is that of CN2 compared with AQR. However, as was shown in table 1, the *predictive* accuracy of the rules was not impaired whilst their complexity reduced.

Table 3. Degree of generalization achieved per complex, measured as the average number of training examples covered per complex in the final rule set.

Domain	Algorithm	Coverage of rule set (no. of exs)	Number of cmpxs	Coverage per cmpx (no. of exs)
Lymphography	AQR	102	20	5.1
	CN2 (90% threshold)	97	18	5.4
	CN2 (95% threshold)	86	15	5.7
	CN2 (99% threshold)	73	8	9.1
Breast Cancer	AQR	197	47	4.2
	CN2 (90% threshold)	75	12	6.3
	CN2 (95% threshold)	63	8	7.9
	CN2 (99% threshold)	24	3	12.0
Primary Tumour	AQR	204	105	1.9
	CN2 (90% threshold)	73	8	9.1
	CN2 (95% threshold)	95	10	9.5
	CN2 (99% threshold)	51	5	10.2

The generality of induced rules was also investigated. In table 3 the number of examples covered per complex in the final rule set is shown. As the significance threshold for CN2 is increased, the generality of the rules becomes greater as they cover more examples each. By a complex using more examples to assess its performance, its apparent accuracy on training data is more likely to reflect its genuine accuracy on new testing data. This desirable feature is

counteracted by the increasing rarity of such general, accurate rules in the space as the generality of complexes is increased. The resultant rule set at high significance threshold contains a smaller number of more general rules, which may classify the training examples well but not 100% correctly.

AQ15 similarly achieved large reduction in rule complexity without damage to predictive accuracy, compared with complete rule sets generated by the AQ algorithm. However, the methods by which AQ15 and CN2 achieve these results differ. AQ15 first uses an advanced form of the AQ algorithm to generate a rule set. Secondly, a method of *knowledge reduction* is applied which truncates ordered covers and uses them to classify new examples by a technique of *flexible matching*, based on an assessment of the degree of similarity between the example and the condition part of rules. CN2 uses a different rule generation procedure and may halt specialization of candidate rules in regions of the search space where there are few examples (where further specialization is considered insignificant as described in section 2). These rules are applied to new examples using a 'strict match', examining whether the rule's conditions are satisfied or not by new examples. AQ15's method of cover truncation and CN2's halting of rule specialization can be viewed as applying techniques of 'post-pruning' and 'pre-pruning' respectively, in order to avoid specific poorly-classifying rules being included in the final rule set.

5. Conclusion and Future Issues

This paper has demonstrated that a relatively simple rule induction algorithm is able to achieve a large reduction in rule complexity without damaging classificational accuracy as compared with the other algorithms tested. Especially important is that this result was found in both relatively noiseless and noisy domains. Secondly, the induced rules are in a relatively simple form, similar to the the standard production rule framework used in many expert systems. Both these features are important requirements for practical applications of such an induction system. It should be noted that the rules induced by CN2 may not cover the entire space of examples (ie. there may exist examples for which no rule has its condition part satisfied). This is a feature also common to many AQ-based induction systems, but not to many ID3-based systems where the decision tree will classify all new examples.

Another important result is that the method of significance testing using the likelihood ratio test proved an effective mechanism for controlling search and avoiding regions of the space where examples were sparse.

Currently CN2 requires that an arbitrary confidence parameter is supplied by the user. One direction for future work would be to investigate methods whereby such a parameter can be chosen automatically by the system.

CN2, like tree pruning Assistant, may halt specialization of rules before they classify the training examples perfectly during rule generation. An alternative approach would be to allow the specialization process to continue until the rules did classify perfectly, and then apply a post-pruning technique to either generalize the rules again or reject them completely. CN2 makes an estimate of the utility of attempting rule specialization (using the significance test as described in section 2). Post-pruning techniques actually perform such specialization and then having done so can evaluate the performance exactly. Post-pruning techniques have been used in AQ15 as mentioned earlier, on ID3 trees [22] and may also be applicable to the CN2 rule generation procedure.

Acknowledgments

This work was supported by the Office of Naval Research under contract N00014-85-G-0243. We would like to thank Professor Donald Michie for his careful reading and valuable comments on earlier drafts of this paper.

REFERENCES

1. Quinlan J. (1983) Learning efficient classification procedures and their application to chess end games, *Machine Learning: an artificial intelligence approach* Ed. R.Michalski, J.Carbonell and T.Mitchell, Palo Alto, CA: Tioga.

2. Shapiro A., Niblett T. (1982) Automatic induction of classification rules for a chess endgame, *Advances in Computer Chess 3.* Ed. Clarke M.R. Oxford: Pergamon pp.73-91.

3. O'Rorke P. (1982) *A comparative study of inductive learning systems AQ11P and ID3 using a chess end-game test problem,* Urbana: Univ. of Illinois at Urbana-Champaign, Dept. of Computer Science report (ISG 82-2).

4. Quinlan J., Compton P.J., Horn K.A., Lazarus L. (1986) Inductive knowledge acquisition: a case study *Proceedings of the second Australian Conference on the Applications of Expert Systems* Sydney: New South Wales Institute of Technology pp.183-204.

5. Michalski R., Larson J. (1983) *Incremental generation of VL1 hypotheses : the underlying methodology and the description of program AQ11,* Urbana: Univ. of Illinois at Urbana-Champaign, Dept. of Computer Science report (ISG 83-5).

6. Reinke R.E. (1984) *Knowledge acquisition and refinement tools for the ADVISE META-EXPERT system,* Urbana: Univ. of Illinois at Urbana-Champaign, Dept. of Computer Science, M.S.Thesis (ISG 84-4, UIUCDCS-F-84-921).

7. Michalski R., Chilausky R. (1980) Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean diagnosis, *Policy Analysis and Information Systems, Vol.4 No.2* pp.125-160.

8. Michalski R., Larson J. (1978) *Selection of most representative training examples and incremental generation of VL1 hypotheses : the underlying methodology and the description of programs ESEL and AQ11* Urbana: Univ. of Illinois at Urbana-Champaign, Dept. of Computer Science (UIUCDCS-R-78-867).

9. Michalski R., Mozetic I., Hong J., Lavrac N. (1986a) The AQ15 inductive learning system : an overview and experiments *Proceedings of IMAL 1986,* Orsay: Université de Paris-Sud.

10. Michalski R., Mozetic I., Hong J., Lavrac N. (1986b) The multi-purpose incremental learning system AQ15 and its testing application to three medical domains *AAAI-86* Ca: Kaufmann.

11. Gascuel O. (1986) PLAGE: A way to give and use knowledge in learning *Proceedings of EWSL 1986* Orsay: Université de Paris-Sud.

12. Quinqueton J., Sallantin J. (1986) CALM: contestation for argumentative learning machine *Machine Learning: A guide to current research* Ed: Mitchell T.M., Carbonell J.G., Michalski R.S. Boston: Kluwer.

13. Kononenko I., Bratko I., Roskar E. (1984) Experiments in automatic learning of medical diagnostic rules. Presented at the *International School for the Synthesis of Expert Knowledge Workshop 1984*, Bled, Yugoslavia. Also published as a Technical Report, Faculty of Electrical Engineering, E. Kardelj University, Ljubljana, Yugoslavia 1984.

14. Quinlan J. (1986) Induction of Decision Trees *Machine Learning vol. 1 no. 1* Boston: Kluwer.

15. Quinlan J. (1986) Learning from noisy data, *Machine Learning vol. 2* Ed. R.Michalski, J.Carbonell and T.Mitchell, Palo Alto, CA: Tioga.

16. DeJong G. (1981) Generalizations based on explanations *IJCAI-81* Vancouver, B.C.: Kaufmann pp. 67-69.

17. Mitchell T.M., Keller R.M., Kedar-Cabelli S.T. (1986) Explanation-Based Generalization : A unifying view. *Machine Learning vol. 1 no. 1* pp. 47-80.

18. Kodratoff Y., Tecuci G. (1986) Rule Learning in DISCIPLE *Proceedings of EWSL 1986* Orsay: Université de Paris-Sud.

19. Kalbfleish J. (1979) *Probability and Statistical Inference II*, New York: Springer-Verlag.

20. Haiech J., Quinqueton J., Sallantin J. (1986) Concept formation from sequential data *Proceedings of EWSL 1986* Orsay: Université de Paris-Sud.

21. King R. (1987) An inductive learning approach to the problem of predicting a protein's secondary structure from its amino acid sequence *Proceedings EWSL 1987* Wilmslow, UK: Sigma (to be published).

22. Niblett T., Bratko I. (1986) Learning decision rules in noisy domains. Presented at *Expert Systems 86* Brighton, 15-18 Dec. and published in Conference Proceedings.

23. Quinqueton J., Sallantin J. (1983) Algorithms for learning logical formulas *IJCAI-83*, pp.476-478.

24. Quinlan J. (1979) Discovering rules by induction from large collections of examples *Introductory readings in expert systems* Ed. D. Michie, London: Gordon and Breach pp.33-46.

25. Hunt E.B., Marin J., Stone P.T. (1966) *Experiments in Induction* New York: Academic Press.

26. Wald A. (1947) *Sequential Analysis* New York: Wiley.

27. Jackson J.A. (1985) *Economics of automatic generation of rules from examples in a chess end-game*, Urbana: Univ. of Illinois at Urbana-Champaign, Dept. of Computer Science (UIUCDCS-F-85-932).