

The RISE System: Conquering Without Separating

Pedro Domingos

Department of Information and Computer Science

University of California, Irvine

Irvine, California 92717, U.S.A.

Abstract

Current rule induction systems (e.g. CN2) typically rely on a “separate and conquer” strategy, learning each rule only from still-uncovered examples. This results in a dwindling number of examples being available for learning successive rules, adversely affecting the system’s accuracy. An alternative is to learn all rules simultaneously, using the entire training set for each. This approach is implemented in the RISE 1.0 system. Empirical comparison of RISE with CN2 suggests that “conquering without separating” performs similarly to its counterpart in simple domains, but achieves increasingly substantial gains in accuracy as the domain difficulty grows.

1 Introduction and motivation

Current machine learning approaches to the induction of concept definitions from examples fall mainly into two categories: “divide and conquer” [8] and “separate and conquer” [6, 3]. The former recursively partition the instance space until regions of roughly uniform class membership are obtained. The latter induce one rule at a time, removing the newly covered examples from the training set after each step. Both suffer from the “splintering problem”: as induction progresses, the size of the available sample dwindles, resulting in decisions being made with less and less statistical support. Statistical anomalies become harder to weed out, and noise sensitivity increases. As a result, it may not be possible to reliably induce some rules to their full length, causing either overly general or incorrect rules to be produced, and negatively affecting accuracy.

A related problem, first identified by Holte and coworkers [5], is that of small disjuncts. While covering relatively few examples, small disjuncts tend to be responsible for a disproportionate share of the clas-

sification errors committed. Many of these disjuncts undoubtedly represent actual small disjuncts in the domain. The small size of others, however, may be an artifact of the “separate and conquer” strategy: since each such disjunct only attempts to cover a subset of the examples comprising the corresponding target disjunct (the rest having been removed because they were also covered by previously induced disjuncts), it may come out smaller than it should actually be.

All of these problems are alleviated if each rule is learned taking into consideration the entire training set, i.e. if the induction algorithm “conquers without separating.” This is the approach taken in the RISE 1.0 system (Rule Induction from a Set of Examples). In the remainder of this paper RISE is described and then empirically compared with CN2 [3].

2 The RISE algorithm

The RISE 1.0 system induces a set of rules (the hypothesis) from a set of examples (the training set). Rules and examples have a similar representation: a conjunction of attribute values (the antecedents), and a predicted class. Each attribute may be nominal or numeric; numeric antecedents are ranges represented by their limits (which coincide, in the case of an example). Antecedents of both types can take on the special value *, which stands for “any,” and is equivalent to dropping the antecedent.

The RISE algorithm conducts a batch, hill-climbing, specific-to-general search through the space of rule sets, and can be summarized by the pseudo-code presented in Table 1, where $H(\cdot)$ is the heuristic evaluation function. A nominal antecedent is generalized by assigning it the value *, and a numeric antecedent is generalized by extending its range to include the next higher or the next lower value outside it (both alternatives are tried). These values are taken from an ordered list of the midpoints between consecutive

Input: ES is the training set.

Procedure RISE (ES)

Let RS be ES .

Compute $H(RS)$.

Repeat

 For each rule in RS ,

 Let R be the current version of the rule.

 For each one of R 's antecedents A ,

 Try generalizing A , and

 Compute the resulting $H(RS')$.

 Select the A whose generalization results in the greatest $H(RS')$.

 If this $H(RS') \geq H(RS)$,

 Then Generalize A ,

 If R' is identical to another rule in RS ,

 Then delete R' from RS ,

 Replace $H(RS)$ with the max. $H(RS')$,

Until no increase in H is obtained.

Return RS .

Table 1: The RISE algorithm.

observed values of the attribute. Missing values are taken to be *.

RISE currently considers the heuristic value of a rule set to be the sum of the heuristic values of the rules that comprise it, and the heuristic value of an individual rule to be of the form:

$$h(r) = p(r) - (1 - \eta) S n(r)$$

where r is the rule, $p(r)$ is the number of examples covered by the rule whose consequents are identical to the rule's ("positive" examples), $n(r)$ is the number of examples covered by the rule whose consequents are different from the rule's ("negative" ones), S is the sample (training set) size, and $\eta \in [0, 1]$ is a noise tolerance coefficient.

The idea behind η is that in noisier domains rules should be allowed to cover a greater proportion of negative examples; in noiseless domains this proportion should be 0. The ideal "coarseness" of the rules is domain-dependent and there is no universally good level for it; hence the use of a parameter. If $\eta = 0$, the S factor in h ensures that no number of positive examples will make up for covering even a single negative example, and RISE is thus completely noise intolerant, i.e. any generalization that covers one or more neg-

ative examples will necessarily be rejected. If $\eta = 1$ RISE is completely noise tolerant, i.e. all rules will be generalized to a null left-hand side and the most frequent class will always be selected by the classification procedure below. In practice, depending on domain characteristics, the useful range for η will be more restricted. A notable point is $\eta = 1 - \frac{1}{S}$; for this value the heuristic becomes simply $h(r) = p(r) - n(r)$.

To classify a test example RISE matches it with every rule in the induced rule set, using weighted voting among the successful rules to resolve conflicts. Currently each rule's weight is simply its heuristic value, leading to a preference for rules that (1) cover many examples, i.e. have substantial statistical support, and (2) cover few examples that are not of the predicted class, i.e. are accurate classifiers. The test example is assigned to the most voted class.

When no rule matches, RISE defaults to a best-match policy, instead of the usual policy of selecting the most frequent class. With the current version of RISE and the domains used so far, however, no-match situations are extremely rare, so this feature has little impact and will not be discussed further here.

Simple computations show that RISE's worst-case time complexity is quadratic in the number of examples and cubic in the number of attributes, which is competitive with e.g. CN2. (Note that the computations in [3] are only for the basic step of the algorithm.) With numeric attributes, RISE's worst-case running time also depends linearly on the average number of observed values per attribute. See [4] for details.

3 Empirical evaluation

With the goal of empirically evaluating the usefulness of the "conquering without separating" strategy, RISE was compared with a current "separate-and-conquer" rule induction algorithm on a number of natural and artificial domains. CN2 [3] was chosen for this purpose because it is probably the most extensively evaluated noise-tolerant algorithm of this type. A recent, enhanced version was used [2]. All options were set so as to maximize similarity to RISE; all save the star size (set to 1) are the defaults. Star size 1 implements a hill-climbing search, losing some power relative to a beam one, but in practice this seldom decreases accuracy significantly; in fact, in some domains the results reported here for CN2 are the best ever.

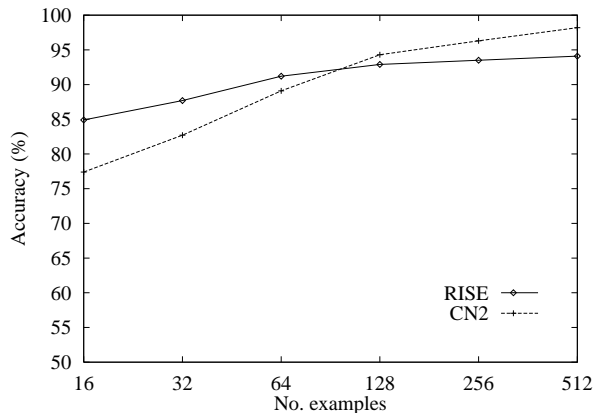


Figure 1: Performance on the task $Concept_1 = A \vee BC \vee DEF \vee GH IJ$.

3.1 Artificial domains

RISE and CN2 were tested on three boolean functions of varying degrees of difficulty according to the blurring measure [9]. Learning was carried out on sets of 16, 32, 64, 128, 256 and 512 examples, randomly chosen with uniform probability from the 1024 possible, and the induced rules were tested on the remaining examples. This procedure was repeated 20 times for each domain. The averaged results are presented graphically in Figs. 1-3. All nonzero differences in performance between RISE and CN2 are statistically significant at the 5% level, using a one-tailed paired t test.¹

Overall RISE and CN2 seem to perform similarly in the simpler case (Fig. 1), with RISE faring better when there are fewer examples and conversely when there are many, but RISE achieves consistently higher accuracy in the intermediate concept (Fig. 2), and by a wider margin. In the hardest concept (Fig. 3) this difference is even more pronounced, leading to the hypothesis that conquering without separating becomes more advantageous as domain difficulty increases.

3.2 Natural domains

Tests were conducted in 16 domains from the UCI repository [7] to determine if this behavior is observable in practical situations (see Table 2). The voting domain was tried in two forms: with and without the “physician fee freeze” attribute. Removing this attribute causes the difficulty of the domain to increase

¹Right tail if the difference was positive, left one if negative.

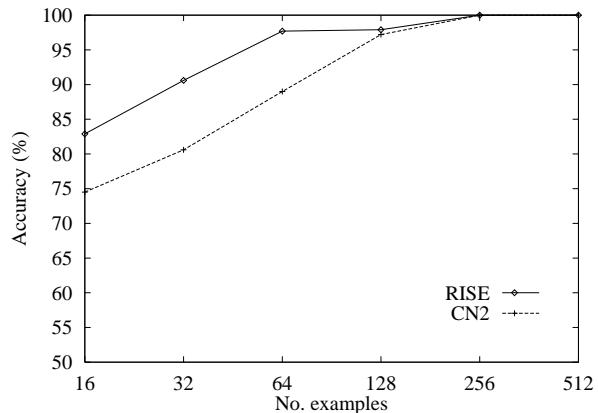


Figure 2: Performance on the task $Concept_2 = ABC \vee BCD \vee CDA \vee DAB$, with 6 irrelevant attributes.

substantially [1], making possible an evaluation of performance variation with difficulty while holding other factors constant.

Ten-fold cross-validation was performed for RISE and CN2, using the same training and test sets for the two at each step. The following policy was used to set RISE’s noise tolerance parameter η : in domains with inconsistent examples, i.e. domains that are guaranteed *a priori* to be noisy, it was set to $1 - \frac{1}{5}$, yielding $h = p - n$ as explained in a previous section; otherwise it was set to 0.² The results obtained are summarized in Table 2, where an asterisk indicates that the difference between RISE and CN2 was significant at the 5% level using a one-tailed paired t test.

The results here are not as clear; in many cases there is no significant difference between RISE and CN2. Two observations, however, support the hypothesis above that RISE improves relative to CN2 as the complexity of the domain increases. One comes from the voting domain, where CN2 performs significantly better in the “easy” version, but suffers a more pronounced drop when the “hard” one is tried, resulting in no significant difference in this case. The other observation comes from the trio of medical domains (lymphography, breast cancer, and primary tumor) where CN2 was originally tested [3] and that is probably the most widely used in the machine learning literature: there is no significant difference in the “easier” ones, but RISE is significantly better in the “harder” one (primary tumor). Overall RISE does better than CN2, both on average and in number of sig-

²Several other variations of RISE have been tested, often with superior results, but they are not reported here.

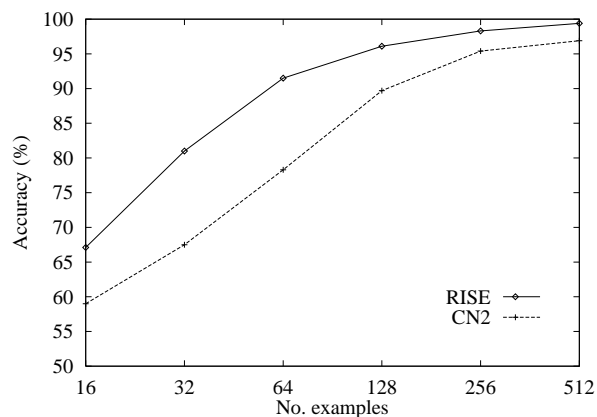


Figure 3: Performance on the task $Concept_3 = ABCD \vee BCDE \vee CDEA \vee DEAB \vee EABC$, with 5 irrelevant attributes.

nificant wins; this remains true even if the iris domain, where CN2’s performance was anomalously low (only the default rule was induced), is omitted.

4 Discussion

The preliminary results in artificial domains presented here suggest that “conquering without separating” and “separate-and-conquer” may perform similarly in simple domains, but the former may be substantially better in harder ones. This is tentatively confirmed by observations in natural domains. This behavior may be attributable to the fact that harder domains typically contain a greater number of disjuncts, each covering fewer examples, worsening the splintering and small disjuncts problems. Conversely, the presence of fewer large disjuncts reduces their masking effect on non-separating algorithms, improving their relative performance. See [4] for further discussion and future work.

Acknowledgments

This work was partly supported by JNICT/Programa Ciência and Fulbright scholarships. The author is grateful to Dennis Kibler and Mike Pazzani for many helpful comments and suggestions, to Peter Clark for supplying CN2, and to M. Zwitter and M. Soklic of the University Medical Centre, Ljubljana, for supplying the lymphography, breast cancer and primary tumor datasets.

Domain	RISE	CN2	Signif.
Breast cancer	69.9	71.6	
Chess endgames	99.1	95.4	*
Credit screening	80.6	80.2	
Voting records	90.8	96.1	*
Voting records modif.	89.2	90.6	
Contact lenses	68.3	65.0	
Hepatitis	79.4	80.7	
Iris	94.0	19.4	*
Labor negotiations	80.0	74.7	
Lung cancer	40.8	39.2	
Lymphography	77.0	81.6	
Pima diabetes	59.0	65.1	
Primary tumor	42.5	37.2	*
Shuttle landing	30.0	40.0	
Soybean	98.0	98.0	
Wine	82.1	39.9	*

Table 2: Results in domains from the UCI repository.

References

- [1] W. Buntine and T. Niblett, “A further comparison of splitting rules for decision tree induction,” *Machine Learning*, Vol. 8, pp. 75-86, 1992.
- [2] P. Clark and R. Boswell, “Rule induction with CN2: Some recent improvements,” *Proc. EWSL-91*, pp. 151-163, 1991.
- [3] P. Clark and T. Niblett, “The CN2 induction algorithm,” *Machine Learning*, Vol. 3, pp. 261-283, 1989.
- [4] P. Domingos, “Design and Evaluation of the RISE 1.0 Learning System”, Technical Report 94-34, UCI, Dept. ICS, Irvine, CA, 1994.
- [5] R. C. Holte, L. E. Acker, and B. W. Porter, “Concept learning and the problem of small disjuncts,” *Proc. IJCAI-89*, pp. 813-818, 1989.
- [6] R. S. Michalski, “A theory and methodology of inductive learning,” *Artificial Intelligence*, Vol. 20, pp. 111-161, 1983.
- [7] P. M. Murphy and D. W. Aha, UCI repository of machine learning databases, machine-readable data repository, UCI, Dept. ICS, Irvine, CA, 1992.
- [8] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, Vol. 1, pp. 81-106, 1986.
- [9] L. Rendell and H. Ragavan, “Improving the design of induction methods by analyzing algorithm functionality and data-based concept complexity,” *Proc. IJCAI-93*, pp. 952-958, 1993.