Proceedings of the International ICSC Symposium on Advanced in Intelligent Data Analysis (AIDA),
The Rochester Institute of Technology, June 22-25, 1999.

A Measure of Description Quality for Data Mining and its Implementation in the AQ18 Learning System

Ryszard S. Michalski* and Kenneth A. Kaufman

Machine Learning and Inference Laboratory
George Mason University
{michalski,kaufman}@gmu.edu

* Also with Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland

Abstract

Given a sufficiently large database, it is usually possible to derive many different hypotheses about the data. Therefore, an important problem in data mining is which hypothesis to select, or, more specifically, how to define a criterion that best articulates the requirements of the given task domain. This paper presents an approach to this problem, in which a knowledge agent seeks descriptions that optimize a problem-oriented *description quality* measure. The proposed measure flexibly combines two criteria characterizing descriptions, *completeness* and *consistency gain*, into a single numerical measure of description *quality* Q(w). It is shown that by modifying the parameter w in Q(w), the proposed description quality specializes to different measures described in the literature. The Q(w) measure has been implemented in the AQ18 learning system, and compared to several other methods, such as Information Gain, PROMISE, CN2, IREP and RIPPER. A general measure of *description utility* can be obtained by integrating the description quality with *description simplicity* through a *Lexicographic Evaluation Functional* (LEF). Experimental results have demonstrated the generality and the flexibility of the proposed method.

1 The Problem Statement

A typical objective in extracting knowledge from data is to hypothesize general rules or patterns that can be used effectively for predicting or classifying future data. Given a sufficiently large database, one can generate a large number of hypotheses characterizing the data. A question then arises as to what criteria will lead to a hypothesis with the maximum predictive accuracy, as well as other desirable properties, such as generality and simplicity.

If one can make the assumption (usually unrealistic) that data contains no noise, the preconditions for admissibility of a hypothesis are consistency (no contradiction with any datapoints) and completeness (coverage of all datapoints). In real-world applications involving large databases, data often contains errors or inconsistencies; the completeness and consistency criteria should therefore be relaxed in such situations. These two criteria tend to push in opposite directions; increasing completeness may also increase inconsistency, and vice versa. Therefore, one may seek a hypothesis that is maximally consistent with the data at the expense of completeness, maximally complete at the expense of inconsistency, or a hypothesis representing some combination of the two criteria. In addition, one may seek a hypothesis that strives to maximize some criterion of computational and/or cognitive simplicity.

The problem of concern in this paper is how to combine the above criteria of hypothesis desirability into one general measure of hypothesis quality. Specifically, it proposes a method for combining completeness and *consistency gain* (a newly introduced measure that characterizes the description improvement over a random guess) into one flexible measure Q(w). It also shows how to combine Q(w) with a measure of description simplicity into one general measure of *hypothesis utility*. Q(w) is compared experimentally to several other measures, such as Information Gain, PROMISE, and those used in the CN2, IREP and RIPPER rule learning systems.

2 Combining Multiple Criteria

A general measure of description quality needs to take into consideration several criteria, such as description completeness, consistency, computational simplicity and comprehensibility. This paper discusses two methods for integrating multiple criteria into one measure, combining individual criteria through a numerical function, and employing a *lexicographical evaluation functional*, (Michalski, 1983). The first method is used to combine completeness and consistency gain (a newly introduced measure) into the description quality measure Q(w) (see Section 4), and the second method is used to combine Q(w) with some measure of description simplicity. The description quality measures presented here are general, i.e., independent of the description language. They can therefore be applied to all types of data descriptions or patterns in data. In this paper, we present them in the context of rule learning methods.

The proposed measure has been implemented in the AQ18 learning system (Kaufman and Michalski, 1999). Here a description of data representing one class (i.e., data records sharing the same value of a designated output attribute) consists of a set of *attributional rules*. Unlike the conventional decision rules that use only *<attribute>* = *<value>* or *<attribute>* *<rel>* *<value>* conditions, the attributional rules employ a much wider set of operators, defined in the *attributional calculus* (Michalski, 1999). The attributional calculus is a logic-based description language, with an expressive power between propositional calculus and predicate calculus. It was specifically designed to enhance both the expressive power and understandability of the descriptions, as well as their closeness to equivalent natural language descriptions. For that reason, the method of learning attributional descriptions from examples has been termed *natural induction* (Michalski, 1999).

Among the most important characteristics of a single rule are the *completeness* (the percentage of positive examples covered by it), and the *consistency* (the percentage of examples covered by it that are in the positive class). These two criteria pull in opposite directions. A rule that covers more positive examples has a tendency to cover more negative examples, and conversely, a rule that covers fewer negative examples has a tendency to also cover fewer positive examples. Therefore, one seeks a criterion for evaluating the rule quality that strikes an appropriate balance between these two requirements (e.g., Bergadano et al, 1992). Several measures for evaluating rule quality have been presented in the literature (e.g., Bruha, 1997). No single criterion can fit, however, all practical problems. For different problems, different criteria may be preferable. Therefore, it would be advantageous to define a general and flexible criterion that takes into consideration different aspects of a description, and can be adjusted appropriately for different problems. This paper proposes such a criterion and compares it with several existing criteria.

The method presented here views pattern detection as a process of searching for a general rule that optimizes a multicriterion measure, defined to best reflect the characteristics of the problem at hand. The measure consists of *elementary criteria*, each representing one aspect of the rule being evaluated. These elementary criteria are selected from a menu of available elementary criteria. They are combined into one measure through a *lexicographical evaluation functional* (LEF), defined as a sequence:

$$<(c_1, \tau_1), (c_2, \tau_2), \ldots, (c_n, \tau_n)>$$

where c_i represents the ith elementary criterion, and τ_i is the *tolerance* associated with c_i. The tolerance defines the range, either absolute or relative, within which the value of c_i for a candidate rule can deviate from the best value in the current set of rules. Let us assume, for example, that S is a set of candidate patterns, and we define two elementary evaluation criteria: to maximize completeness, and also consistency. Let us assume further that hypotheses with completeness less than 10% below the maximum completeness achievable by any single rule in S may still be acceptable, and that if two or more hypotheses satisfy this criterion, the one with the highest consistency is to be selected. This rule selection process can be specified by the following LEF:

LEF = < (completeness, 10%), (consistency, 0%)>

It is possible that after applying both criteria, more than one hypothesis remains in the set of candidates. In this case the one that maximizes the first criterion is selected.

The advantages of the LEF approach are that it is very simple and efficient, so that it can be effectively applied to a very large number of candidate hypotheses. This way of taking into consideration both completeness and consistency requires a specification of the relative importance of these criteria in the problem domain.

3 Completeness, Consistency, and Consistency Gain

As the primary purpose of the rules is to classify future, unknown cases, a typical measure of rule quality is the *testing accuracy*, that is, the accuracy of classifying testing examples, which are different from the training examples. During a learning process, however, testing examples are unavailable; one therefore needs a criterion that will be a good approximator of the testing accuracy using only training examples. Before we propose such a measure, we need to introduce some notation and terminology. In the literature, some of the concepts used here have been given different names and notations. Here, we propose a notation that is easy to understand and as consistent as possible with the previous terminology.

Let P and N denote the total number of positive and negative examples, respectively, in the training set of some concept or decision class. Let R be a rule or a set of rules generated to cover examples of that class, and p and n be the number of positive and negative examples covered by R, respectively. The ratio p / P, denoted compl(R), is called the *completeness* or *relative coverage* of R. The ratio $p / (p + n)$, denoted cons(R), is called the *consistency* or *training accuracy* of R and $n / (p + n)$, denoted incon(R), is the *inconsistency* or *training error rate*. If the relative coverage of a ruleset (a set of rules for a single class) is 100%, the ruleset is a *complete cover*. If the inconsistency of the ruleset is 0%, it is then a *consistent cover*.

Let us consider the question as to which is preferable: a rule with 60% completeness and 99.7% consistency, or a rule with 95% completeness and 98% consistency. Clearly, the answer depends on the problem at hand. In some application domains, notably in science, a rule (law) must be consistent with all the data unless some of the data are found erroneous. In other applications, in particular, data mining, one may seek strong patterns that hold frequently, but not always. Therefore, there is no single measure of rule quality that is good for all problems. Instead, we seek a flexible measure that can be easily changed to fit any given problem at hand.

How then can we define such a measure of rule quality? Let us first look at a measure based on the *information gain* criterion, which is used as a method for selecting attributes in decision tree learning (e.g. Quinlan, 1986). This criterion can also be used for selecting rules, because a rule can be viewed as a binary attribute that partitions examples into those that satisfy the rule, and those that do not satisfy it. The information gained by using rule R to partition the example space E is:

$$\text{Gain(R)} = \text{Info(E)} - \text{Info}_{\text{R}}\text{(E)}$$

where Info(E) is the expected information from defining the class of an example in E and $\text{Info}_{\text{R}}\text{(E)}$ is the expected information after partitioning the space using rule R. This measure is based on the distribution of the examples both in the space covered by the rule and in the space not covered by it; as such, it does tend to increase both with rule completeness and consistency.

Information gain has, however, one major disadvantage as a rule evaluator. As mentioned above, it relies not only on the informativeness of the rule, but also the informativeness of the complement of the rule. That is, it takes into consideration the entire partition created by the rule, rather than solely on the space covered by it. This concern is especially valid if there are more than two decision classes. In such a situation, a rule may be highly valuable for classifying examples of one specific class, even if it does little to reduce the entropy of the training examples in other classes. Another limitation of this measure is that it does not provide means for modifications in order to fit different problems (i.e., putting different levels of emphasis on consistency and completeness).

Before we propose another measure of rule quality, let us observe that relative frequency of positive and negative examples in the training set of a given class should be a factor in evaluating a rule. Clearly, a rule with 15% completeness and 75% consistency could be quite attractive if the total number of positive examples was very small and the total number of negative examples was very large. On the other hand, the same rule would be useless if P was very large and N was very small. The distribution of positive and negative examples in the whole training set can be measured by the ratio $P / (P + N)$. The distribution of positive and negative examples in the set covered by the rule can be measured by the consistency $p / (p + n)$. Thus, the difference between these distributions is $(p / (p + n)) - (P / (P + N))$. This value can be normalized by dividing it by $(1 - (P / (P + N)))$, or equivalently $N / (P + N)$, so that in the case of identical distribution of positive and negative events in the set covered by the rule and in the training set, it returns 0, and in the case of perfect consistency, it will return 1. This normalized consistency measure thus shares the *independence property* with statistical rule quality measures (Bruha, 1997).

This normalized consistency thus provides an indication of the benefit of the rule over a randomly guessed assignment to the positive class. It also allows for the possibility of negative values, in accordance with our assertion that a rule less accurate than the random guess based the example distribution has a negative benefit. Reorganizing the normalization term, we define the *consistency gain* of a rule R, or briefly consig(R), as:

$$\text{consig(R)} = ((p / (p + n)) - (P / (P + N))) * (P + N) / N$$

4 Measuring Description Quality

In developing a description quality measure, one may assume the desirability of maximizing both the completeness compl(R), and the consistency gain, consig(R). Clearly, a rule with both higher compl(R) and consig(R) is more desirable than one with lower values. A rule with either value equal to 0 is worthless. It makes sense, therefore, to define a description quality measure that evaluates to 1 when both of these components reach maximum (have value 1), and 0 when either is equal to 0.

One way to achieve such a behavior would be to define a description quality as a product of compl(R) and consig(R). Such a formula, however, would not allow one to weigh these factors differently in different applications. To achieve this flexibility, we introduce a weight, w, defined as the percentage of the description quality measure to be borne by the completeness condition. Thus, the final form of the *description quality*, Q(R,w) with weight w, or just Q(w), if the rule R is implied, is:

$$Q(R, w) = \text{compl}(R)^{w} * \text{consig}(R)^{(1 - w)}$$

where compl(R) = p / P, and consig(R) = ((p / (p + n)) – (P / (P + N))) * (P + N) / N.

By changing parameter w, one can change the relative importance of completeness and the consistency gain to fit a given problem. The above definition satisfies all the criteria regarding desirable features of a rule evaluation function given by Piatetsky-Shapiro (1991):

1. The rule quality should be 0 if the example distribution in the space covered by the rule is the same as in the entire data set. Note that Q(R,w) = 0 when $p / (p + n) = P / (P + N)$, assuming $w < 1$.

2. All other things being equal, an increase in the rule's completeness should increase the quality of the rule. Note that Q(R,w) increases monotonically with p.

3. All other things being equal, the quality of the rule should decrease when the ratio of covered positive examples in the data to either covered negative examples or total positive examples decreases. Note that Q(R,w) decreases monotonically as either n or (P - p) increases, when $P + N$ and p remain constant.

Another important criterion in measuring description utility is its *simplicity*, measured in terms of the number of expressions and operators are required to represent it in the given description language. This measure can be combined with Q(R,w) in a LEF to express overall *description utility*.

5 Empirical Comparison of Rule Evaluation Methods

In order to develop a sense of how the Q(w) rule rankings compare to those done by other methods used in machine learning systems, we performed a series of experiments on different datasets. In the experiments we used the Q(w) method with different weights (0, 0.25. 0.5, 0.75, 1), the information gain criterion (Section 3), the PROMISE method (Baim, 1982; Kaufman, 1997), and the methods employed in the CN2 (Clark and Niblett, 1989), IREP (Fürnkranz and Widmer, 1994), and RIPPER (Cohen, 1995) learning programs.

As was mentioned above, the information gain criterion is based on the entropy of the examples in the area covered by a rule, the area not covered by the rule, and the event space as a whole. Like the information gain criterion, the PROMISE method (Baim, 1982) was developed to evaluate the quality of attributes, but can be usedfor rule evaluation in a similar manner to the aforementioned information gain criterion. In this context, the value is determined based on the following expressions (which assume that $p > n$):

(1) Compute $M = \max(P - p, N - n)$
(2) Assign to T the value P if $P - p > N - n$, and value N if $P - p \leq N - n$

PROMISE returns a value of (p / P) + (M / T) - 1 (the "1" is a normalization factor to assure that all values are in the range 0 to 1). When M is based on the positive class (when $P - p > N - n$), PROMISE returns a value of zero. Hence, PROMISE is not a useful measure of rule quality when positive examples significantly outnumber the negative ones. Note also that when $P = N$ and p exceeds n, the PROMISE value reduces to (p / P) + ((N - n) / N) – 1, which is equal to (p / P) + ((P - n) / P) – 1, or:

$$(p - n) / P$$

CN2 (Clark and Niblett, 1989) builds rules using a beam search, as does the AQ-type learner, on which it was partially based. In the case of two decision classes, it selects a rule that minimizes the expression:

$$-((p /(p+n)) \log_2(p /(p+n)) + (n /(p+n)) \log_2(n /(p+n)))$$

This expression involves only the consistency, p / (p + n); it does not involve any completeness component. Thus, a rule that covers 50 positive and 5 negative examples is deemed of identical value to another rule that covers 50,000 positive and 5000 negative examples. Although the CN2 formula has a somewhat different form than the consistency gain component of Q(w), CN2's rule evaluation can be expected to be similar to Q(0) (consig(R) only). Indeed, in the examples shown below, the two methods provide identical rule rankings.

A later version of CN2 (Clark and Boswell, 1991) offered a new rule quality formula based on a Laplace error estimate. This formula is closely tied to a rule's consistency level, while completeness still plays a minimal role.

IREP's formula for rule evaluation (Fürnkranz and Widmer, 1994) is:

$$(p \ + N - n) \ / \ (P + N)$$

RIPPER (Cohen, 1995) uses a slight modification of the above formula :

$$(p \ - n) \ / \ (P + N)$$

Since P and N are constant for a given problem, a rule deemed preferable by IREP will also be preferred by RIPPER. Thus, these two measures produce the same ranking; in comparing different measures, we therefore only show RIPPER's rankings below. One can also notice that RIPPER evaluation function returns a value the equal to half of the PROMISE value when $P = N$ and p exceeds n. Thus, in such cases, the RIPPER ranking is same as the PROMISE Ranking. Indeed, in the examples shown below that have equal P and N, the two methods provide identical rule rankings.

We compared these above methods on three datasets, each consisting of 1000 training examples. Dataset A has 20% positive and 80% negative examples, Dataset B has 50% positive and negative examples, and Dataset C has 80% positive examples and 20% of negative examples. In each dataset, rules with different completeness and consistency levels were compared using the following criteria: Information Gain, PROMISE, the original CN2 evaluation method, RIPPER, Q(0), Q(.25), Q(.5), Q(.75), and Q(1). Results are shown in Table 1. The leftmost column identifies the dataset, the next two give the numbers of examples of each class covered by a hypothetical rule, and the remaining columns list the evaluations and ranks on the dataset of the rules by the various methods, where 1 indicates the best ranked rules and 7 indicates the worst. There is, of course, no one answer regarding which ranking is superior. It should be noted, however, that by modifying the Q weights, one can tailor the rule evaluation criterion according to the problem at hand.

Table 1. Rule Evaluations by Different Methods

Data	Pos	Neg	Info Gain		PROMISE		CN2		RIPPRR		Q(0)		Q(.25)		Q(.5)		Q(.75)		Q(1)	
Set			*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*	*V*	*R*
A	50	5	.10	**7**	.24	**7**	.44	**4**	.05	**7**	.89	**4**	.65	**7**	.47	**7**	.34	**7**	.25	**6**
	50	0	.12	**6**	.25	**6**	0	**1**	.05	**6**	1	**1**	.71	**6**	.5	**6**	.35	**6**	.25	**6**
200	200	5	.69	**1**	.99	**1**	.17	**2**	.20	**1**	.97	**2**	.98	**1**	.99	**1**	.99	**1**	1	**1**
pos	150	10	.39	**2**	.74	**2**	.34	**3**	.14	**2**	.92	**3**	.88	**2**	.83	**2**	.79	**2**	.75	**2**
	150	30	.33	**3**	.71	**3**	.65	**6**	.12	**3**	.79	**6**	78	**3**	.77	**3**	.76	**3**	.75	**2**
800	100	15	.21	**5**	.48	**5**	.56	**5**	.09	**5**	.84	**5**	.74	**4**	.65	**4**	.57	**5**	.5	**5**
neg	120	25	.24	**4**	.57	**4**	.66	**7**	.10	**4**	.78	**7**	.73	**5**	.69	**5**	.64	**4**	.6	**4**
B	50	5	.03	**7**	.09	**7**	.44	**3**	.05	**7**	.82	**3**	.48	**7**	.29	**7**	.17	**7**	.1	**7**
	250	25	.21	**6**	.45	**5**	.44	**3**	.23	**5**	.82	**3**	.72	**5**	.64	**5**	.57	**5**	.5	**5**
500	500	50	.76	**1**	.9	**1**	.44	**3**	.45	**1**	.82	**3**	.86	**1**	.91	**1**	.95	**1**	1	**1**
pos	500	150	.49	**2**	.7	**3**	.78	**7**	.35	**3**	.54	**7**	.63	**6**	.73	**4**	.86	**2**	1	**1**
	200	5	.21	**5**	.39	**6**	.17	**1**	.20	**6**	.95	**1**	.77	**4**	.62	**6**	.5	**6**	.4	**6**
500	400	35	.44	**3**	.73	**2**	.40	**2**	.37	**2**	.84	**2**	.83	**2**	.82	**2**	.81	**3**	.8	**3**
neg	400	55	.38	**4**	.69	**4**	.53	**6**	.35	**4**	.76	**6**	.77	**3**	.78	**3**	.79	**4**	.8	**3**
C	50	5	.004	**7**	0	–	.05	**3**	.44	**7**	.55	**3**	.32	**6**	.18	**6**	.11	**6**	.06	**7**
	250	25	.02	**5**	0	–	.44	**3**	.23	**5**	.55	**3**	.47	**4**	.41	**5**	.36	**4**	.31	**5**
800	500	50	.07	**1**	0	–	.44	**3**	.45	**1**	.55	**3**	.56	**3**	.58	**1**	.60	**1**	.63	**1**
pos	500	150	.01	**6**	0	–	.78	**7**	.35	**3**	< 0	**7**	< 0	**7**	< 0	**7**	< 0	**7**	.63	**1**
	200	5	.05	**3**	0	–	.17	**1**	.20	**6**	.88	**1**	.64	**1**	.47	**3**	.34	**5**	.25	**6**
200	400	35	.05	**2**	0	–	.40	**2**	.37	**2**	.6	**2**	.57	**2**	.55	**2**	.52	**2**	.5	**3**
neg	400	55	.02	**4**	0	–	.53	**6**	.35	**4**	.4	**6**	.42	**5**	.44	**4**	.47	**3**	.5	**3**

Columns labeled *V* indicate raw value.
Columns labeled *R* indicate rank assigned by the given evaluation method in the given dataset

6 Implementation of the Q(*w*) Method in AQ18

The AQ18 learning system (Michalski, 1999) can operate in one of two modes: the standard or "noisy" mode, which relaxes the rule consistency requirement, and seeks rules with the highest rank on the LEF criterion, and the special or "no-noise" mode, which accepts only fully consistent rules, and creates a complete cover.

To implement the "noisy" mode, the Q(w) evaluation method has been employed in two places: one—during a multi-step rule growing process (*star generation*), in which the system repeatedly selects the best set of candidate rules for the next step of rule specialization, and second—during the completion of the process (*star termination*), in which the best rule is determined and submitted for output. The user may select a Q(*w*) weight for the program to use, or AQ18 will apply a default weight of 0.5 (equal emphasis on completeness and consistency).

During star generation, AQ18 uses a beam search strategy to find the "best" generalizations of a "seed" example by repeated applications of the "extension-against" generalization rule (Michalski, 1983). In the "noisy" mode, the system determines the Q(*w*) value of the generated rules after each extension-against operation; the rules with Q(*w*) value lower than that of the parent rule (the rule from which they were generated through specialization), are discarded. If that is the case for all rules stemming from a given parent rule, the parent rule is retained instead; this operation is functionally equivalent to considering the negative example extended against as noise.

In order to speed up the star generation, the user may specify a time-out threshold on the extension-against process. If after a given number of consecutive extensions, there has been no further improvement in rule quality Q(*w*), the system considers the current ruleset of sufficient quality, and terminates the extension process, thus ignoring any remaining unexamined negative examples.

In the star termination step (i.e., after the last extension-against operation), candidate rules are generalized in various ways to determine if the resulting rules have a higher Q(*w*) value. This optimization may introduce additional inconsistency, but makes possible that a lower quality rule may overtake initially higher quality ones if it produces superior generalizations. The best resulting is selected for output through the normal LEF process.

In the process of attempting to improve a rule, AQ18 applies a hill-climbing method. It generalizes the rule by extending the reference in each of its component conditions, then selects the highest-quality rule from among those generalizations, until no generalization creates further improvement. For conditions with nominal attributes, the only generalization rule (operator) considered is *condition dropping*. For conditions with linear attributes (rank, interval, cyclic, or continuous), the system applies the condition dropping, and the *extending* or *closing the interval* generation rules. For conditions with structured (hierarchically ordered) attributes, the system applies the condition dropping operator and the *generalization tree climbing* operator (Michalski, 1983).

Table 2. Effects of different generalization operators on the base rule:
[*color* = red v blue] & [*length* = 2 v 5] & [*animal_type* = dog v lion v bat]

Generalization Rule	Resulting Rule
Dropping condition (applied to nominal attributes)	[*length* = 2..4 v 8] & [*animal_type* = dog v lion v bat]
Dropping condition (applied to linear attributes)	[*color* = red v blue] & [*animal_type* = dog v lion v bat]
Extending interval (applies to linear attributes only)	[*color* = red v blue] & [*length* = 2..6 v 8] & [*animal_type* = dog v lion v bat]
Closing interval (applies to linear attributes only)	[*color* = red v blue] & [*length* = 2..8] & [*animal_type* = dog v lion v bat]
Dropping condition (applied to structured attributes)	[*color* = red v blue] & [*length* = 2..4 v 8]

Climbing generalization tree (applies to structured attribute only)	[*color* = red v blue] & [*length* = 2..4 v 8] & [*animal_type* = mammal]

Examples of each of the application of these generalization rules to the base rule [*color* = red v blue] & [*length* = 2..4 v 8]& [*animal_type* = dog v lion v bat] are presented in Table 2. In the base rule, *color* is a nominal attribute, *animal_type* is a structured attribute, and *length* is a linear attribute.

7 Summary

This paper introduced the measure of consistency gain, and presented a method for integrating it with completeness into a general and flexible measure of rule quality. The proposed Q(*w*) measure can be specialized to different specific formulae that weigh differently the two measures. Experiments have shown that by varying the *w* parameter, one can obtain different measures of rule quality, while machine learning programs have traditionally had inflexible criteria with respect to rule completeness and consistency.

Additionally, the Q(*w*) measure does not have to be used as a single measure, but can be employed as one of multiple criteria in the lexicographical evaluation functional (LEF). Through a LEF, one may thus optimize a rule learning process according to many different criteria. A planned future research topic involves the quantification of rule simplicity and its incorporation into a description simplicity measure.

The Q(*w*) measure has been implemented in AQ18 during the star generation and star termination processes (when generating a set of rules covering a specific seed example). In addition to handling inconsistency, the AQ18 program also includes a mechanism for generating incomplete rulesets without the computational overhead of generating and then truncating complete rulesets (Kaufman and Michalski, 1999). Such rulesets have the advantage of not containing many spurious rules.

We have also introduced a mechanism especially useful for data mining applications, in which AQ18 determines when some negative examples should be ignored as noise. Such determinations have resulted in rules with substantially higher completeness, at a small cost to consistency, while reducing the time required for learning.

Acknowledgments

The authors thank Qi Zhang for his comments and criticism on the ideas presented in earlier versions of this paper. This research was conducted in the Machine Learning and Inference Laboratory at George Mason University. The Laboratory's activities have been supported in part by the National Science Foundation under Grants No. IIS-9904078 and IRI-9510644, in part by the Defense Advanced Research Projects Agency under Grant No. F49620-95-1-0462 administered by the Air Force Office of Scientific Research, and in part by the Office of Naval Research under Grant No. N00014-91-J-1351.

References

Baim, P.W., The PROMISE Method for Selecting Most Relevant Attributes for Inductive Learning Systems. Report No. UIUCDCS-F-82-898. Department of Computer Science, University of Illinois, Urbana (1982).

Bergadano, F., Matwin S., Michalski, R.S. and Zhang, J., Learning Two-tiered Descriptions of Flexible Concepts: The POSEJDON System. *Machine Learning* 8, (1992) 5-43.

Bruha, I, Quality of Decision Rules: Definitions and Classification Schemes for Multiple Rules, in Nakhaeizadeh, G. and Taylor, C.C. (eds.) *Machine Learning and Statistics, The Interface*, New York: John Wiley & Sons., Inc. (1997) 107-131.

Clark, P. and Boswell, R. Rule Induction with CN2: Some Recent Improvements. in Kodratoff, Y. (ed.), *Proceedings of the Fifth European Working Session on Learning (EWSL-91)*, Berlin: Springer-Verlag (1991) 151-163.

Clark, P. and Niblett, T. The CN2 Induction Algorithm. *Machine Learning* 3 (1989) 261-283.

Cohen, W. Fast Effective Rule Induction. *Proceedings of the 12th International Conference on Machine Learning* (1995).

Fürnkranz, J. and Widmer, G. Incremental Reduced Error Pruning. *Proceedings of the 11th International Conference on Machine Learning* (1994).

Kaufman, K.A. INLEN: A Methodology and Integrated System for Knowledge Discovery in Databases. Ph.D. dissertation, George Mason University, Fairfax, VA (1997).

Kaufman, K.A. and Michalski, R.S. Learning in an Inconsistent World: Rule Selection in AQ18. *Reports of the Machine Learning and Inference Laboratory*. MLI 99-1. George Mason University, Fairfax, VA (1999).

Michalski, R.S. A Theory and Methodology of Inductive Learning. In: Michalski, R.S. Carbonell, J.G., Mitchell, T.M.

(eds.), *Machine Learning: An Artificial Intelligence Approach*. Tioga Publishing, Palo Alto, CA (1983) 83-129.

Michalski, R.S. NATURAL INDUCTION: A Theory and Methodology of the STAR Approach to Machine Learning and Data Mining. *Reports of the Machine Learning and Inference Laboratory*. George Mason University (1999).

Piatetsky-Shapiro, G. Discovery, Analysis, and Presentation of Strong Rules. In: Piatetsky-Shapiro, G., Frawley, W. (eds.): *Knowledge Discovery in Databases*. AAAI Press, Menlo Park, CA, (1991) 229-248.

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