The RISE 2.0 System: A Case Study in Multistrategy Learning

Pedro Domingos
pedrod@ics.uci.edu

Technical Report 95-2

January 9, 1995

The RISE 2.0 System: A Case Study in Multistrategy Learning

Pedro Domingos
Department of Information and Computer Science
University of California, Irvine
Irvine, California 92717, U.S.A.
Tel. 714-856-0182
pedrod@ics.uci.edu

Abstract

Several well-developed approaches to inductive learning now exist, but each has specific limitations that are hard to overcome. Multi-strategy learning attempts to tackle this problem by combining multiple methods in one algorithm. This report describes a unification of two widely-used empirical approaches: rule induction and instance-based learning. In the new algorithm, instances are treated as maximally specific rules, and classification is performed using a best-match strategy. Rules are learned by gradually generalizing instances until no improvement in apparent accuracy is obtained. Theoretical analysis shows this approach to be efficient. It is implemented in the RISE 2.0 system. In an extensive empirical study, RISE consistently outperforms state-of-the-art representatives of both its parent approaches (PEBLS and CN2), as well as a decision tree learner (C4.5). Most significantly, in 15 of the domains studied, RISE achieves higher accuracy than the best of PEBLS and CN2, showing that a significant synergy can be obtained by combining multiple empirical methods.

Keywords: Concept learning, multi-strategy learning, rule induction, instance-based learning, nearest-neighbor classification, case-based reasoning

1 The empirical multi-strategy learning problem

Inductive learning is the creation of general concept or class descriptions from examples. Given a training set of preclassified examples, where each example (also called instance, observation or case) is described by a vector of feature or attribute values, the goal is to form a description that can be used to classify previously unseen examples with high accuracy. The last decade has witnessed renewed interest in this area, largely due to its relevance to the "knowledge acquisition bottleneck" problem: the costliest component in the creation and deployment of an expert system is the construction of the knowledge base, and if this construction can be partly automated by the use of induction techniques, the bottleneck will be greatly reduced. As a result, research in this field has blossomed, and several mature approaches to inductive learning are now available to the practitioner. These include induction of decision trees (Quinlan, 1986), rule induction (Michalski, 1983), instance-based learning (Aha *et al*, 1991), Bayesian classification (Buntine, 1989), back-propagation (Rumelhart *et al*, 1986), and genetic algorithms (Booker *et al*, 1989).

Empirical comparison of these different approaches in a variety of application domains has shown that each performs best in some, but not all, domains. This has been termed the "selective superiority" problem (Brodley, 1993), and presents a dilemma to the knowledge engineer approaching a new task: which induction paradigm should be used? One solution is to try each one in turn, and use cross-validation to choose the one that appears to perform best. This is a long and tedious process, especially considering the large number of algorithms and systems now available, and the fact that each typically has options and parameters that themselves need to be fine-tuned by cross-validation before the system can be said to be doing its "best."

Another approach, known as multi-strategy learning (Michalski and Tecuci, 1994), attempts to combine two or more different paradigms in a single algorithm. Most research in this area has been concerned with combining empirical (i.e. purely inductive) approaches with analytical ones (e.g. Ourston and Mooney, 1994; Towell *et al*, 1990; Pazzani and Kibler, 1992; Muggleton and Feng, 1990). The expression "empirical multi-strategy learning" will therefore be used to distinguish the case where all the components are empirical. Ideally, an empirical multi-strategy learning algorithm would always perform as well as the best of its "parents," obviating the need to try each one and simplifying the knowledge acquisition task. Even more ambitiously, there is hope that this combination of paradigms might produce synergistic effects (e.g. by allowing different types of frontiers in different areas of the instance space), leading to levels of accuracy that neither atomic approach by itself would be able to achieve. Indeed, in many application domains the accuracy of even the best methods is far from 100%, and the question of whether it can be improved, and if so how, is an open and important one.

Unfortunately, the success of this approach has so far been moderate. The resulting algorithms are prone to be cumbersome, and often achieve accuracies that lie

between those of their parents, instead of matching the highest (e.g. Brodley, 1993). Here a theoretical question arises. It is well known that no induction algorithm can be the best in all possible domains; each algorithm contains an explicit or implicit bias (Mitchell, 1980) that leads it to prefer certain generalizations over others, and it will be successful only insofar as this bias matches the characteristics of the application domain. Further, recent results (Schaffer, 1994) show that performance over the set of all possible domains is subject to a "conservation law": if one algorithm is better than another in some domains, then there are necessarily other domains in which this relationship is reversed. The average accuracy of an algorithm over all domains is a constant, independent of the algorithm. The results of e.g. Brodley (1993) seem to confirm this. Should we conclude, then, that empirical multi-strategy learning is doomed to failure?

Not necessarily. A distinction should be made between all the mathematically possible domains, which are simply a product of the representation languages used, and the domains that occur in the real world, and are therefore the ones of primary interest. Without doubt there are many domains in the former set that are not in the latter, and average accuracy in the real-world domains can be increased at the expense of accuracy in the domains that never occur in practice. Indeed, achieving this is, in a nutshell, the goal of inductive learning research. It is still true that some algorithms will match certain classes of naturally-occurring domains better than other algorithms, and so achieve higher accuracy than them, and that this may be reversed in other real-world domains; but this does not preclude an improved algorithm from being as accurate as the best in each of the domain classes.

Two induction paradigms that appear to have largely complementary strengths and weaknesses are rule induction and instance-based learning (IBL). IBL algorithms (also known as nearest-neighbor classifiers) are able to induce complex frontiers from relatively few examples, and are naturally suited to numeric domains, but can be very sensitive to irrelevant attributes. Conversely, rule induction algorithms perform well at finding simple axis-parallel frontiers but have difficulty when they do not exist, are best suited to symbolic domains, and can often dispose easily of irrelevant attributes. (The two paradigms also share a number of characteristics, of course, most notably the assumption that the instance space contains large continuous regions of constant class membership—the similarity hypothesis.)

Instances and rules also form the basis of two competing approaches to reasoning: case-based reasoning (Kolodner, 1993) and the rule-based reasoning more often found in expert systems. In recent years, case-based reasoning has gained popularity as an alternative to rule systems, but its proponents recognize that there is a wide spectrum between specific cases and the very general rules typically used (Riesbeck and Schank, 1989), one that has so far been left largely unexplored.

This report describes and evaluates the RISE 2.0 algorithm, an approach to inductive learning that produces knowledge bases spanning this entire spectrum. It achieves this by intelligently searching for the best mixture of selected cases and in-

creasingly abstract rules, based on the notion that instances are maximally specific rules, and that applying rules with a best-match strategy is essentially a more general form of nearest-neighbor classification.

The report is structured as follows. The next two sections briefly review the characteristics of IBL and rule induction most relevant to this work. The RISE algorithm will then be presented. Theoretical bounds for its time complexity will be derived, and an extensive empirical study comparing multiple versions of RISE and then the best version with several current induction algorithms will be described. Finally the results of this study will be discussed and interpreted, RISE will be placed in the context of related work, and some directions for future research will be suggested.

2 Instance-based learning

Instance-based learning, also known as exemplar-based learning and nearest-neighbor classification (Cover and Hart, 1967; Duda and Hart, 1973; Aha *et al*, 1991; Cost and Salzberg, 1993), is founded on a direct application of the similarity assumption. In the simplest case, learning is performed by storing all the observed instances. A new example (or "test case") is classified by finding the nearest stored example according to some metric, and assigning the latter's class to the former. The performance of IBL depends critically on the similarity (or, conversely, distance) metric used. In numeric domains (i.e., domains where all the features are real-valued), city-block and Euclidean distance are natural candidates. The component distance $\delta(x_i, x_j)$ between two values x_i and x_j of an attribute is then simply the absolute value of their difference; however, this may lead to attributes with a large spread of values having undue weight in the result, compared to attributes with smaller spreads. A commonly used alternative is to normalize the difference by its largest observed value:

$$\delta(x_i, x_j) = \left| \frac{x_i - x_j}{x_{max} - x_{min}} \right| \tag{1}$$

If there are A attributes, the distance between two instances $X_1 = (x_{11}, x_{12}, \ldots, x_{1A}, c_1)$ and $X_2 = (x_{21}, x_{22}, \ldots, x_{2A}, c_2)$ can then be defined as:

$$\Delta(X_1, X_2) = \sum_{i=1}^{A} \delta^r(x_{1i}, x_{2i}) \tag{2}$$

with $r = 1$ yielding city-block distance, $r = 2$ the square of Euclidean distance, and so on.

Symbolic attributes pose a more difficult problem. Most IBL systems (e.g. Aha *et al*, 1991) use a simple overlap metric:

$$\delta(x_i, x_j) = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

This measure is obviously less informative than its numeric counterpart, and its use has often been found to lead to poor performance (Cost and Salzberg, 1993). A more insightful alternative, first proposed by Stanfill and Waltz in 1986, consists of considering two symbolic values to be similar if they make similar predictions, i.e. if they correlate similarly with the class feature. According to this measure, which will henceforth be called the value difference metric, or VDM for short, the distance between two symbolic values is defined as:

$$\delta(x_i, x_j) = VDM(x_i, x_j) = \sum_{h=1}^{C} |P(c_h|x_i) - P(c_h|x_j)|^k \tag{4}$$

where C is the number of classes, c_h is the hth class, and k is a natural-valued parameter. The latter can be determined *ad hoc* or empirically. Notice that, in particular, $\delta(x_i, x_j)$ is always 0 if $i = j$. The total distance $\Delta(X_1, X_2)$ is computed as before. Slightly different variants of this metric have been successfully used in pronunciation and molecular biology tasks (Stanfill and Waltz, 1986; Cost and Salzberg, 1993).

Nearest-neighbor methods are highly sensitive to irrelevant attributes. The contributions of these attributes to the global distance constitute noise as far as the classification task is concerned, and they can swamp out the relevant components. We have found, however, that use of the VDM substantially attenuates this problem for symbolic attributes, as long as a large number of examples is available. This is due to the fact that by definition $P(c_m|x_i)$ will be the roughly the same for all values x_i of an irrelevant attribute, leading to zero distance (i.e. equivalence) between them. With numeric attributes, however, or when the available sample is small, the problem of sensitivity to irrelevant attributes remains.

Another issue in IBL methods is their sensitivity to noise. Incorrect instances are liable to create an area around them where new examples will also be misclassified. Several methods have been successfully introduced to deal with this problem. IB3 (Aha *et al*, 1991) retains only reliable instances, reliability being judged by the instance's classification performance over a "probation period." PEBLS (Cost and Salzberg, 1993) assign weights to instances, making their apparent distance to new examples increase with their misclassification rate.

Given two instances of opposite classes, the frontier between classes induced by them is a hyperplane perpendicular to the line connecting the two instances, and bisecting it. With multiple instances of each class, the frontier will be composed of a number of hyperplanar sections, and can thus become quite complex even with few instances (Aha *et al*, 1991). The introduction of weights further increases this complexity, turning the hyperplanes into hyperquadrics (Cost and Salzberg, 1993).

Another extension to the basic IBL paradigm consists in using the k nearest neighbors for classification, instead of just the nearest one (Duda and Hart, 1973). The class assigned is then that of the majority of those k neighbors, or alternatively the most voted one, with a neighbor's vote decreasing with its distance from the test example. This also gives rise to a convex division of the instance space.

It is important to note that, even though the stored instances used in classification are syntactically identical to examples, their semantic content (i.e. their extension) is quite different. An example is a single point in instance space, whereas a stored instance represents the entire region that it wins over in the competition with other instances. To highlight this distinction, these "active" examples are more correctly referred to as "exemplars."

3 Rule induction

Rule induction algorithms (Michalski, 1983; Michalski *et al*, 1986; Clark and Niblett, 1989; Pagallo and Haussler, 1990) typically employ a set covering or "separate and conquer" approach to induction. Table 1 summarizes this strategy in pseudo-code. Some definitions are in order. A rule is composed of a consequent part and and antecedent part or body. The consequent is the predicted class. The body is a conjunction of antecedents, each antecedent being a condition involving a single attribute. For symbolic attributes, this condition is a simple equality test; in some systems, negation and disjunction of values are possible. For numeric attributes, the condition is typically inclusion in a one-sided interval. A rule is said to cover an example, and conversely the example is said to satisfy it, if all the conditions in the rule are true for the example. Given a class, its members in the training set are called positive examples, and the remainder are negative.

The set covering or "separate and conquer" strategy derives its name from the fact that it forms a class definition by constructing a rule that covers many positive examples, and few or no negative ones, then "separating out" the newly covered examples and starting again on the remainder. It is a "general to specific" strategy, in that each rule initially covers all examples and is then gradually made more specific.

The choice of evaluation heuristic $H(N_\oplus, N_\ominus)$ is of some importance to the algorithm's performance. $H(N_\oplus, N_\ominus)$ should be an increasing function of $N_\oplus$, and a decreasing function of $N_\ominus$. The AQ series of algorithms (Michalski *et al*, 1986) uses apparent accuracy, i.e. the accuracy of the rule on the training set:

$$H(N_\oplus, N_\ominus) = \frac{N_\oplus}{N_\oplus + N_\ominus} \tag{5}$$

The CN2 system (Clark and Niblett, 1989) originally used entropy gain (Quinlan, 1986). The problem with both these measures, however, is that they tend to favor overly specific rules, because they attain their maximum value with a rule covering a single example. This can be overcome by use of the Laplace correction (Niblett, 1987):

$$H(N_\oplus, N_\ominus) = \frac{N_\oplus + 1}{N_\oplus + N_\ominus + C} \tag{6}$$

Table 1: General structure of rule induction algorithms.

Input: ES is the training set.

Procedure Rule_Induction (ES)

Let $RS = \emptyset$.
For each class C
  Let $\oplus = \{e \in ES \mid \text{Class}(E) = C\}$.
  Let $\ominus = \{e \in ES \mid \text{Class}(E) \neq C\}$.
  Repeat
    Let R = Find_Best_Rule (C, $\oplus$, $\ominus$).
    Let $\oplus = \oplus - \{e \in \oplus \mid R \text{ covers } e\}$.
  Until $\oplus = \emptyset$ or R = Nil.
Return RS.

Function Find_Best_Rule (C, $\oplus$, $\ominus$)

Let Body = True.
Let R be the rule: Body $\Rightarrow C$.
Repeat
  For each possible antecedent A
    Let B_A = Body $\wedge A$.
    Let $N_\oplus = \#(\{e \in \oplus \mid e \text{ satisfies } B_A\})$.
    Let $N_\ominus = \#(\{e \in \ominus \mid e \text{ satisfies } B_A\})$.
  Let Body = B_A that maximizes some heuristic H($N_\oplus$, $N_\ominus$).
Until no antecedent causes a significant improvement in H($N_\oplus$, $N_\ominus$).
Return R, or Nil if Body = True.

---

where C is the number of classes, as before. This measure tends to the uncorrected accuracy when the rule has strong statistical support, i.e. when it covers many examples, but tends to $1/C$, i.e. "maximum ignorance," when it covers few. It is used in recent versions of CN2 (Clark and Boswell, 1991).

Classification of a new example is performed by matching each rule against it, and selecting those it satisfies. If there is only one, its class is assigned to the example. If there is none, the generally adopted solution is to use the so-called "default rule," i.e. to assign the example to the class that occurs most frequently in the entire training set, or among those examples not covered by any rule. Finally, if more than one rule covers the example, two strategies are possible. One is to order the rules into a "decision list," and select only the first rule that fires (Pagallo and Haussler, 1990). The other is to let the different rules vote, and select the class receiving the most votes. Recent versions of CN2 attach to each rule the number of examples of each class that it covers, and use these numbers as votes at classification time (Clark and Boswell, 1991). The use of unordered rules has been found to generally achieve higher accuracy (Clark and Boswell, 1991), and also has the advantage of greater comprehensibility, since in a decision list each rule body is implicitly conjoined with the negations of all those that precede it.

In rule induction algorithms that do not deal with noise (e.g. AQ), construction of a new rule stops only when all negative examples are excluded. In noise-tolerant ones, a measure of statistical significance may be used to halt growth (as in CN2), or a later post-pruning step may remove superfluous antecedents and/or rules (e.g. GROVE; Pagallo and Haussler, 1990). Irrelevant attributes tend to produce no significant improvement in the evaluation heuristic, and thus be excluded; however, attributes that are relevant only in combination with other attributes may also be discarded. In numeric domains, the fact that only single-attribute tests are used in rules means that all decision boundaries are parallel to the coordinate axes, i.e. class definitions can only be unions of hyperrectangles, leading to innacurate definitions when this is not the case and only a limited number of examples is available.

Another shortcoming of the "separate and conquer" strategy is that it causes a dwindling number of examples to be available as induction progresses, both within each rule and for successive rules. This may result in later rules, and later antecedents within each rule, being induced with insufficient statistical support, leading to greater noise sensitivity and missing or incorrect rules/antecedents.

4 The RISE algorithm

We now describe an approach to induction that attempts to overcome some of the limitations of IBL and rule induction outlined above by unifying the two. This methodology is implemented in the RISE 2.0 system (Rule Induction from a Set of Exemplars; an earlier version, RISE 1.0, is described in (Domingos, 1994)). One of its basic features is that rules and instances are treated uniformly; no distinction is made between

the two. Another characteristic that distinguishes it from previous empirical multi-strategy learning systems is that it does not consist of a global procedure calling the individual algorithms as subprocedures, but rather a single, simple algorithm that can be viewed as both IBL and rule induction, according to its behavior. For these reasons, we speak of a "unification" of the two approaches, rather than using the somewhat weaker term "combination." Obviously, this should not be taken to imply that the form of unification proposed here is the only possible one.

4.1 Representation and definitions

A rule in RISE is composed of a consequent part that is the predicted class, and an antecedent part that is a conjunction of conditions, as before. Each condition involves only one attribute; for symbolic attributes it is an equality test (e.g. $x_1 = \alpha$), and for numeric attributes it is membership in an interval closed on both sides (e.g. $3 \leq x_2 \leq 7$). In each rule there is at most one condition involving each attribute, and there may be none. An example or instance is simply a rule in which the consequent is the example's class, there is exactly one condition per attribute, and all the intervals are degenerate (e.g. $4 \leq x_2 \leq 4$, i.e. $x_2 = 4$). In the remainder of this report, the word "rule" is used to refer indiscriminately to exemplars and to rules of the more general type.

A rule is said to cover an example if all its conditions are true for the example; a rule is said to *win* an example if it is the nearest rule to the example according to the distance metric that will be defined below. A rule may cover an example and not win it. The extension of a rule is constituted by all the points in instance space that it wins, whether or not it covers them, and therefore depends not only on the rule itself but on all the other rules.

The accuracy $Acc(RS, ES)$ of a rule set RS on a set of examples ES is defined as the fraction of those examples that it correctly classifies. A rule set classifies an example correctly when the nearest rule to the example has the same class as it. Whenever the example set ES is simply the whole training set this will be left implicit, i.e. the accuracy will be represented by $Acc(RS)$. Note that, when comparing the accuracy of different rule sets on a training set, there is no need to use the Laplace correction, because the denominator of the accuracy (the number of examples matched) is exactly the same for all rule sets (it is the size of the training set).

Let $E = (e_1, e_2, \ldots, e_A, c_E)$ be an example with value e_i for the ith attribute and class c_E. Let $R = (a_1, a_2, \ldots, a_A, c_R)$ be a rule with class c_R and condition a_i on the ith attribute, where a_i = True if there is no condition on i, otherwise a_i is $x_i = r_i$ if i is symbolic and a_i is $r_{i,lower} \leq x_i \leq r_{i,upper}$ if i is numeric. The distance $\Delta(R, E)$ between R and E is then defined as:

$$\Delta(R, E) = \sum_{i=1}^{A} \delta^r(i) \tag{7}$$

where r is a natural-valued parameter, and the component distance $\delta(i)$ for the ith attribute is:

$$\delta(i) = \begin{cases} 0 & \text{if } a_i = \text{True} \\ VDM(r_i, e_i) & \text{if } i \text{ is symbolic and } a_i \neq \text{True} \\ \delta_{num}(i) & \text{if } i \text{ is numeric and } a_i \neq \text{True} \end{cases} \quad (8)$$

where in turn $VDM(r_i, e_i)$ is the value difference metric as defined in Eq. 4, and:

$$\delta_{num}(i) = \begin{cases} 0 & \text{if } r_{i,lower} \leq e_i \leq r_{i,upper} \\ \frac{e_i - r_{i,upper}}{x_{max} - x_{min}} & \text{if } e_i > r_{i,upper} \\ \frac{r_{i,lower} - e_i}{x_{max} - x_{min}} & \text{if } e_i < r_{i,lower} \end{cases} \quad (9)$$

x_{max} and x_{min} being respectively the maximum and minimum observed values for the attribute.

The distance from a missing numeric value to any other is defined as 0. If a symbolic attribute's value is missing, it is assigned the special value "?". This is treated as a legitimate symbolic value, and its VDM to all other values of the attribute is computed and used. In the present framework, this is a sensible policy: a missing value is taken to be roughly equivalent to a given possible value if it behaves similarly to it, and inversely if it doesn't.

4.2 Control structure

Unlike conventional rule induction algorithms, RISE does not construct one rule at a time, but instead induces all rules in parallel. In addition, heuristic evaluation is not performed for each rule separately, but for the whole rule set at once. Changes to an individual rule are evaluated in terms of their effect on the global accuracy of the rule set. This "conquering without separating" strategy differs markedly from the earlier "separate and conquer" one; the aim is to attenuate the splintering problem as much as possible. Another major difference is that RISE's direction of search is specific-to-general. Rules are generalized by dropping conditions on symbolic attributes, and broadening intervals for numeric ones. This is not done one attribute at a time, but rather by a clustering-like approach: each rule repeatedly finds the nearest example of its class that it doesn't yet cover, and attempts to minimally generalize itself to cover it. If the effect of this on global accuracy is positive, the change is retained. This process stops when no further change causes any improvement. The initial rule set is the training set itself, i.e. each example is a candidate rule. In the worst case no generalizations are accepted, and the final rule set is still the training set, leading to a pure nearest-neighbor algorithm. More generally, the final rule set may contain some ungeneralized exemplars as well as more abstract rules. In the course of generalization two rules may become identical, in which case they are merged. Table 2 summarizes this process in pseudo-code. Note that in each cycle the new rule set is adopted even

Table 2: The RISE algorithm.

Input: ES is the training set.

Procedure RISE (ES)

Let RS be ES.
Compute $Acc(RS)$.
Repeat
  For each rule R in RS,
    Find the nearest example E to R not already covered by it, and of R's class.
    Let R' = Most_Specific_Generalization(R, E).
    Let RS' = RS with R replaced by R'.
    If $Acc(RS') \geq Acc(RS)$
    Then Replace RS by RS',
      If R' is identical to another rule in RS,
      Then delete R' from RS.
Until no increase in $Acc(RS)$ is obtained.
Return RS.

---

if its apparent accuracy is the same as the old one's. This is a direct application of Occam's Razor: when two theories appear to perform identically, prefer the simpler one.

This method would not be efficient if the accuracy of the entire rule set had to be computed from scratch every time an individual change is considered. This would involve repeatedly matching all rules against all examples, leading to a clearly unacceptable time cost. Fortunately, only the change in accuracy $\Delta Acc(RS)$ needs to be considered. Each example memorizes the distance to the nearest rule (i.e. the rule that wins it) and that rule's identification. The memory cost of this is $O(1)$ per example, i.e. negligible. When a rule is generalized, all that is necessary is then to match that single rule against all examples, and check if it wins any that it did not before. Its effect on these is then ascertained. If a previously misclassified example is now correctly classified, the numerator of $Acc(RS)$ is incremented; if the reverse takes place, it is decremented. Otherwise there is no change. If the sum of increments and decrements is greater than or equal to 0, the new rule is adopted, and the relevant structures are updated; otherwise it is rejected.

Table 3 shows in pseudo-code how a rule is minimally generalized to cover an ex-

Table 3: Generalization of a rule to cover an example.

Inputs: $R = (a_1, a_2, \ldots, a_A, c_R)$ is a rule, $E = (e_1, e_2, \ldots, e_A, c_E)$ is an example.
a_i is either True, $x_i = r_i$, or $r_{i,lower} \leq x_i \leq r_{i,upper}$.

Function Most_Specific_Generalization (R, E)

For each attribute i,
  If a_i = True then Do nothing.
  Else if i is symbolic and $e_i \neq r_i$ then Drop a_i from R.
  Else if $e_i > r_{i,upper}$ then $r_{i,upper} = e_i$.
  Else if $e_i < r_{i,lower}$ then $r_{i,lower} = e_i$.

---

ample previously outside its scope. In a nutshell, all conditions on symbolic attributes that are not satisfied by the example are dropped, and all intervals are extended to include the example attribute's value at the border, if necessary. Note that this is indeed the most specific generalization that will work, given the representation language used (e.g. internal disjunction is not permitted). Missing values are treated in a fashion consistent with the definition of distance above: missing numeric values have no effect, and a missing symbolic value in the example or in the rule (but not both) causes the corresponding condition to be dropped.

As mentioned before, classification of a new example is performed by assigning it the class of the nearest rule in the knowledge base. The question arises of how to choose the winning rule when several are equally near. This is of more importance in RISE than in IBL, because it may frequently occur that several rules cover the example, i.e. are at distance 0 from it. This corresponds to the multiple-match case in rule induction systems. RISE selects the rule with the highest Laplace accuracy. This means that neither very general nor very specific rules are unduly favored; rather, preference goes to rules with high apparent accuracy as well as strong statistical support. In the event the accuracies are the same, RISE chooses the most frequent class among those represented, and if there is still a draw, the winner is chosen at random. Other policies were also tried, and a comparative evaluation is included in a later section.

5 Time complexity of RISE

It is possible to derive an upper bound for the time complexity of the algorithm just described, showing that its efficiency is comparable to that of other induction algorithms. Let E represent the number of examples in the training set, A the number of attributes used to describe each, V the average number of values per attribute, R the number of rules, and C the number of classes into which the examples fall. Assume for now that all attributes are nominal. The initialization phase of the algorithm consists of three operations. The first is copying the examples to the rules, and takes $O(EA)$ time. The second is compiling a table of VDM distances, taking $O(EA + AV^2C)$ (EA to run through all the examples, for each one noting the correspondence between each attribute's value and the class, and AV^2C to sum the results for all classes, for each pair of values of each attribute). The third operation is finding each example's closest rule and computing the initial accuracy of the rule set, which involves matching all rules against all examples, and so takes $O(E^2A)$ time. The total time necessary for initialization is therefore $O(E^2A + AV^2C)$.

The heart of the algorithm consists of four steps: finding a rule's nearest example, generalizing the rule to cover it, comparing the altered rule to all examples to see if any are newly won, and (if the change is adopted) comparing the rule to all other rules to check for duplications. These operations consume respectively $O(EA)$, $O(A)$, $O(EA)$ and $O(RA)$ time, for a total of $O(EA + RA)$. Since each "repeat" cycle (see Table 2) consists of doing this for all R rules, each such cycle takes at worst $O[R(EA + RA)]$ time. In RISE each example produces at most one rule; therefore $R \leq E$, and this time is at worst $O(E^2A)$.

How many "repeat" cycles can the algorithm perform in the worst case? Two answers are possible, depending on how the stopping criterion is interpreted. If it is applied individually, i.e. generalization of a given rule stops as soon as covering the nearest example produces no improvement, then the "repeat" cycle is performed at worst $O(A)$ times, since each cycle must remove at least one condition, and a rule contains at most A conditions, this being true for each rule. On the other hand, if the stopping criterion is applied globally, i.e. generalization of a given rule stops only when no change to *any* rule produces an improvement, the "repeat" cycle can in theory be performed up to $O(EA)$ times, because in the worst case only one condition of one rule will be dropped in each entire cycle, each time causing some currently-unprofitable change in another rule to become profitable in the next round. The two policies are empirically compared in a later section, showing no appreciable difference between the two in accuracy or time. Multiplying the values above by the cost of a single cycle yields a total time complexity of $O(E^2A^2)$ or $O(E^3A^2)$ respectively. Since $E \geq V$, and assuming that $A \geq C$, which is generally the case, the smaller of these values dominates the complexity of the initialization phase, and both therefore constitute upper bounds on the time complexity of the whole algorithm in their respective situations.

The time complexity of e.g. CN2 is $O(BE^2A^2)$, where B is the beam size, an integer-valued internal parameter of the algorithm. (Note that the computations in (Clark and Niblett, 1989) are only for the basic step of the algorithm, which is embedded in loops that may run $O(EA)$ in the worst case.) Thus RISE's worst-case time complexity is comparable to that of CN2, and to those of of similar rule and decision tree induction systems. Average-case time is also likely to be substantially smaller than the worst case, because some of the assumptions above are overly pessimistic (e.g. in general R will seldom remain equal to E, but will instead shrink rapidly; attributes will be dropped several at a time, and not all will be removed; increasing E may not increase the number of "repeat" cycles, even with a global stopping criterion; etc.).

The introduction of numeric values simply increases the values above by a factor of V, since the single-step removal of a condition may now be replaced by at most $O(V)$ steps of expanding the corresponding interval. Again, in practice only a small number of steps may actually be required.

6 Empirical study

With the twin goals of refining RISE and comparing its performance with that of previous approaches, an extensive empirical study was carried out. The next few subsections describe the characteristics and report the results of this study.

6.1 Application domains used

Thirty real-world domains were used in the study. An attempt was made to include every available domain that has been widely used in inductive learning studies, and beyond that to provide a wide sampling of: symbolic, numeric and mixed domains; small, medium and large domains; domains of varying difficulty, as expressed in the highest accuracy previously achieved; and domains from a wide range of application areas. All domains were drawn from the UCI repository (Murphy and Aha, 1992). Reasons for excluding datasets in this repository from the study were:

- Some datasets are inappropriate for this type of algorithm. This includes: datasets involving relational data, domain theories, structured instances, or variant instances; datasets intended for psychological studies, and therefore minuscule; and datasets intended for regression tasks (ie. where the predicted value is numeric).

- In some cases there are multiple datasets from the same domain, and duplicated datasets. In this case only the most widely used dataset was included.

- There are too many purely numerical domains in the repository. Use of all would result in a higher proportion of this type of domain than is usually the case in

machine learning studies, undermining comparisons with previous literature.

- Some datasets are inadequately documented and/or formatted.
- Some datasets are artificially generated ones, without correspondence to any real-world problem.
- In some datasets there is no clear feature to use as class.

Essentially all the datasets in the UCI repository that didn't fall under one of these restrictions were included in the study. Table 4 lists these datasets in alphabetical order of code-name, and summarizes some of their main characteristics. Domains included in the listing of empirical results in (Holte, 1993) are referred to by the same codes. The filenames of the datasets used, and any conversions that had to be performed, are listed in an appendix to this report.

6.2 Variations on RISE

In the first phase of the study, half of the domains were used to compare different versions of RISE, and select the best one by 10-fold cross-validation. The domains used were: breast cancer, credit screening, chess endgames, Pima diabetes, hepatitis, iris, labor negotiations, lung cancer, liver disease, contact lenses, lymphography, primary tumor, soybean, voting records, and wine.

For the sake of conciseness, tables containing numerical results for each comparison are omitted, but the main observations are of interest, not only to RISE but also in the wider context of the issues they address, and are therefore summarized below. In each case, only the general trend is reported; almost invariably, exceptions to it were also observed. Whenever no significant difference in accuracy was observed, the simpler version of the algorithm was chosen; otherwise the more accurate one prevailed. The comparisons made and respective conclusions were as follows.

- *Use of weights on exemplars.* Two versions were compared: no weights, and each rule weighted by the inverse of its Laplace accuracy, leading unreliable rules to appear farther from new instances. This is similar to the weighting scheme used in PEBLS (Cost and Salzberg, 1993), but slightly more sophisticated because the Laplace correction is used. Note that this correction is indeed justified when dealing with individual rules. No significant difference was observed. This may be due to the fact that the domains are not too noisy, to the fact that RISE's basic approach to induction is successful by itself in combatting noise, or to both. No weights is the default in RISE.

- *Tie-breaking.* When more than one rule was equally near the test example, ties were broken by choosing the one with highest Laplace accuracy, by choosing the most specific one, and by frequency-based voting as done in CN2. The

Table 4: Domains used in the empirical study. The columns are, in order: name of the domain; 2-letter code used to refer to it in subsequent tables; number of examples; number of attributes; number of numeric attributes; number of classes; number of missing values in the entire dataset; and whether or not the dataset includes inconsistent examples (i.e. identical examples with different classes).

Domain	Code	Exs.	Atts.	Num.	Classes	Missing	Incons.
Audiology	AD	200	69	0	24	291	No
Annealing	AN	798	38	9	5	19692	No
Breast cancer	BC	286	9	4	2	9	Yes
Credit screening	CE	690	15	6	2	67	No
Chess endgames	CH	3196	36	0	2	0	No
Pima diabetes	DI	768	8	8	2	0	No
Echocardiogram	EC	131	7	6	2	40	Yes
Glass	GL	214	9	9	6	0	No
Heart disease	HD	303	13	6	2	7	No
Hepatitis	HE	155	19	6	2	167	No
Horse colic	HO	300	22	7	2	1605	Yes
Thyroid disease	HY	3163	25	7	2	5329	Yes
Iris	IR	150	4	4	3	0	No
Labor negotiations	LA	57	16	8	2	326	No
Lung cancer	LC	32	56	0	3	5	No
Liver disease	LD	345	6	6	2	0	No
Contact lenses	LE	24	4	0	3	0	No
LED	LI	100	7	0	10	0	Yes
Lymphography	LY	148	18	3	4	0	No
Mushroom	MU	8124	22	0	2	2480	No
Post-operative	PO	90	8	8	3	3	Yes
DNA promoters	PR	106	57	0	2	0	No
Primary tumor	PT	339	17	0	21	225	Yes
Solar flare	SF	323	12	3	6	0	Yes
Sonar	SN	208	60	60	2	0	No
Soybean	SO	47	35	0	4	0	No
Splice junctions	SP	3190	60	0	3	0	Yes
Voting records	VO	435	16	0	2	392	No
Wine	WI	178	13	13	3	0	No
Zoology	ZO	101	16	1	7	0	No

best-performing alternative was using Laplace accuracy. Use of specificity had a clear negative effect on accuracy, contradicting heuristics used in some other systems (e.g. EACH: Salzberg, 1991). Tie-breaking when the accuracies are also identical was described in a previous section.

- *Distance computation.* Values of $k = 1$ and $k = 2$ (see Eq. 4) were tried, each combined with values of $r = 1$ and $r = 2$ (see Eq. 7). No appreciable difference was observed, confirming previous results (Cost and Salzberg, 1993). The default values for RISE are $k = 1$ and $r = 2$ (Euclidean distance).

- *Treatment of numeric values.* The following versions were compared: normalization by the difference of the limits, as in Eq. 9; normalization by 3 and 4 times the standard deviation for the attribute; discretization into equal-sized intervals up to a maximum of 10; and intervalization by entropy minimization, i.e. ordering the values and successively choosing the splitting point that most reduces the entropy, until a maximum number of intervals is reached (10, 100) or the reduction obtained is at or below a given minimum (10%, 1%, 0%). The two first methods were in general clearly superior to the latter two, although this was reversed in some domains. The entropy-based method also caused a noticeable increase in computation time for the larger datasets. The two types of normalization performed very similarly; limit difference was chosen, due to its greater simplicity.

- *Treatment of missing values.* Missing symbolic values were treated in two ways: as legitimate symbolic values(see earlier discussion), and matching every value as done for numeric values. The first alternative proved superior.

- *Search.* Two types of search were attempted: finding only the nearest example and attempting to cover it, and finding the 3 nearest examples, attempting to cover each and choosing the best result (or no change, as before). The latter alternative produced no substantial improvement, as well as being predictably slower. The first was chosen.

- *Final search.* Two alternatives were tried when the generalization of a rule to the nearest example does not improve accuracy. One was to do nothing. The other was to attempt generalization to all other examples of the rule's class in order of increasing distance from it, until an improvement was obtained or the examples were exhausted. Again, this variation produced a slowdown and no overall improvement in accuracy, and was not adopted.

- *Stopping criterion.* Two stopping criteria were compared: local, where a rule's generalization is terminated as soon as an attempt to generalize it fails, and global, where this termination only occurs when such attempts have failed for all rules. There was no significant difference between the two in running time,

and global stopping tended to produce slightly higher accuracies, so this policy was chosen.

- *Merging rules.* Three policies for merging rules were compared: deleting duplicate rules, deleting subsumed rules (i.e. rules logically implying the subsuming rule), and deleting subsumed rules only if they were not more accurate than the subsuming rule. The rationale for the last one is that in RISE a subsumed rule can have a positive effect on overall accuracy, because it may win examples that the subsuming rule wouldn't. The two latter approaches produced some speedup and somewhat more compact rule sets, but also had a noticeable negative effect on accuracy. The default for RISE is therefore deleting only duplicated rules.

- *Simplification.* Two post-processing techniques that further simplified the final rule set were tested, individually and in combination. One was to delete all rules that won no examples. The other was to delete all attributes in a rule that did not have different values in examples of other classes. Both strategies were successful in simplifying the rule sets, particularly the first, with the greatest reductions predictably produced by the combination of the two; compression rates in excess 90% for the whole algorithm were common. However, this simplification was accompanied by a small overall decrease in accuracy, and post-processing is thus not the default in RISE.

  These observations stand in contrast to studies on decision tree learners (Quinlan, 1987b) and the AQ15 rule induction system (Michalski et al, 1986), where simplification was (surprisingly) accompanied by an increase in accuracy. Together with a recent study by Murphy and Pazzani (1994) that suggests the most accurate decision tree is not always the simplest, this may indicate that the accuracy/complexity curve is not as simple as formerly supposed.

6.3 Systems compared and experimental procedure

Thus selected, the default version of RISE was then compared on all 30 domains with several other systems. (Strictly speaking, this comparison should be performed on only the remaining 15 domains, but as will be seen below, the pattern of results is identical in the two groups, and testing on more domains produces a higher level of confidence in the results.) A representative of each of RISE's parent approaches was used: PEBLS for IBL (Cost and Salzberg, 1993), and CN2 for rule induction (Clark and Niblett, 1989). Note that PEBLS is a state-of-the art system, as opposed to the skeleton nearest-neighbor implementations typically used in empirical comparisons. PEBLS 2.1's inability to deal with missing values was overcome by grafting onto it an approach similar to the one selected for RISE (see previous section). All default settings were used. The number of intervals for each numeric attribute has to be set by the user, and it is recommended that it be kept small for best results. The policy

was adopted of using the number of observed values or 10, whichever was lowest. *Ad hoc* variation of this value produced no appreciable change in results. A recent version of CN2 (6.1) was used, one incorporating Laplace accuracy and the use of unordered rules (Clark and Boswell, 1991). Again, all default settings were used.

To gauge its position in the overall spectrum of induction methods, RISE was also compared with a decision tree learner, C4.5 (Quinlan, 1993a). C4.5RULES, the version of C4.5 that converts fully-sprouted decision trees into production rules and then prunes the latter, was chosen because this approach has been observed to achieve the highest accuracy (Quinlan, 1987a), as well as being the one most directly comparable to RISE. The default classifier (always choosing the most frequent class) was also included in the study to provide a baseline. Back-propagation (Rumelhart *et al*, 1986), although a widely-used learning algorithm, was left out because its need for extensive fine-tuning and very long running times would make a large-scale study of this type difficult to carry out.

Each dataset was randomly divided 50 times into a training set containing two-thirds of the examples, and a testing set containing the remainder. In the interests of making it possible to efficiently perform many runs on many domains, datasets with more than 1000 examples were first reduced to this size by random selection. All of the algorithms were trained on each of the 50 training sets and tested on the corresponding testing set. The results reported below are averages of these 50 runs.

6.4 Results

The average accuracies obtained and their standard deviations are presented in Table 5 below. Comparison of RISE with each algorithm is then performed in Table 6, where a + indicates that RISE was more accurate than the algorithm in that column, – indicates the opposite, and $*$ represents a draw. Note that Table 5 contains rounded values, and accuracies that appear there as identical may in fact be different. The confidence levels in Table 6 were obtained using a one-tailed paired t test with critical values spaced at 5% intervals, plus 97.5%, 99% and 99.5%.

Although many results in individual domains are interesting in themselves, these tables are more easily understood if their results are summarized in a few global measures of comparative performance. Six such measures and their values are presented in Table 7. They can be described and interpreted as follows.

- *Number of wins.* The first and most obvious test is simply to count the number of domains where RISE achieved higher accuracy than the second algorithm, count those where its accuracy was lower, and compare the two (draws are not counted either way). The first line of Table 7 shows this: for example, RISE performed better than PEBLS in 22 domains, and inversely in 7 (there was 1 draw). As can be seen, RISE outperformed every other algorithm in this respect by a ratio of roughly 3 to 1.

Table 5: Empirical results: average accuracies and standard deviations.

Domain	RISE	Default	PEBLS	CN2	C4.5
AD	76.9±5.3	20.8±3.6	75.8±5.2	69.4±6.6	70.9±6.0
AN	97.4±0.9	76.2±2.2	99.0±0.7	82.1±6.6	93.6±1.6
BC	67.7±4.7	68.5±6.8	64.4±4.1	68.2±5.7	68.6±5.4
CE	83.3±2.2	56.7±3.7	81.0±2.2	82.2±2.5	84.6±2.4
CH	96.2±0.9	50.3±3.5	95.5±1.0	95.5±1.5	97.4±1.3
DI	70.3±2.4	65.4±2.4	70.7±3.0	73.7±2.4	73.7±3.0
EC	66.2±5.9	68.1±6.9	63.1±6.6	68.3±6.9	65.8±7.0
GL	70.6±5.6	31.4±5.0	58.6±5.9	64.2±6.0	64.4±7.7
HD	79.8±3.7	55.1±3.8	78.2±4.3	78.5±3.6	76.2±4.3
HE	79.2±4.7	79.1±3.6	78.6±4.7	80.6±4.6	78.4±5.5
HO	82.0±3.0	64.1±3.8	76.1±3.7	83.0±3.6	81.0±3.8
HY	96.8±0.7	95.1±0.9	94.3±1.6	98.0±0.7	98.7±0.7
IR	94.0±2.9	26.3±3.8	93.2±2.9	93.0±3.6	93.8±2.9
LA	88.6±8.5	66.1±10.2	88.5±5.4	80.9±8.8	83.9±9.2
LC	45.1±16.6	24.7±15.5	42.0±15.2	34.0±15.4	40.4±14.9
LD	62.3±4.8	57.8±3.2	62.1±3.6	65.2±4.2	64.1±4.3
LE	77.2±12.8	60.8±13.1	72.5±12.6	71.5±14.1	77.0±11.7
LI	59.8±7.1	7.3±3.2	53.0±7.0	55.8±9.5	58.8±8.7
LY	78.6±6.3	55.5±6.9	81.7±5.4	77.6±5.5	75.0±4.6
MU	99.7±0.4	48.3±1.4	99.8±0.3	99.6±0.4	99.3±0.6
PO	64.0±7.5	71.0±6.2	58.3±7.9	62.1±7.5	67.5±8.3
PR	86.9±5.3	42.1±5.3	90.6±5.2	74.2±9.7	79.0±9.8
PT	40.4±4.8	24.3±3.4	31.4±4.1	40.0±5.2	37.5±5.2
SF	71.5±3.7	25.4±4.1	68.2±3.5	69.8±4.5	71.3±4.1
SN	77.9±9.2	51.0±7.0	76.9±6.1	68.8±7.6	66.4±6.5
SO	100.0±0.0	25.9±16.2	100.0±0.0	97.1±5.4	96.0±6.5
SP	91.4±1.7	54.0±3.4	92.9±1.2	83.8±3.2	90.5±2.0
VO	95.2±1.5	60.7±3.0	94.6±1.3	95.3±1.5	95.4±1.6
WI	96.9±1.9	36.9±8.6	95.9±2.3	91.5±4.6	91.3±4.9
ZO	93.9±3.7	40.5±6.3	94.9±4.2	91.9±5.2	90.8±5.4

Table 6: Empirical results: comparison of RISE with each algorithm. A + following the confidence level indicates that RISE's accuracy was higher, a – indicates the opposite, and * denotes a draw.

Domain	Default	PEBLS	CN2	C4.5
AD	99.5 +	90.0 +	99.5 +	99.5 +
AN	99.5 +	99.5 –	99.5 +	99.5 +
BC	80.0 –	99.5 +	70.0 –	85.0 –
CE	99.5 +	99.5 +	99.5 +	99.5 –
CH	99.5 +	99.5 +	99.0 +	99.5 –
DI	99.5 +	80.0 –	99.5 –	99.5 –
EC	90.0 –	99.5 +	95.0 –	65.0 +
GL	99.5 +	99.5 +	99.5 +	99.5 +
HD	99.5 +	99.0 +	97.5 +	99.5 +
HE	55.0 +	80.0 +	95.0 –	80.0 +
HO	99.5 +	99.5 +	97.5 –	95.0 +
HY	99.5 +	99.5 +	99.5 –	99.5 –
IR	99.5 +	95.0 +	95.0 +	60.0 +
LA	99.5 +	50.0 +	99.5 +	99.5 +
LC	99.5 +	80.0 +	99.5 +	90.0 +
LD	99.5 +	55.0 +	99.5 –	97.5 –
LE	99.5 +	99.5 +	99.5 +	55.0 +
LI	99.5 +	99.5 +	99.5 +	75.0 +
LY	99.5 +	99.5 –	80.0 +	99.5 +
MU	99.5 +	80.0 –	97.5 +	99.5 +
PO	99.5 –	99.5 +	97.5 +	99.5 –
PR	99.5 +	99.5 –	99.5 +	99.5 +
PT	99.5 +	99.5 +	65.0 +	99.5 +
SF	99.5 +	99.5 +	97.5 +	60.0 +
SN	99.5 +	75.0 +	99.5 +	99.5 +
SO	99.5 +	50.0 *	99.5 +	99.5 +
SP	99.5 +	99.5 –	99.5 +	99.0 +
VO	99.5 +	97.5 +	80.0 –	75.0 –
WI	99.5 +	99.5 +	99.5 +	99.5 +
ZO	99.5 +	97.5 –	99.5 +	99.5 +

Table 7: Summary of empirical results.

Measure	RISE	PEBLS	CN2	C4.5
No. wins	-	22-7	22-8	22-8
No. signif. wins	-	16-5	20-6	15-6
Sign test	-	99.5	99.1	99.1
Wilcoxon test	-	99.8	99.9	99.6
Average	79.7	77.7	76.5	77.7
Score	93.5	61.0	64.0	68.5

- *Number of significant wins.* The previous measure is clearly imperfect, for some of the differences may be very small and of no real significance. A good alternative is then to count only those domains where the difference was significant at a confidence level of 95% or higher (see Table 6). This yields roughly similar ratios, with all values somewhat reduced as might be expected, confirming the previous observation of RISE's substantially better performance compared to every other algorithm.

- *Sign test.* The goal of machine learning is not in general to produce algorithms targeted to one specific domain, but rather to produce algorithms that are accurate across broad classes of domains. In trying to answer the question "Is RISE a more accurate system?" it is then useful to look at the 30 domains used in this study as a sample of size 30 taken from the set of real-world domains, and consider each domain as a Bernoulli trial whose outcomes are + (RISE better) and – (RISE worse). The sign test then measures how unlikely the number of +'s observed is according to the binomial distribution, under the null hypothesis that there is no difference, i.e. $P(+) = \frac{1}{2}$ (DeGroot, 1986). (A draw adds 0.5 to the binomial variable. This is not a problem, since for 30 trials the normal approximation with the continuity correction can reliably be used.) The results are conclusive: in each case, they support the hypothesis that RISE is a more accurate algorithm with a confidence in excess of 99%.

- *Wilcoxon test.* The sign test takes only into account the sign of the difference in performance between the two algorithms, so again it can be misled by very small differences. Intuitively the magnitude of the differences should also be taken into account, but not so much that domains where there are naturally higher variations acquire undue weight. The Wilcoxon test is a more sensitive procedure that addresses these issues (DeGroot, 1986). It takes into account the *ranks* of the magnitudes, though not their exact values, i.e. the largest difference counts more than the second largest one and so forth, but it does not matter by how much. Reflecting this test's greater sensitivity, the results

are even more markedly favorable to RISE in every case, with error levels invariably below 0.5%. This means that overall the larger and "surer" differences tend to be favorable to RISE, and the small or "doubtful" ones to the second algorithm. Is is therefore highly unlikely that RISE performed better than the other algorithms by chance. If the domains used constitute a good sample of the set of real-world domains to which these algorithms will be applied, we can then conclude with high confidence that RISE is the most accurate one.

- *Average.* The average performance across all domains is a measure of debatable significance, but it has often been used (e.g. Quinlan, 1993b; Clark and Boswell, 1991) and provides additional perspective. Again RISE does visibly better than every other algorithm. It is also interesting to note that, although IBL (PEBLS) and rule induction (CN2) often differ by large margins in specific domains, globally these differences tend to cancel each other out. This is also true when comparing C4.5 with PEBLS.

- *Score.* The score is a measure that compares all 5 algorithms simultaneously (default included), looking not only at which one is the most accurate one, but also taking into consideration that being the second is better than being the worst, etc. Specifically, for each domain the most accurate algorithm receives 4 points, the second 3, the third 2, the fourth 1 and the worst 0. RISE obtains the largest score by a wide margin: whereas PEBLS, CN2 and C4.5 all have values in the 60's, RISE's is greater than 90. C4.5 is the second best algorithm, followed by CN2 and then PEBLS.

The same general pattern typically emerges if the results are broken down by domain size (small, medium and large, i.e. $E < 100$, $100 \leq E < 1000$, and $E \geq 1000$), domain type (symbolic, numeric and mixed), difficulty (easier and harder, as measured by the highest accuracy being above or below 75%, the global average), and area (medical diagnosis, engineering, social science, life sciences and miscellaneous). Confidence levels are of course reduced due to the smaller sample size, and there is sometimes no clear pattern when the number of domains is very small, but RISE's excellent results clearly hold across the board.

A very significant observation is that in 15 domains RISE's accuracy exceeds the highest of CN2's and PEBLS's, i.e. RISE not only matches the results of the best of its parent approaches, but is able to improve on them. In 10 of those domains this is true with a confidence level of 95% or higher, using a one-tailed paired t test (see Tables 5 and 6). This stands in contrast to the results achieved by e.g. MCS (Brodley, 1993), and shows that a significant synergy can be obtained by combining multiple empirical learning approaches.

7 Discussion

The empirical study just described leads to the conclusion that RISE represents a significant advance in the state-of-the-art in empirical learning. RISE's higher accuracy may be attributed to several factors. Compared to IBL systems, RISE draws strength from its ability to perform abstraction and simplify frontiers, successfully combatting noisy instances and irrelevant attributes. In particular, RISE does not depend on VDM alone to filter out irrelevant attributes, meaning that it can do so without discretizing numeric attributes and even when the training set is small. RISE also produces significantly more compact knowledge bases than nearest-neighbor algorithms, while generally exceeding their performance.

Compared to "divide and conquer" or "separate and conquer" systems like C4.5 and CN2, RISE also has several advantages. It is able to form complex non-axis-parallel frontiers when they are justified. It minimizes the splintering problem through its global evaluation and induction strategies, and at the same time it is able to take individual examples into consideration, instead of using only statistics derived from them. It also uses a more sensitive best-match strategy, instead of always choosing the most frequent class when there is no covering rule or branch.

RISE has some disadvantages. It is on average the slowest of the systems compared, although this is only of any significance in the largest domains (in all others, every algorithm runs in seconds on a Sun Sparcstation). RISE has not been optimized, however, and several important components of the system are amenable to such optimization. Beyond that, windowing and other sampling techniques can be used without expected loss in accuracy (Catlett, 1991). RISE's rule sets are also not as compact as those produced by C4.5 or CN2, although as reported earlier there may be a trade-off between simplicity and accuracy. However, for domains of very large size, and/or when comprehensibility is paramount, a system like C4.5 will still be the first choice.

The RISE approach should be seen in the context of previous work in inductive learning and related areas. The AQ15 system (Michalski *et al*, 1986) employed best-match classification, but used a more primitive overlap-based distance measure. Golding and Rosenbloom (1991) designed a system that gainfully combined case-based and rule-based reasoning, but it did not learn, and was hindered by the use of different representations for cases and rules. Quinlan (1993b) has successfully combined IBL with trees and other methods, but for the purpose of regression as opposed to classification, performing this combination only at classification time, and in a way that depends critically on the predicted value being continuous. Several induction algorithms proposed in the literature can be seen as empirical multi-strategy learners, but combining different paradigms from RISE's: decision trees and rules (Quinlan, 1987), decision trees and perceptrons (Utgoff, 1989), rules and Bayesian classification (Smyth *et al*, 1990), back-propagation and genetic algorithms (Belew *et al*, 1992), etc.

In form, the most similar system to RISE in the literature is EACH (Salzberg, 1991), which produces and uses hyperrectangles generalized from specific instances. It differs from RISE in many ways, however: it is applicable only in purely numerical domains, is an incremental algorithm, never drops attributes, uses different heuristics and search strategies, allows only nested hyperrectangles as opposed to arbitrary intersecting ones, always prefers the most specific hyperrectangle, etc. MCS (Brodley, 1993) has the most similar aims to RISE's, but uses an entirely different approach (applying meta-knowledge to detect when one algorithm should be applied instead of another), and generally achieved accuracies between those of the individual algorithms.

A major direction for future research is extending the current RISE framework to include analytical as well as empirical learning. This will make it possible to perform theory revision and interleaved automatic/manual construction of knowledge bases, as well as pure induction. The first step towards this goal is the introduction of specialization operators, in addition to the current generalization procedure. The use of general-to-specific search in a "conquering without separating" mode will also allow closer comparison of the latter with the currently prevalent "separate and conquer" strategy. Finally, a user-friendly version of RISE will be made available.

In conclusion, the research reported here attempted to bring together the best features of rule induction and instance-based learning in a single algorithm. Extensive empirical evaluation has shown this attempt to be remarkably successful, contributing to the simplification of the knowledge engineer's task, and improving the accuracy of learning systems.

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 the creators of the CN2, PEBLS and C4.5 systems, and to all the people who provided the datasets used in the empirical study. Please see the documentation in the UCI Repository for detailed information.

References

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning, 6*, 37–66.

Belew, R. K, McInerney, J., & Schraudolph, N. N. (1992). Evolving networks: Using the genetic algorithm with connectionist learning. In C. G. Langton, C. Taylor, J. D. Farmer, & S. Rasmussen (Eds.), *Artificial Life II.* Redwood City, CA: Addison-Wesley.

Booker, L. B, Goldberg, D. E., & Holland, J. H. (1989). Classifier systems and genetic algorithms. *Artificial Intelligence, 40*, 235–282.

Brodley, C. E. (1993). Addressing the selective superiority problem: Automatic algorithm/model class selection. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 17–24). San Mateo, CA: Morgan Kaufmann.

Buntine, W. (1989). Learning classification rules using Bayes. *Proceedings of the Sixth International Conference on Machine Learning* (pp. 94–98). San Mateo, CA: Morgan Kaufmann.

Catlett, J. (1991). Megainduction: A test flight. *Proceedings of the Eighth International Conference on Machine Learning* (pp. 589–604). San Mateo, CA: Morgan Kaufmann.

Clark, P., & Boswell, R. (1991). Rule induction with CN2: Some recent improvements. *Proceedings of the Sixth European Working Session on Learning* (pp. 151–163). London: Pitman.

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning, 3*, 261–283.

Cost, S., & Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning, 10*, 57–78.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*, 21–27.

DeGroot, M. H. (1986). *Probability and Statistics* (Second Edition). Reading, MA: Addison-Wesley.

Domingos, P. (1994). The RISE system: Conquering without separating. Submitted to the *Sixth IEEE International Conference on Tools with Artificial Intelligence.*

Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis.* New York: Wiley.

Golding, A. R., & Rosenbloom, P. S. (1991). Improving rule-based systems through case-based reasoning. *Proceedings of the Ninth National Conference on Artificial Intelligence* (pp. 22–27). Menlo Park, CA: AAAI Press.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning, 11*, 63–91.

Kolodner, J. (1993). *Case-Based Reasoning.* San Mateo, CA: Morgan Kaufmann.

Michalski, R. S., & Tecuci, G. (Eds.) (1994). *Machine Learning: A Multistrategy Approach.* San Mateo, CA: Morgan Kaufmann.

Michalski, R. S. (1983). A theory and methodology of inductive learning. *Artificial Intelligence, 20*, 111–161.

Michalski, R. S., Mozetic, I., Hong, J., & Lavrac, N. (1986). The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 1041–1045). Menlo Park, CA: AAAI Press.

Mitchell, T. M. (1980). The need for biases in learning generalizations. Technical report. New Brunswick, NJ: Rutgers University, Computer Science Department.

Muggleton, S., & Feng, C. (1990). Efficient induction of logic programs. *Proceedings of the First Workshop on Algorithmic Learning Theory* (pp. 368–381). Tokyo: Ohmsha.

Murphy, P., & Pazzani, M. (1994). Exploring the decision forest: An empirical investigation of Occam's razor in decision tree induction. *Journal of Artificial Intelligence Research, 1*, 257–275.

Murphy, P. M., & Aha, D. W. (1992). UCI repository of machine learning databases. Machine-readable data repository. Irvine, CA: University of California, Department of Information and Computer Science.

Niblett, T. (1987). Constructing decision trees in noisy domains. *Proceedings of the Second European Working Session on Learning* (pp. 67–78). London: Pitman.

Ourston, D., & Mooney, R. J. (1994). Theory refinement combining analytical and empirical methods. *Artificial Intelligence, 66*, 273–309.

Pagallo, G., & Haussler, D. (1990). Boolean feature discovery in empirical learning. *Machine Learning, 3*, 71–99.

Pazzani, M., & Kibler, D. (1992). The utility of knowledge in inductive learning. *Machine Learning, 9*, 57–94.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*, 81–106.

Quinlan, J. R. (1987a). Generating production rules from decision trees. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence* (pp. 304–307). San Mateo, CA: Morgan Kaufmann.

Quinlan, J. R. (1987b). Simplifying decision trees. *International Journal of Man-Machine Studies, 27*, 221–234.

Quinlan, J. R. (1993a). *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Quinlan, J. R. (1993b). Combining instance-based and model-based learning. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 236–243). San Mateo, CA: Morgan Kaufmann.

Riesbeck, C. K., & Schank, R. C. (1989). *Inside Case-Based Reasoning.* Hillsdale, NJ: Lawrence Erlbaum.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (Vol. 2). Cambridge, MA: MIT Press.

Salzberg, S. (1991). A nearest hyperrectangle learning method. *Machine Learning, 6*, 251–276.

Schaffer, C. (1994). A conservation law for generalization performance. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 259–265). San Mateo, CA: Morgan Kaufmann.

Smyth, P. R., Goodman, M., & Higgins, C. (1990). A hybrid rule-based/Bayesian classifier. *Proceedings of the Seventh European Conference on Artificial Intelligence* (pp. 610–615). Berlin: Springer-Verlag.

Stanfill, C., & Waltz, D. (1986). Toward memory-based reasoning. *Communications of the ACM, 29*, 1213–1228.

Towell, G. G., Shavlik, J. W., & Noordewier, M. O. (1990). Refinement of approximate domain theories by knowledge-based artificial neural networks. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 861–866). Menlo Park, CA: AAAI Press.

Utgoff, P. E. (1989). Perceptron trees: A case study in hybrid concept representations. *Connection Science, 1*, 377–391.

Appendix

This appendix lists the UCI repository file containing each dataset, together with any conversions performed and any other necessary information. Instance identification codes were deleted whenever present. The UCI repository is accessible by anonymous ftp from ics.uci.edu, subdirectory pub/machine-learning-databases.

AD: audiology/audiology.standardized.data

AN: annealing/anneal.data

BC: breast-cancer/breast-cancer.data

CE: credit-screening/crx.data

CH: chess/king-rook-vs-king-pawn/kr-vs-kp.data

DI: pima-indians-diabetes/pima-indians-diabetes.data

EC: echocardiogram/echocardiogram.data
Class is 2nd attribute, attributes 1 and 10-13 deleted, example with unknown class deleted.

GL: glass/glass.data

HD: heart-disease/cleve.mod
Last attribute deleted to yield a two-class problem.

HE: hepatitis/hepatitis.data

HO: horse-colic/horse-colic.data
Class is 24th attribute, attributes 3 and 25-28 deleted.

HY: thyroid-disease/hypothyroid.data

IR: iris/iris.data

LA: Quinlan's formatting of this dataset, distributed with C4.5, was used instead of the inadequate one in the UCI repository.

LC: lung-cancer/lung-cancer.data

LD: liver-disorders/bupa.data

LE: lenses/lenses.data

LI: led-display-creator/led-creator.c
Program run with the following parameters: 100 examples, seed = 1, 10% noise.

LY: lymphography/lymphography.data

MU: mushroom/agaricus-lepiota.data

PO: postoperative-patient-data/post-operative.data
Pseudo-discretized values converted to numeric (e.g. high, mid, and low become 1, 0 and -1).

PR: molecular-biology/promoter-gene-sequences/promoters.data

PT: primary-tumor/primary-tumor.data

SF: solar-flare/flare.data1
First attribute used as class.

SN: undocumented/connectionist-bench/sonar/sonar.all-data

SO: soybean/soybean-small.data

SP: molecular-biology/splice-junction-gene-sequences/splice.data

VO: voting-records/house-votes-84.data

WI: wine/wine.data

ZO: zoo/zoo.data