Eleventh International Symposium on Methodologies for Intelligent Systems,
Warsaw, pp. 411-419, 1999

Learning From Inconsistent and Noisy Data: The AQ18 Approach[*]

Kenneth A. Kaufman and Ryszard S. Michalski*

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

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

Abstract. In concept learning or data mining tasks, the learner is typically faced with a choice of many possible hypotheses characterizing the data. If one can assume that the training data are noise-free, then the generated hypothesis should be complete and consistent with regard to the data. In real-world problems, however, data are often noisy, and an insistence on full completeness and consistency is no longer valid. The problem then is to determine a hypothesis that represents the "best" trade-off between completeness and consistency. This paper presents an approach to this problem in which a learner seeks rules optimizing a *description quality criterion* that combines completeness and *consistency gain*, a measure based on consistency that reflects the rule's benefit. The method has been implemented in the AQ18 learning and data mining system and compared to several other methods. Experiments have indicated the flexibility and power of the proposed method.

1 Introduction

In concept learning and data mining tasks, a typical objective is to determine a general hypothesis characterizing training instances that will classify future instances as correctly as possible. For any non-trivial generalization problem, one can usually generate many plausible hypotheses that are complete and consistent with regard to the input data. If one can assume that the training data contain no noise, hypothesis consistency and completeness are justifiable preconditions for admissibility, but in real-world applications, data are often noisy and/or inconsistent; the completeness and consistency criteria are therefore no longer essential. One may then seek a hypothesis that is maximally consistent at the expense of completeness, maximally complete at the expense of inconsistency, or representing some combination of the two criteria.

The problem then arises as to what kind of trade-off should be chosen between completeness and consistency, or, more generally, what criteria should be used to guide the learner in problems with noisy and/or inconsistent data? To illustrate this

[*] This research was supported in part by the National Science Foundation under grants No. NSF 9904078 and IRI-9510644.

problem, suppose that a training set contains 1000 positive and 1000 negative examples of a concept to be learned. Suppose further that a learning system generated two hypotheses, one covering 600 positive examples and 2 negative examples, and the other covering 950 positive examples and 20 negative examples. Which is better?

Clearly, the choice of which hypothesis to select is not domain-independent. In some domains, the first hypothesis may be preferred because it represents a more consistent pattern. In other domains, the second hypothesis may be viewed as better, because it represents a more dominant pattern.

This paper explores issues related to the trade-off between hypothesis completeness and consistency in the case of noisy and/or inconsistent training data, and proposes a single *description quality measure*, which reflects this trade-off. This measure can be combined with a *description simplicity measure* (a reciprocal of description complexity) into a *general description quality measure* using the *lexicographical evaluation functional* (Section 2). A learning or pattern discovery process is presented as a search for a hypothesis that maximizes a given description quality measure. These measures have been implemented in the AQ18 rule learning system [9].

2 Multicriterion Ranking of Hypotheses

We explore the issue of choosing the best hypothesis or pattern in the context of a progressive covering (a.k.a. separate-and-conquer) approach to concept learning. In this approach, a complete concept description (a ruleset) is created by sequentially determining decision rules. If the training data contain errors or inconsistency, some degree of inconsistency and incompleteness of the rules may not only be acceptable, but also desirable (e.g., [2]).

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 in the positive class). Therefore, a simple criterion for evaluating rule quality can be some function of the completeness and the consistency. Different forms of such a criterion have been presented in the literature, and appear to be adequate for problems on which they have been tested. It is, however, unrealistic to expect that any single form will fit all practical problems. For different problems, a different criterion of rule optimality may lead to better performance.

This paper views a learning process as a problem of constructing a description that optimizes a description quality criterion. This view has been implemented in the AQ18 rule learning system. To tailor the description quality measure to the problem at hand, AQ18 offers a user a range of *elementary criteria*, each representing one aspect of the hypothesis being learned. A subset of these criteria is selected from a menu of available criteria, and assembled into a *lexicographic evaluation functional* (LEF) [8], defined by a sequence:

$$<(c_1, \tau_1), (c_2, \tau_2), \ldots, (c_n, \tau_n)> . \qquad \textbf{(1)}$$

where c_i represents the *i*th elementary criterion, and τ_i is the *tolerance* associated with c_i. The latter defines the range (either absolute or relative) within which a candidate rule's c_i evaluation value can deviate from the best evaluation value of this criterion in the current set of rules. Criteria in LEF are applied sequentially to the set of hypotheses. A hypothesis passes a given criterion if its evaluation on this criterion is not outside the range as defined by the tolerance from the maximum value among the hypotheses in consideration. To illustrate LEF, let us assume that S is a set of hypotheses, and LEF consists just two elementary criteria, first to maximize the completeness, and second to maximize consistency. Let us assume further that hypotheses with completeness less than 10% below the maximum completeness achieved by any single rule in S is still acceptable, and that if two or more hypotheses satisfy this criterion, the one with the highest consistency is to be selected. Such a rule selection process can be specified by the following LEF:

$$\text{LEF} = <(\text{completeness}, 10\%), (\text{consistency}, 0\%)> . \qquad (\mathbf{2})$$

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 both very simple and very efficient; thus it is applicable where there is a very large number of candidate hypotheses. In this paper, we combine completeness and consistency gain (a measure of a rule's improvement in accuracy over a random guess) into one numerical measure, which serves as an elementary description quality criterion. Through LEFs, this criterion can be combined with a measure of description simplicity into one general description quality measure.

3 Completeness, Consistency, and Consistency Gain

An important role of a description is to classify future, unknown cases. Therefore, a useful measure of description quality is its *testing accuracy*, that is, its accuracy of classifying testing examples, as opposed to training examples. During a learning process, testing examples, by definition, are not being used; one therefore needs a criterion that is a good approximator of the testing accuracy, but uses only training examples. In order to propose such a measure, some notation needs to be introduced.

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

Let us return to the question posed in the Introduction as to which is preferable: a rule with 60% completeness and 99.8% consistency, or a rule with 95% completeness and 98% consistency. As indicated earlier, the answer depends on the problem at hand. In some application domains (e.g., science), a rule (law) must be consistent with all the data, unless some of the data are found erroneous. In other applications (e.g., data mining), one may seek strong patterns that hold frequently, but not always. Thus, there is no single measure of rule quality that will be good for all problems. We instead seek a flexible measure that can be easily changed to fit a given problem.

How can one define such a measure of rule quality? Let us first look at a measure based on the *information gain* criterion, which is used widely as a method for selecting attributes in decision tree learning (e.g. [11]). 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:

$$\mathrm{Gain(R)} = \mathrm{Info(E)} - \mathrm{Info_R(E)}\ . \qquad (\mathbf{3})$$

where Info(E) is the expected information from defining the class of an example in E and $\mathrm{Info_R(E)}$ is the expected information after partitioning the space using rule R.

Information gain has one major disadvantage as a rule evaluator, namely, it relies not only on a rule's informativeness, but also on the informativeness of the complement of the rule. This is undesirable, especially in situations with more than two decision classes. Clearly, 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 disadvantage is that it does not allow one to vary the importance of consistency in relation to completeness.

Let us also observe that relative frequency of positive and negative examples in the training set of a given class should also 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 uninformative if P was very large and N was very small.

The distribution of positive and negative examples in the 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 $p / (p + n)$. The difference between these values, $(p /(p + n)) - (P /(P + N))$ (the gain of consistency over the dataset's distribution as a whole), can be normalized by dividing it by $N /(P + N)$, 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 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 has a negative benefit. Reorganizing the normalization term, we define the *consistency gain* of a rule R, consig(R), as:

$$\mathrm{consig(R)} = ((p / (p + n)) - (P /(P + N)))\ *\ (P + N) /N\ . \qquad (\mathbf{4})$$

4 A Formula for 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 a higher compl(R) and a higher consig(R) is more desirable than a rule with lower measures. A rule with either compl(R) or consig(R) equal to 0 is worthless. It makes sense, therefore, to define a rule quality measure that evaluates to 1 when both of these components have value 1, and 0 when either is equal to 0.

A simple way to achieve such a behavior is to define rule quality as a product of compl(R) and consig(R). Such a formula, however, does 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. The *description quality*, Q(R, *w*) with weight *w*, or just Q(*w*) if the rule R is implied, can thus be written:

$$\mathrm{Q}(\mathrm{R}, w) = \mathrm{compl}(\mathrm{R})^{w} * \mathrm{consig}(\mathrm{R})^{(1 - w)} . \qquad (\mathbf{5})$$

By changing parameter *w*, one can change the relative importance of the completeness and the consistency gain to fit a given problem. The Q(*w*) definition satisfies all of the criteria regarding desirable features of a rule evaluation function given in [10]:

(1) 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 else 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 else being equal, rule quality 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.

The next section compares the proposed Q(*w*) rule evaluation method with other methods, and Section 6 discusses its implementation in AQ18.

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 machine learning systems, we performed a series of experiments on different datasets. In the experiments we used the Q(*w*) method with different weights *w*, the information gain criterion (Section 3), the PROMISE method [1][6], and the methods employed in CN2 [3], IREP [5], and RIPPER [4] rule 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 [1][6] was developed to evaluate the quality of attributes. It can be used, however, for rule evaluation by considering a rule to be a binary attribute that splits the space into the part covered by the rule and the part not covered by it. 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 $(p / P) + (M / T) - 1$. When M is based on the positive class (when $P - p > N - n$), PROMISE returns a value of zero. Hence, PROMISE is not useful for measuring rule quality in domains in which the total number of positive examples significantly outnumbers the total number of negative ones. Note also that when p exceeds n and $P = N$, the PROMISE value reduces to:

$$(p - n) / P \ . \qquad \textbf{(6)}$$

To see this, note that when $P = N$, $(p / P) + ((N - n) / N) - 1$ can be transformed into $(p / P) + ((P - n) / P) - 1$,which is equivalent to (6).

CN2 [3] 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 minimizes the expression:

$$-((p /(p + n)) \log_2(p /(p + n)) + (n /(p + n)) \log_2(n /(p + n))) \ . \qquad \textbf{(7)}$$

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 indistinguishable from another rule that covers 50,000 positive and 5000 negative examples. Although (7) has a somewhat different form than (4), CN2's rule evaluation can be expected to be similar to Q(0) (consistency gain only). Indeed, in the examples shown below, the two methods provide identical rule rankings.

IREP's formula for rule evaluation [5] is:

$$(p + N - n) / (P + N) \ . \qquad \textbf{(8)}$$

RIPPER [4] uses a slight modification of the above formula:

$$(p - n) / (P + N) \ . \qquad \textbf{(9)}$$

Since P and N are constant for a given problem, a rule deemed preferable by IREP will also be preferred by RIPPER. We therefore only show RIPPER's rankings below. Comparing (9) to (6), one can notice that RIPPER evaluation function returns a value equal to half of the PROMISE value when $P = N$ and p exceeds n. Thus, in such cases, the RIPPER ranking is the same as the PROMISE ranking.

We compared the 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 ranked using the following criteria: Information Gain, PROMISE, RIPPER, CN2, 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 positive and negative examples of each class covered by a hypothetical rule, and the remaining columns list the 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 weight, one can obtain different rule rankings, among them those identical to rankings by some other methods.

Table 1. Rule Rankings by Different Methods

Data Set	Pos	Neg	RANKS								
			I. G.	PROM	CN2	RIPP	Q(0)	Q(.25)	Q(.5)	Q(.75)	Q(1)
A	50	5	7	7	4	7	4	7	7	7	6
	50	0	6	6	1	6	1	6	6	6	6
200	200	5	1	1	2	1	2	1	1	1	1
pos	150	10	2	2	3	2	3	2	2	2	2
	150	30	3	3	6	3	6	3	3	3	2
800	100	15	5	5	5	5	5	4	4	5	5
neg	120	25	4	4	7	4	7	5	5	4	4
B	50	5	7	7	3	7	3	7	7	7	7
	250	25	6	5	3	5	3	5	5	5	5
500	500	50	1	1	3	1	3	1	1	1	1
pos	500	150	2	3	7	3	7	6	4	2	1
	200	5	5	6	1	6	1	4	6	6	6
500	400	35	3	2	2	2	2	2	2	3	3
neg	400	55	4	4	6	4	6	3	3	4	3
C	50	5	7	–	3	7	3	6	6	6	7
	250	25	5	–	3	5	3	2	5	4	5
800	500	50	1	–	3	1	3	3	1	1	1
pos	500	150	6	–	7	3	7	7	7	7	1
	200	5	3	–	1	6	1	1	3	5	6
200	400	35	2	–	2	2	2	2	2	2	3
neg	400	55	4	–	6	4	6	5	4	3	3

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

The AQ18 learning system [9] operates in two modes: the “noisy” mode (default), which relaxes the rule consistency requirement, and the “pure” mode, which accepts only fully consistent rules, and creates a complete cover. The “noisy” mode was implemented by employing the Q(*w*) evaluation method in two places: one—during a multi-step rule growing process (*star generation*), which 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*), which determines the best rule for output. The default value of *w* in Q(*w*) is 0.5.

During star generation, AQ18 uses a beam search strategy to find the “best” generalizations of a “seed” example by a repeated application of the “extension-against" generalization rule [8]. In the "noisy" mode, the system determines the Q value of the generated rules after each extension-against operation; the rules with Q(w) lower than that of the parent rule (the rule from which they were generated through specialization), are discarded. If the Q(w) value of all rules stemming from a given parent rule is lower, the parent rule is retained instead; this operation is functionally equivalent to discarding the negative example extended against as noise.

In the star termination step (i.e., after the last extension-against operation), the candidate rules are generalized additionally to determine if a resulting rule has a higher Q(w) value. This chosen generalization operator may vary from among the *condition dropping*, *generalization tree climbing*, *interval extending* and the *interval closing* generalization operators depending on the type of the attributes in the rules, as described in [8]. As a result of this hill-climbing optimization step, the best rule in the resulting star is selected for output through the LEF process. Examples of the application of these generalization rules may be found in [7].

Experiments with AQ18 in the "noisy" mode revealed a difference in performance when applying different coverage criteria available in AQ18 [7]. The available criteria for defining LEF include maximizing two measures of coverage, one being the total number of positive examples covered by a rule, and the other being the coverage of positive events that have not been covered by any previously generated and retained rule for that class. The rulesets generated using the latter criterion were found to be superior, hence this criterion was adapted as a default. This idea was subsequently applied to the Q(w) measure, with a hope that it may lead to a reduction of the likelihood that a rule may be supplanted by an inferior generalization. Thus we have replaced the original completeness measure, compl(R), by ncompl(R), defined as:

$$\text{ncompl(R)} = p_{new} \,/\, P \,. \qquad \textbf{(10)}$$

where p_{new} is the number of positive examples newly covered by the candidate rule.

When AQ18 was applied to a medical dataset consisting of 32 attributes (all but 5 of which were of Boolean type) describing individuals' lifestyles and whether or not they had been diagnosed with various diseases, experiments were run with three different Q(w) weights: 0.25, 0.5, and 0.75. For the decision class *arthritis*, the training set (7411 examples) had a highly unbalanced distribution of positive examples (P = 1171 or 16%) and negative examples (N = 6240 or 84%). The highest coverage rules learned by AQ18 for each w value in Q(w) were:

For w = 0.25: [d =*arthritis*] <= [education ≤ vocational] &
[years in neighborhood > 26] & [rotundity ≥ low] &
[tuberculosis = no]: p = 271, n = 903, Q = 0.111011

For w = 0.5: [d =*arthritis*] <= [high blood pressure = yes] &
[education ≤ college grad] p = 355, n = 1332, Q = 0.136516

For w = 0.75: [d =*arthritis*] <= [education ≤ college grad] p = 940, n = 4529,
Q = 0.303989

Because increasing weight w in the Q(w) measure gives higher importance to completeness in comparison to consistency, we see increasing p values (completeness) and decreasing values of consistency (p /(p + n)) in the rules above.

7 Summary

This paper introduced consistency gain as a measure of a rule's level of improvement in accuracy over a random guess, and presented a method for integrating it with the completeness criterion into one general and flexible numerical measure for guiding the process of generating inductive descriptions. This measure, Q(*w*), allows one to change the relative importance of description coverage (completeness) in relation to consistency by varying the *w* parameter. The method has been implemented in AQ18 learning system and tested on a range of problems. A topic for future research and experimentation involves the quantification and integration of description simplicity into the description quality formula.

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

References

1. 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)
2. Bergadano, F., Matwin, S., Michalski R.S., Zhang, J.: Learning Two-tiered Descriptions of Flexible Concepts: The POSEIDON System. Machine Learning 8 (1992) 5-43
3. Clark, P., Niblett, T.: The CN2 Induction Algorithm. Machine Learning 3 (1989) 261-283
4. Cohen, W.: Fast Effective Rule Induction. Proc. 12th Intl. Conf. on Machine Learning (1995)
5. Fürnkranz, J., Widmer, G.: Incremental Reduced Error Pruning. Proc. 11th Intl. Conf. on Machine Learning (1994)
6. Kaufman, K.A.: INLEN: A Methodology and Integrated System for Knowledge Discovery in Databases. Ph.D. diss. George Mason University, Fairfax, VA (1997)
7. Kaufman, K.A., 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)
8. 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
9. Michalski, R.S.: NATURAL INDUCTION: A Theory and Methodology of the STAR Approach to Symbolic Learning and Its Application to Machine Learning and Data Mining. *Reports of the Machine Learning and Inference Laboratory.* George Mason University (1999)
10. 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
11. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1 (1986) 81-106