Machine Learning, 24, 141–168 (1996)

Unifying Instance-Based and Rule-Based Induction

PEDRO DOMINGOS pedrod@ics.uci.edu
Department of Information and Computer Science, University of California, Irvine, CA 92717

Editor: Raymond J. Mooney

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 article 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 3.1 system. In an extensive empirical study, RISE consistently achieves higher accuracies than state-of-the-art representatives of both its parent approaches (PEBLS and CN2), as well as a decision tree learner (C4.5). Lesion studies show that each of RISE's components is essential to this performance. Most significantly, in 14 of the 30 domains studied, RISE is more accurate 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 explicit or implicit creation of general concept or class descriptions from examples. Many induction problems can be described as follows. A *training set* of preclassified *examples* is given, where each example (also called *observation* or *case*) is described by a vector of *features* or *attribute values*, and the goal is to form a description that can be used to classify previously unseen examples with high accuracy. (Examples can also be described in other languages, like first-order logic (Quinlan, 1990), and other goals are often also important, like comprehensibility of the description.) 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, Kibler & Albert, 1991), Bayesian classification (Buntine, 1989), back-propagation (Rumelhart, Hinton & Williams, 1986), and genetic algorithms (Booker, Goldberg & Holland, 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, 1995), 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 (Schaffer, 1994a). 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 or a similar method before the system can be said to be doing its "best."

Another approach, known as multi-strategy learning (Michalski & 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 & Mooney, 1994; Towell & Shavlik, 1994; Pazzani & Kibler, 1992; see also Michalski & Tecuci, 1993). 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 example 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 below 100%, and the question of whether it can be improved, and if so how, is an open and important one.

Unfortunately, this approach has often been only moderately successful. The resulting algorithms are prone to be cumbersome, and often achieve accuracies that lie between those of their parents, instead of matching the highest. 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, 1994b) 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. 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 (Rao, Gordon & Spears, 1995). 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 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, are best suited to symbolic domains, and can often dispose easily of irrelevant attributes; but they can have difficulty with non-axis-parallel frontiers, and suffer from the *fragmentation* problem (i.e., the available data dwindles as induction progresses) and the *small disjuncts* problem (i.e., rules covering few training examples have a high error rate (Holte, Acker & Porter, 1989)). (The two paradigms also share a number of characteristics, of course, most notably the assumption that the example space contains large continuous regions of constant class membership—the similarity hypothesis (Rendell, 1986).)

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 from specific cases to the very general rules typically used (Riesbeck & Schank, 1989), and it deserves to be further explored.

This article describes and evaluates the RISE 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 increasingly 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 article 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 RISE with several current induction algorithms and investigating the sources of its power will be described. Finally, RISE will be placed in the context of related work and directions for future research will be pointed out.

2. Instance-based learning

Instance-based learning[1] (Cover & Hart, 1967; Duda & Hart, 1973; Aha *et al.*, 1991; Cost & Salzberg, 1993; Aha, in press) is founded on a direct application of the similarity assumption. In the simplest case, learning is performed by storing all the observed examples. A new example (or "test case") is classified by finding the nearest stored example according to some similarity function, and assigning the latter's class to the former. The stored examples used to classify new cases are referred to as *instances* or *exemplars*. 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 can lead to attributes with a large spread of values having undue weight in the result, compared to attributes with smaller spreads. A commonly used approach (see, e.g., Salzberg, 1991) is to normalize the difference by its largest observed value:

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

If there are a attributes, the distance between two instances $E_1 = (e_{11}, e_{12}, \ldots, e_{1a}, C_1)$ and $E_2 = (e_{21}, e_{22}, \ldots, e_{2a}, C_2)$ can then be defined as:

$$\Delta(E_1, E_2) = \sum_{i=1}^{a} \delta^s(e_{1i}, e_{2i}) \quad (2)$$

with $s = 1$ yielding city-block distance and $s = 2$ the square of Euclidean distance.

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

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

This measure is obviously less informative than its numeric counterpart, and, although it is appropriate in some cases, its use can lead to poor performance (Cost & Salzberg, 1993). A more sophisticated alternative consists of considering two symbolic values to be similar if they make similar predictions (i.e., if they correlate similarly with the class feature). This was first proposed by Stanfill and Waltz (1986) as part of their value difference metric (VDM) for a memory-based reasoner. Here we will consider a simplified version of the VDM, which defines the distance between two symbolic values as:

$$\delta(x_i, x_j) = SVDM(x_i, x_j) = \sum_{h=1}^{c} |P(C_h|x_i) - P(C_h|x_j)|^q \quad (4)$$

where c is the number of classes, C_h is the hth class, and q is a natural-valued parameter ($q = 1, 2, 3, \ldots$). 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 $\Delta(E_1, E_2)$ is computed as before. Different variants of this metric have been successfully used in pronunciation, molecular biology and other tasks (Stanfill & Waltz, 1986; Cost & Salzberg, 1993; Biberman, 1994).

IBL methods need to address the problem of sensitivity 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. Many approaches have been proposed to deal with this problem (Aha, 1990; Salzberg, 1991; Kelly & Davis, 1991; Wettschereck, 1994; Townsend-Weber & Kibler, 1994; Aha & Bankert, 1994). We have also found that use of the SVDM 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 $P(C_h|x_i)$ will be roughly the same for all values x_i of an irrelevant attribute, leading to zero distance (i.e., equivalence) between them.

Another issue in IBL methods is their sensitivity to noise. Incorrect instances are liable to create a region 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." Cameron-Jones (1992) uses instead a criterion based on the minimum description length principle to decide which instances to retain. PEBLS (Cost & Salzberg, 1993) assign weights to instances, making their apparent distance to new examples increase with their misclassification rate.

Given two instances of different 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 when few instances are present (Aha *et al.*, 1991). The introduction of weights further increases this complexity, turning the hyperplanes into hyperquadrics (Cost & 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 & Hart, 1973). The class assigned is then that of the majority of those k neighbors, or the class receiving the most votes, with a neighbor's vote decreasing with its distance from the test example.

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 the example space, whereas a stored instance represents the entire region that it wins over in the competition with other instances.

3. Rule induction

Rule induction algorithms (Michalski, 1983; Michalski, Mozetic, Hong & Lavrac, 1986; Clark & Niblett, 1989; Rivest, 1987) typically employ a set covering or "separate and conquer" approach to induction. This 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 summarized in pseudo-code in Table 1. Some definitions are in order. A *rule* is composed of a *consequent* and an *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* examples.

The "best" rule in each covering cycle (see Table 1) may also be found by beam search (e.g., Clark & Niblett, 1989). In this case a list of the b best rule bodies found so far is maintained, instead of a single body. At each step, specialization of each of those bodies with each possible antecedent is attempted, and the best b bodies are selected to continue the search. At the end the best rule body overall is selected.

The choice of evaluation heuristic H is of some importance to the algorithm's performance. Given a rule, H should increase with $n_{\oplus}$, the number of positive examples that satisfy the rule, and decrease with $n_{\ominus}$, the number of negative examples that satisfy it. The

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 $\oplus = \oplus - \{E \in \oplus \mid R \text{ covers } E\}$.
    Let $RS = RS \cup \{R\}$.
  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.

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

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

where c is the number of classes. 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 & 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 such rule, its class is assigned to the example. If there are 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, then two strategies are possible. One is to order the rules into a "decision list," and select only the first rule that fires (Rivest, 1987). 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 & Boswell, 1991). Other voting schemes are possible (e.g., Michalski *et al.*, 1986). The use of unordered rules has been found to generally produce higher accuracy (Clark & 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, 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 & 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. Also, 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 inaccurate definitions when boundaries are non-axis-parallel 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 fragmentation effect may cause later rules, and later antecedents within each rule, to be induced with insufficient statistical support, leading to greater noise sensitivity and missing or incorrect rules/antecedents. This in turn aggravates the "small disjuncts problem," first observed by Holte *et al.* (1989): rules covering few training examples (five or less, say) tend to be highly error-prone, but removing them often increases the global error even further.

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 3.1 system (Rule Induction from a Set of Exemplars; earlier versions are described in (Domingos, 1994) and (Domingos, 1995a)). 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 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 instance is simply a rule in which the consequent is the instance'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 article, the word "rule" is used to refer indiscriminately to exemplars and to rules of the more general type.

RISE classifies a new example by assigning it the class of the nearest rule in the knowledge base. The distance between a rule and an example is defined as follows. 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. If there is no condition on i, A_i = True, otherwise A_i is $e_i = r_i$ if i is symbolic and A_i is $r_{i,lower} \leq e_i \leq r_{i,upper}$ if i is numeric. The distance $\Delta(R, E)$ between R and E is then defined as:

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

where s is a natural-valued parameter ($s = 1, 2, 3, \ldots$), and the component distance $\delta(i)$ for the ith attribute is:

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

where in turn $SVDM(r_i, e_i)$ is the simplified value difference metric as defined in Equation 4, and:

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

$e_{i,max}$ and $e_{i,min}$ being respectively the maximum and minimum values for the attribute found in the training set. This definition of $\delta_{num}(i)$ is also used in EACH (Salzberg, 1991).

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 SVDM to all other values of the attribute is computed and used. In the context of VDM-type metrics, 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 distinct from that value if it does not.

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 can 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 described in (Domingos, 1995b).

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 and conflict resolution policy just described. A rule can cover an example and not win it. The extension of a rule is constituted by all the points in the example 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 denoted by $Acc(RS)$). There is no need to use the Laplace correction when comparing the accuracy of different rule sets on a training set, because the denominator of the accuracy (i.e., the number of examples matched) is exactly the same for all rule sets (being the size of the training set).

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 fragmentation 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 does not yet cover, and attempts to minimally generalize itself to cover it (see Table 2). 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 instance is a candidate rule). In the worst case no generalizations are accepted, and the final rule set is still the

Table 2. 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, $e_i = r_i$, or $r_{i,lower} \leq e_i \leq r_{i,upper}$.

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 A_i = True.
  Else if i is numeric and $e_i > r_{i,upper}$ then $r_{i,upper} = e_i$.
  Else if i is numeric and $e_i < r_{i,lower}$ then $r_{i,lower} = e_i$.

Table 3. 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.

training set, leading to a pure instance-based 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 3 summarizes this process in pseudo-code. In each cycle the new rule set is adopted even 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.

Table 2 shows in pseudo-code how a rule is minimally generalized to cover an example 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. 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: a missing numeric value is considered to match any other value, and a missing

symbolic value in the example or in the rule (but not both) causes the corresponding condition to be dropped.

Accuracy is measured using a leave-one-out methodology: when attempting to classify an example, the corresponding rule is left out, unless it already covers other examples as well. 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 (or all but one) against all examples, leading to a clearly unacceptable time cost. Fortunately, at each step 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, and is therefore negligible. With this information, all that is necessary when a rule is generalized is 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.

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 worst-case efficiency is comparable to that of other rule induction algorithms. Let e represent the number of examples in the training set, a the number of attributes used to describe each, v_s (v_n) the average number of observed values per symbolic (numeric) 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 SVDM distances, taking $O(ea + av_s^2c)$ (ea to run through all the examples, for each one noting the correspondence between each attribute's value and the class, and av_s^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 $O(e^2a + av_s^2c)$.

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 3) 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. However, this is extremely unlikely. The two policies were empirically compared (see Domingos, 1995b), 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 $e \geq v_s$, 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 CN2 and AQ-style algorithms is $O(be^2a^2)$, where b is the beam size, an integer-valued internal parameter of the algorithm (Clark & Niblett, 1989).[2] The worst-case complexity of growing a decision tree is $O(ea^2)$ when only symbolic attributes are present (Utgoff, 1989a), but becomes quadratic or worse in e with numeric ones, mainly due to the need for repeated sorting operations (Catlett, 1991). The final pruning stage used in C4.5 and several rule induction systems may also in general be worse than quadratic in e (Cohen, 1995). Thus RISE's worst-case time complexity is comparable to that of other rule and decision tree learners. 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_n, since the single-step removal of a condition may now be replaced by at most $O(v_n)$ steps of expanding the corresponding interval. Again, in practice only a small number of steps may actually be required. However, in the case of real-valued attributes for which arbitrarily fine distinctions are possible, and therefore as many different values as there are training examples may occur, this can lead to some inefficiency. This problem is only potentially significant in large datasets, since the number of observed values of an attribute is bounded from above by the number of training examples. To obviate it, in datasets with more than 3000 examples the generalization of a rule to cover an example (Table 2) is adapted as follows. When extending an interval to include an example value above it (or below), the interval's upper (lower) limit is not set to the example's value, but to that value plus (minus) a delta which is set by default to 5% of the maximum range the value was observed to vary over. In addition to potentially reducing the running time of the algorithm, this policy can have a positive effect on its accuracy, by avoiding overfitting. This was indeed observed in the empirical study described below.

RISE has been optimized with respect to running time in two additional ways: pruning and windowing. Pruning is performed by sometimes cutting short the computation of the distance between a rule and an example; this is akin to reducing search by means of branch-

and-bound techniques. This is possible in two situations (see Table 3). When a rule searches for its nearest example, it can discard a candidate as soon as its distance becomes larger than the shortest one found so far. Similarly, when a tentatively-generalized rule searches for the examples it now wins, its distance to each example needs to be computed only until it becomes larger than the current winning rule's one. This form of pruning has no effect on the algorithm's output, and in general will also not change its worst-case time complexity, but can significantly reduce its average running time.

In datasets with more than 3000 examples, windowing is used (Catlett, 1991; Quinlan, 1993a). This proceeds as follows. Initially, only $2\sqrt{e}$ examples randomly extracted from the training set are used for learning. If the remaining training examples are correctly classified by the resulting rule set, this set is output. Otherwise, the misclassified examples are added to the initial example set, and this process repeats until it produces no improvement in accuracy, or up to a prespecified number of times (5 by default). In the best case, only $O(\sqrt{e})$ examples are used, and the algorithm becomes linear in the training set size. In the worst case, the window grows to include the entire training set, and the process is more costly than learning directly on that set. Thus windowing may or may not decrease running time. Its effect on accuracy can also be positive or negative. A more detailed discussion of these issues in the context of decision tree induction can be found in (Catlett, 1991) and (Quinlan, 1993a). In the empirical study described below, windowing improved RISE's running time in all the datasets where it was used (those with more than 3000 examples), sometimes by a large factor, while improving accuracy in all but one (where it had no significant effect). Windowing thus appears to be substantially more useful in RISE than in decision tree induction. This may be due to several factors, including RISE's lower sensitivity to the global proportions of different classes, and its higher resistance to the fragmentation problem, which enables it to correctly approximate class frontiers using fewer examples.

6. Empirical study

An extensive empirical study was carried out with the twin goals of comparing RISE's performance to that of previous approaches, and determining the role of its main components in that performance. The remainder of this section describes the characteristics and reports the results of this study.

6.1. Experimental design

RISE was compared with a representative of each of its parent approaches: PEBLS for IBL (Cost & Salzberg, 1993), and CN2 for rule induction (Clark & Niblett, 1989). 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 Section 4). A recent version of CN2 (6.1) was used, one incorporating Laplace accuracy and unordered rules (Clark & Boswell, 1991). To gauge its position in the overall spectrum

of induction methods, RISE was also compared with a system that learns rule sets by way of decision trees, C4.5/C4.5RULES (Quinlan, 1993a). 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.

Thirty datasets 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 datasets; small, medium and large datasets; datasets of varying difficulty, as expressed in the highest accuracy previously achieved; and datasets from a wide range of application areas. All datasets were drawn from the UCI repository (Murphy & Aha, 1995), and are obtainable by anonymous ftp from ics.uci.edu, subdirectory pub/machine-learning-databases. Table 4 lists the datasets used, and summarizes some of their main characteristics.[3] Datasets included in the listing of empirical results in (Holte, 1993) are referred to by the same codes.

In the first phase of the study, the first 15 datasets in Table 4 (from breast cancer to wine) were used to fine-tune the algorithms, choosing by 10-fold cross-validation the most accurate version of each. Since a complete factor analysis would be too expensive, the relevant parameters were instead varied one at a time, starting with the ones thought to be most critical, and selecting the best value for each parameter before going on to the next one. When no clear differences were present, the default was retained. For RISE this process is described in more detail in (Domingos, 1995b), and resulted in the system described in the previous sections, with $q = 1$ (Equation 4), $s = 2$ (Equation 7), and global stopping (Section 5). In PEBLS, 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 up to a maximum of m, and the optimum m found was 15. All other parameters retained their default values. In CN2, the only non-default choice produced was that of accuracy instead of Laplace accuracy for the evaluation function.[4] In C4.5, a major question is whether to output trees or rules. C4.5RULES produced slightly better results, and was the version chosen. It also has the advantage of being the one most directly comparable to RISE. The non-default choices made were: use windowing (growing 10 trees, the default), require a minimum of 4 examples (instead of 2) in two branches of a test, and use a confidence level of 37.5% for rule pruning (instead of 25%).

The fine-tuned algorithms were then tested on the remaining 15 datasets in Table 4 (from audiology to zoology). Note that this procedure is somewhat unfavorable to RISE, since some of these domains were previously used in the development of the other algorithms, as reported in the references above. Each dataset was randomly divided 50 times into a training set containing two-thirds of the examples, and a testing set containing the remainder. All of the algorithms were trained on each of the 50 training sets and tested on the corresponding testing set. The results reported are averages of these 50 runs. For the sake of completeness, the algorithms were also rerun in these conditions on the 15 tuning datasets.

Table 4. Datasets 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; percentage of missing values; and whether or not the dataset includes inconsistent examples (i.e., identical examples with different classes).

Domain	Code	Exs.	Atts.	Num.	Classes	Missing	Incons.
Breast cancer	BC	286	9	4	2	0.3	Yes
Credit screening	CE	690	15	6	2	0.6	No
Chess endgames	CH	3196	36	0	2	0.0	No
Pima diabetes	DI	768	8	8	2	0.0	No
Hepatitis	HE	155	19	6	2	5.7	No
Iris	IR	150	4	4	3	0.0	No
Labor negotiations	LA	57	16	8	2	35.7	No
Lung cancer	LC	32	56	0	3	0.3	No
Liver disease	LD	345	6	6	2	0.0	No
Contact lenses	LE	24	4	0	3	0.0	No
Lymphography	LY	148	18	3	4	0.0	No
Primary tumor	PT	339	17	0	21	3.9	Yes
Soybean	SO	47	35	0	4	0.0	No
Voting records	VO	435	16	0	2	5.6	No
Wine	WI	178	13	13	3	0.0	No
Audiology	AD	200	69	0	24	2.1	No
Annealing	AN	798	38	9	5	64.9	No
Echocardiogram	EC	131	7	6	2	4.4	Yes
Glass	GL	214	9	9	6	0.0	No
Heart disease	HD	303	13	6	2	0.2	No
Horse colic	HO	300	22	7	2	24.3	Yes
Thyroid disease	HY	3163	25	7	2	6.7	Yes
LED	LI	100	7	0	10	0.0	Yes
Mushroom	MU	8124	22	0	2	1.4	No
Post-operative	PO	90	8	8	3	0.4	Yes
DNA promoters	PR	106	57	0	2	0.0	No
Solar flare	SF	323	12	3	6	0.0	Yes
Sonar	SN	208	60	60	2	0.0	No
Splice junctions	SP	3190	60	0	3	0.0	Yes
Zoology	ZO	101	16	1	7	0.0	No

6.2. *Accuracy comparisons*

The average accuracies and sample standard deviations obtained are presented in Table 5. Superscripts indicate confidence levels for the difference between RISE and the corresponding algorithm, using a one-tailed paired t test. The first half of the table shows results on tuning datasets, and the second half shows results on test datasets.

Although many results in individual datasets are interesting in themselves, these tables are more easily understood if their results are summarized in a few global measures of comparative performance. Five such measures and their values are presented in Table 6. They can be described and interpreted as follows.

Table 5. Empirical results: average accuracies and standard deviations. Superscripts denote confidence levels: 1 is 99.5%, 2 is 99%, 3 is 97.5%, 4 is 95%, 5 is 90%, and 6 is below 90%.

Domain	RISE	Default	PEBLS	CN2	C4.5
BC	67.7±4.5	68.5±6.8^{6}	64.9±4.0^{1}	65.9±4.7^{1}	67.9±5.1^{6}
CE	83.3±2.4	56.7±3.7^{1}	81.6±2.0^{1}	82.3±2.2^{1}	84.5±2.5^{1}
CH	98.2±0.6	52.3±1.8^{1}	96.8±0.7^{1}	98.7±0.7^{1}	99.2±0.2^{1}
DI	70.4±2.5	65.4±2.4^{1}	70.9±2.6^{6}	73.6±2.4^{1}	73.1±2.5^{1}
HE	78.3±5.7	79.1±3.6^{6}	80.3±5.1^{3}	80.9±3.6^{1}	81.0±5.3^{2}
IR	94.0±2.8	26.3±3.8^{1}	93.2±3.5^{4}	90.8±4.4^{1}	93.5±2.8^{6}
LA	87.2±7.8	66.1±10.2^{1}	91.1±5.1^{1}	82.8±7.6^{1}	81.4±6.1^{1}
LC	44.7±16.4	24.7±15.5^{1}	42.0±15.2^{6}	32.7±13.6^{1}	44.5±14.1^{6}
LD	62.4±4.6	57.8±3.2^{1}	61.2±4.2^{5}	66.8±4.7^{1}	65.1±4.5^{1}
LE	77.2±12.8	60.8±13.1^{1}	72.5±12.6^{1}	67.8±12.4^{1}	63.7±18.4^{1}
LY	78.7±5.9	55.5±6.9^{1}	81.7±5.4^{1}	80.2±5.5^{5}	76.6±5.7^{3}
PT	40.3±4.8	24.3±3.4^{1}	31.4±4.1^{1}	41.4±5.3^{5}	38.9±4.9^{4}
SO	100.0±0.0	25.9±16.2^{1}	100.0±0.0^{6}	97.3±4.9^{1}	97.4±3.1^{1}
VO	95.2±1.5	60.7±3.0^{1}	94.6±1.3^{3}	95.5±1.5^{5}	95.4±1.5^{6}
WI	96.9±2.0	36.9±8.6^{1}	96.5±2.1^{5}	90.5±5.5^{1}	93.0±4.2^{1}
AD	77.0±5.3	20.8±3.6^{1}	75.8±5.2^{5}	63.0±6.6^{1}	66.9±6.0^{1}
AN	97.4±0.9	76.2±2.2^{1}	98.8±0.8^{1}	92.9±2.1^{1}	93.4±1.7^{1}
EC	64.6±7.0	68.1±6.9^{1}	63.0±5.4^{5}	67.9±6.0^{1}	65.0±6.4^{6}
GL	70.6±5.8	31.4±5.0^{1}	66.7±6.2^{1}	55.0±6.9^{1}	66.4±7.4^{1}
HD	79.7±3.8	55.1±3.8^{1}	78.8±3.8^{5}	78.5±3.9^{5}	77.3±3.5^{1}
HO	82.6±3.9	64.1±3.8^{1}	76.1±4.0^{1}	82.2±3.6^{6}	81.0±3.8^{2}
HY	97.5±0.4	95.3±0.5^{1}	97.3±0.5^{1}	98.3±0.5^{1}	99.2±0.2^{1}
LI	59.9±7.2	7.3±3.2^{1}	53.0±7.0^{1}	57.5±7.7^{2}	57.1±8.1^{2}
MU	100.0±0.2	51.8±0.9^{1}	100.0±0.0^{6}	100.0±0.0^{6}	100.0±0.0^{6}
PO	64.1±7.0	71.0±6.2^{1}	58.3±7.9^{1}	59.4±7.7^{1}	67.3±9.7^{3}
PR	86.8±5.2	42.1±5.3^{1}	90.6±5.2^{1}	72.0±8.9^{1}	80.6±8.5^{1}
SF	71.6±3.7	25.4±4.1^{1}	68.2±3.5^{1}	70.7±3.4^{4}	71.5±3.7^{6}
SN	77.9±9.3	51.0±7.0^{1}	75.0±5.6^{3}	63.0±5.5^{1}	69.6±8.0^{1}
SP	93.1±1.6	52.2±1.7^{1}	94.3±0.6^{1}	90.9±1.0^{1}	93.2±1.1^{6}
ZO	93.9±3.7	40.5±6.3^{1}	94.9±4.2^{3}	92.4±5.5^{3}	91.2±5.1^{1}

Table 6. Summary of accuracy results.

	Test datasets				All datasets			
Measure	RISE	PEBLS	CN2	C4.5	RISE	PEBLS	CN2	C4.5
No. wins	-	10-4	12-2	10-4	-	20-8	20-9	18-11
No. signif. wins	-	7-4	10-2	9-2	-	14-7	18-6	15-7
Wilcoxon test	-	96.0	99.6	98.0	-	98.0	99.6	98.0
Average	81.1	79.4	76.2	78.6	79.7	78.3	76.4	77.8
Score	46.5	34.5	27.5	33.5	86.0	65.0	63.5	72.5

- *Number of wins.* The first and most obvious test is simply to count the number of datasets 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 6 shows this: for example, RISE performed better than PEBLS in 10 test datasets, and worse in 4 (there was 1 draw). As can be seen, RISE outperformed every other algorithm in this respect by a ratio of roughly 2 to 1.

- *Number of significant wins.* The previous measure is clearly imperfect, for some of the differences can be very small and of no real significance. A good alternative is then to count only those datasets where the difference was significant at a confidence level of 95% or higher (see Table 5). This yields roughly similar ratios, with all values somewhat reduced as might be expected, confirming the previous observation that RISE performs substantially better than every other algorithm.

- *Wilcoxon 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 15 (or 30) domains used in this study as a sample of size 15 (or 30) taken from the set of real-world domains, consider the accuracy difference in each domain as a sample of a random variable, and test the hypothesis of whether this difference is in general positive. The Wilcoxon signed-ranks test is a procedure that addresses this issue (DeGroot, 1986). It takes into account the signs of the observed differences, and their ranking (i.e., the largest difference counts more than the second largest one and so forth, but it does not matter by how much). In each case, the results support the hypothesis that RISE is a more accurate algorithm with a confidence in excess of 95%. It 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 & 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.

The same general pattern typically emerges if the results are broken down by dataset size (small, medium and large, i.e., $e < 100$, $100 \leq e < 1000$, and $e \geq 1000$), dataset 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 datasets is very small, but RISE's excellent results clearly hold across the board.

A very significant observation is that in 14 datasets 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 9 of those datasets this is true with a confidence level of 95% or higher, using a one-tailed paired t test (see Table 5). This shows that a significant synergy can be obtained by combining multiple empirical learning approaches.

6.3. *Further study*

The empirical results just described lead to the conclusion that RISE represents a significant advance in the state-of-the-art in empirical learning. However, we would also like to know what factors RISE's higher accuracy should be attributed to. To answer this, lesion studies were conducted, and additional aspects of the algorithm's performance were measured. The results are reported in Table 7. Superscripts indicate confidence levels for the accuracy differences between systems, again using a one-tailed paired t test. The first four columns compare the full system's accuracy with that obtained using lesioned versions, and are summarized in Table 8. The last four columns all refer to the full RISE system, and show, respectively: the percentage of test cases that were not matched by any rule, but had a single closest rule, or for which all equally close rules were of the same class (No/One); the percentage not matched by any rule, and for which there were equally close rules of more than one class (No/Multi); the percentage matched by only one rule, or rules of only one class (One); and the percentage matched by rules of more than one class (Multi). These observations aid in interpreting the lesion study results.

The first specific question addressed was whether there is any gain in the rule induction process (i.e., whether RISE constitutes an improvement over pure instance-based learning). The "IBL" column in Table 7 reports the accuracies obtained by the initial, ungeneralized instance set, and shows that generalization often produces significant gains in accuracy,

Table 7. Empirical results: lesion studies and performance monitoring. Superscripts denote confidence levels: 1 is 99.5%, 2 is 99%, 3 is 97.5%, 4 is 95%, 5 is 90%, and 6 is below 90%.

	Accuracy of subsystem				Match type frequency			
Domain	RISE	IBL	Rules	No tie-b.	No/One	No/Multi	One	Multi
BC	67.7	65.1[1]	69.0[4]	68.7[1]	33.4	1.5	59.1	6.0
CE	83.3	81.3[1]	66.9[1]	83.2[6]	51.9	0.0	46.9	1.1
CH	98.2	91.9[1]	91.9[1]	98.0[1]	27.6	0.1	71.4	0.9
DI	70.4	70.3[6]	66.2[1]	70.5[1]	70.9	0.1	27.4	1.6
HE	78.3	78.4[6]	79.6[5]	78.5[5]	55.3	0.1	43.1	1.5
IR	94.0	94.7[3]	71.0[1]	94.0[6]	45.9	0.0	54.0	0.2
LA	87.2	90.8[1]	73.7[1]	87.1[6]	51.8	0.2	46.6	1.4
LC	44.7	42.0[6]	26.5[1]	44.2[5]	88.2	0.4	9.1	2.4
LD	62.4	60.9[4]	62.1[6]	62.3[6]	72.7	0.2	24.0	3.0
LE	77.2	72.5[1]	64.8[1]	75.8[3]	13.2	1.5	83.0	2.2
LY	78.7	82.0[1]	70.1[1]	78.2[3]	42.9	0.4	53.9	2.8
PT	40.3	34.3[1]	33.5[1]	37.0[1]	34.2	4.1	41.6	20.1
SO	100.0	100.0[6]	85.1[1]	100.0[6]	16.9	0.0	83.1	0.0
VO	95.2	94.6[3]	83.7[1]	94.2[1]	7.4	0.1	90.3	2.1
WI	96.9	95.1[1]	51.5[1]	96.9[6]	78.1	0.0	21.9	0.0
AD	77.0	75.8[5]	55.1[1]	76.2[1]	53.6	1.6	43.0	1.8
AN	97.4	97.7[4]	77.7[1]	97.2[1]	24.1	0.0	75.6	0.2
EC	64.6	59.2[1]	65.7[6]	64.5[6]	70.1	0.1	26.8	3.0
GL	70.6	68.3[2]	47.3[1]	70.4[2]	71.5	0.0	27.2	1.3
HD	79.7	77.8[1]	64.7[1]	79.7[6]	63.1	0.0	34.8	2.1
HO	82.6	76.6[1]	79.0[1]	81.7[1]	39.7	0.2	55.4	4.6
HY	97.5	94.1[4]	84.8[1]	97.5[6]	40.7	0.1	58.3	0.9
LI	59.9	55.9[1]	52.7[1]	49.7[1]	14.7	4.5	47.3	33.5
MU	100.0	97.5[6]	100.0[6]	99.8[1]	7.3	0.0	92.5	0.2
PO	64.1	59.1[1]	70.9[1]	65.9[1]	36.5	12.2	44.7	6.6
PR	86.8	90.6[1]	68.4[1]	85.9[4]	59.7	0.0	35.8	4.5
SF	71.6	71.1[6]	65.5[1]	68.1[1]	16.7	2.9	59.9	20.4
SN	77.9	83.8[1]	52.9[1]	77.9[6]	95.6	0.0	4.3	0.1
SP	93.1	87.8[1]	75.9[1]	92.1[1]	67.4	0.0	29.9	2.7
ZO	93.9	94.5[4]	81.0[1]	93.7[4]	17.5	0.0	81.9	0.5

Table 8. Summary of lesion study results.

	Test datasets				All datasets			
Measure	RISE	IBL	Rules	No t.-b.	RISE	IBL	Rules	No t.-b.
No. wins	-	11-4	12-2	11-1	-	21-8	25-4	20-4
No. signif. wins	-	8-4	12-1	10-1	-	16-7	24-2	15-3
Wilcoxon test	-	97.0	99.8	99.0	-	99.5	99.9	99.8
Average	81.1	79.3	69.4	80.0	79.7	78.1	67.9	79.0

while seldom having a negative effect. Not surprisingly, the results are similar to those of RISE vs. PEBLS.

The converse question is whether the instance-based component is really necessary. Simply assigning examples not covered by any rule to a default class, as done in most rule induction systems, might be sufficient. The "Rules" column in Table 7 shows the results obtained using this policy, and confirms the importance of "nearest-rule" classification in RISE. The sum of the "No" columns in the right half of Table 7 is the percentage of test cases assigned to the default class. This is often very high, the more so because RISE tends to produce rules that are more specific than those output by general-to-specific inducers. The use of nearest-rule is thus essential. Note that the results reported in the "Rules" column are for applying the default rule during both learning and classification; applying it exclusively during classification produced only a slight improvement.

Another important component of RISE whose utility needs to be determined is the conflict resolution policy, which in RISE consists of letting the tied rule with the highest Laplace accuracy win. This was compared with simply letting the most frequent class win ("No tie-b." column in Table 7). The sum of the "Multi" columns in the right half of Table 7 is the percentage of cases where tie-breaking is necessary. This is typically small, and the increase in accuracy afforded by RISE's conflict resolution strategy is correspondingly small (0.7% on average, for all datasets). However, this increase is consistently produced, as evinced by the fact that RISE is more accurate than its lesioned version with a 99.8% confidence by the Wilcoxon test.

Taken together, the lesion studies show that each of RISE's components is essential to its performance, and that it is their combination in one system that is responsible for the excellent results obtained by RISE *vis-à-vis* other approaches.

The question now becomes one of why RISE's combined approach is superior to the individual ones, and under what other conditions the combined approach will lead to superior performance. This was studied by conducting experiments in artificial domains. Space limitations preclude full reporting of these studies here, but the main results can be summarized as follows (see Domingos, 1995c, and Domingos, in press, for a more detailed treatment).

Compared to IBL systems, RISE draws strength from its ability to perform context-sensitive feature selection. Conventional feature selection methods, and VDM-style metrics, are only able to weed out features that are globally irrelevant. RISE, like rule and decision tree learners, can retain different sets of features in different parts of the example space. Experiments correlating accuracy with a measure of context dependency of feature relevance showed the advantage of RISE's feature selection strategy increasing with this dependency. RISE's bias will be inappropriate when this dependency is absent, and can hurt performance if the datasets are also small and noisy. An additional hypothesis to explain RISE's advantage vs. "pure" IBL is that, because of its ability to simplify frontiers in the example space through abstraction, it can detect simple frontiers more easily when they exist.

A number of IBL algorithms, including PEBLS, discretize numeric features and then apply a VDM-style metric to the resulting descriptions. This improves their ability to cope with irrelevant features, since the VDM for two values of an irrelevant feature will tend to be zero while Euclidean distance may be arbitrarily large. However, it also loses one

of IBL's main advantages: its ability to form non-axis-parallel frontiers in instance space. Ting (1994) found that this approach improved accuracy in a number of domains, which can be attributed to the first effect having prevailed. In other domains the opposite was observed. RISE overcomes this trade-off: it is still able to form non-axis-parallel frontiers through its use of Euclidean distance, and at the same time it is able to overcome irrelevant features by generalizing or entirely dropping them.

Compared to "divide and conquer" or "separate and conquer" systems like C4.5 and CN2, RISE's higher accuracies can be hypothesized to be due to the following factors. RISE is able to form complex non-axis-parallel frontiers, and is thus at an advantage when these are appropriate. Its "conquering without separating" induction strategy makes it less sensitive to the fragmentation problem, and its specific-to-general search direction allows it to identify small exception regions that will go undetected (or be more poorly identified) by general-to-specific learners. A study designed to exhibit conditions where each bias would be superior by correlating accuracy with concept specificity confirmed RISE's advantage when concepts are highly specific, but failed to produce the opposite result for general concepts, with both noise-free and noisy datasets. The question of what exact conditions will favor algorithms like C4.5 and CN2 over RISE thus remains a topic for further research.

6.4. Space and time comparisons

Besides accuracy, two variables of interest are output size and running time. These are listed in Table 9. Output size is measured by counting one unit for each antecedent of a rule, plus one for the consequent. The size for PEBLS is simply that of the the training set (i.e., $e(a + 1)$, in the notation of Section 5). Running times were obtained on a Sun 670 machine, with all algorithms implemented in C and similarly compiled. All values shown are averages of 50 runs.

With respect to output size, RISE occupies an intermediate position between the IBL and rule induction representatives. It obtains large savings compared to PEBLS, and these increase with training set size (see, e.g., the MU dataset). We can thus expect RISE to be at a significant advantage relative to unedited IBL methods in applications where a large number of examples is available, and the target concept is comparatively simple. On the other hand, RISE's output is still generally much larger than CN2's. The number of rules produced (not shown) is typically similar, but RISE's rules tend to be substantially more specific; this is consistent with the search direction and best-match policy it uses. An important fact is that, if rules that do not win any training examples are discarded, the resulting decrease in RISE's accuracy is minimal (0.05% on average) and has practically no effect on the comparison with CN2, but the output size is reduced by 50% or more, and the average number of rules becomes smaller than CN2's in every domain except VO (where it is about the same). Thus, when simplicity of the output is a goal, RISE can often come quite close to CN2 without compromising accuracy. However, the simplest results are always those obtained by C4.5RULES. (The pruned tree sizes (not shown) are comparable to those of CN2's and RISE's pruned rule sets.)

With respect to running time, RISE is slower than PEBLS on most datasets. This is to be expected, since RISE essentially does everything that PEBLS does, and more. More

Table 9. Empirical results: output size and running time (in minutes and seconds).

	Output size				Running time			
Domain	RISE	PEBLS	CN2	C4.5	RISE	PEBLS	CN2	C4.5
BC	1045.4	1910	410.2	17.5	0:15.11	0:03.16	0:22.86	1:31.41
CE	4110.9	7392	679.4	51.0	4:31.25	0:16.11	2:05.30	2:53.67
CH	3063.8	79217	843.1	96.2	10:39.67	11:59.41	6:41.85	4:00.51
DI	3048.8	4626	802.0	69.3	4:15.37	0:12.30	2:13.93	4:02.76
HE	1231.9	2060	112.8	20.5	0:10.93	0:02.32	0:07.42	0:08.24
IR	382.9	500	63.2	9.9	0:05.42	0:01.74	0:03.01	0:02.09
LA	305.7	646	42.3	10.0	0:01.30	0:01.60	0:02.42	0:01.96
LC	403.3	1197	39.6	7.8	0:00.69	0:01.70	0:04.64	0:02.07
LD	1156.8	1617	436.1	63.6	1:31.02	0:02.82	0:28.21	1:06.12
LE	16.7	80	25.5	6.1	0:00.13	0:01.78	0:00.55	0:01.31
LY	756.4	1881	131.7	25.4	0:04.88	0:02.21	0:07.77	0:06.07
PT	2076.7	4086	653.6	61.1	0:14.38	0:06.07	1:58.39	1:55.04
SO	106.2	1116	18.8	9.9	0:00.50	0:01.64	0:01.46	0:01.74
VO	541.0	4947	113.2	15.9	0:27.71	0:07.07	0:06.47	0:03.41
WI	896.3	1666	87.6	14.3	0:15.31	0:02.77	0:10.30	0:11.18
AD	4165.8	9380	236.4	40.7	0:11.20	0:07.14	1:10.47	0:24.18
AN	13527.3	20826	420.4	62.0	4:26.07	1:46.58	4:34.74	2:46.27
EC	550.2	696	137.7	19.7	0:04.84	0:02.70	0:07.16	0:13.05
GL	696.2	1430	245.8	47.0	0:11.03	0:02.75	0:28.22	1:41.71
HD	1460.5	2842	320.8	42.9	0:26.35	0:03.71	0:24.85	1:53.25
HO	2111.6	4623	268.6	19.4	1:39.64	0:05.77	1:42.63	0:24.29
HY	7465.9	55094	430.7	14.5	14:45.96	8:36.18	11:53.69	1:54.54
LI	217.3	536	125.8	40.3	0:00.60	0:01.68	0:02.42	0:02.89
MU	398.1	125189	176.8	38.6	10:07.22	45:37.28	4:16.37	18:29.79
PO	470.3	540	132.9	4.5	0:01.49	0:01.46	0:05.28	0:02.53
PR	1198.6	4118	82.5	16.0	0:07.85	0:02.94	0:15.03	0:06.28
SF	1316.2	2808	326.3	35.2	0:08.97	0:04.19	0:20.03	0:28.49
SN	3756.4	8479	224.0	33.9	2:35.81	0:09.76	2:39.08	5:15.54
SP	8350.7	130357	1360.5	211.3	51:28.15	19:25.15	32:15.24	24:20.94
ZO	196.2	1139	41.3	20.0	0:01.11	0:01.76	0:02.03	0:02.16

surprising is the fact that RISE is still faster than PEBLS on eight datasets, including several purely symbolic ones. This suggests that PEBLS's implementation could be further optimized. RISE is typically faster than CN2, but this advantage tends to be concentrated in the smaller datasets, raising the possibility that CN2 will be at a considerable advantage in larger datasets than those used here, provided the concept description is still sufficiently simple (if not, CN2's time will grow to the worst-case bound discussed in Section 5, equivalent to RISE's). RISE is also typically faster than C4.5 in this comparison, but this is due to the fact that the windowing option is being used in C4.5; even though each tree is typically grown faster with windowing, this gain tends to be swamped by the fact that 10 trees are being grown.

We conclude that, for datasets of very large size, and/or when comprehensibility is paramount, a system like C4.5 will still be the first choice. If, on the other hand, ac-

curacy is the dominant goal, the results of previous subsections indicate that RISE may be a more appropriate algorithm to use.

7. Related work

The RISE approach should be seen in the context of previous work in inductive learning and related areas.

The AQ15 system employed best-match classification (Michalski *et al*, 1986; a more recent version is AQ17-HCI (Wnek & Michalski, 1994)). It differed from RISE in that it was a general-to-specific, separate-and-conquer system that learned purely logical rules, and only introduced the best-match policy in a post-processing step, with the goal of removing rules from the set without adversely affecting accuracy. It also used a different distance measure. Its approach is carried further in the FCLS system (Zhang, 1990), which combines rules with exemplars in an attempt to alleviate the small disjuncts problem. Unlike RISE, FCLS employs different representations for rules and exemplars, and it uses the separate-and-conquer strategy of its AQ ancestors.

RISE addresses the fragmentation problem in rule induction by employing a "conquering without separating" induction strategy. Other approaches to this problem include constructing new attributes (Pagallo & Haussler, 1990) and converting decision trees to decision graphs (Oliveira & Sangiovanni-Vincentelli, 1995; Kohavi & Li, 1995).

Viewed as an instance-based learner, RISE performs context-sensitive feature selection, which can be seen as an extreme form of context-sensitive feature weighting. A number of methods of this type have been proposed in the literature (Aha & Goldstone, 1992; Atkeson, Moore & Schaal, in press). Non-metric, context-sensitive distance measures for IBL are discussed in (Biberman, 1994). Another important aspect of RISE is its policy for combining numeric and symbolic attributes, using SVDM for the former and Euclidean distance for the latter. Alternative approaches have been explored by Ting (1994) and Mohri and Tanaka (1994).

MCS (Brodley, 1995) has the most similar aims to RISE's, but uses an entirely different approach (applying meta-knowledge to detect when one algorithm should be used instead of another), and combines instead decision trees with IBL and linear discriminant functions. Golding and Rosenbloom (1991) designed a system that gainfully combined case-based and rule-based reasoning, but it did not learn, it matched rules exactly, and it used different representations and match procedures for cases and rules, instead of RISE's unified approach. 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 performance 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, 1989b), rules and Bayesian classification (Smyth, Goodman & Higgins, 1990), back-propagation and genetic algorithms (Belew, McInerney & Schraudolph, 1992), instances and prototypes (Scott & Sage, 1992), decision trees and kernel density estimation (Smyth, Gray & Fayyad, 1995), etc.

In form, the most similar system to RISE in the literature is EACH (Salzberg, 1991), which generalizes instances to hyperrectangles, and classifies each test example according to its nearest hyperrectangle. It differs from RISE in many ways: it is applicable only in purely numerical domains, is an incremental algorithm, never drops attributes, uses different heuristics and search strategies, always prefers the most specific hyperrectangle, etc. Recently Wettschereck and Dietterich (1995) carried out a detailed comparison of EACH and k-nearest-neighbor (kNN), and designed an algorithm that combines the two (Wettschereck, 1994), but does not achieve greater accuracy than kNN alone. They found EACH to be less accurate than kNN in most of the domains studied, and the chief cause of this to be EACH's use of overlapping rectangles. Since RISE and EACH use similar representations in the case of numeric attributes, this warrants closer examination.

Two issues are involved in Wettschereck and Dietterich's study, and they should be clearly distinguished. One is whether using one nearest neighbor is preferable to using several. This has been studied for many years in the nearest neighbor literature (e.g., Cover & Hart, 1967). Another issue, of more interest here, is whether or not generalization of instances to hyperrectangles is beneficial. Unfortunately, the study does not decouple the two issues, and it is not possible to determine which of the two factors (or both) is responsible for the observed differences in performance. EACH (and RISE) can be extended to use the k nearest rules for classification; doing so and comparing the resulting approaches with kNN is a worthy topic for future research. Here we will discuss our results on the issue of whether or not to generalize instances to rules, in the context of using the single nearest rule for classification.

In the cross-validation studies reported above, RISE's tie-breaking policy based on Laplace accuracy was compared with one selecting the most specific rule as in EACH, and found to be clearly superior. This can be understood as follows. In regions of overlap, EACH arbitrarily assigns all examples to the class of the most specific hyperrectangle. In contrast, RISE's learning strategy approximates the optimal decision rule of placing the boundary between two classes at the point where the density of examples from one overtakes that of the other (Duda & Hart, 1973). This is because a rule is started from each example, and its generalization halts when it would include more examples of other classes than of the example's one. The use of Laplace accuracy then implies that, given similar-sized samples, each rule prevails in regions where the density of examples of its class is greater. RISE's batch-learning approach also avoids the problems that EACH's incremental learning one was observed to suffer from. As reported above, RISE outperformed two one-nearest-neighbor algorithms (PEBLS and RISE's own IBL component) in a large-scale empirical study. All these facts support the conclusion that generalizing instances to rules can indeed produce substantial improvements in accuracy, if done in an appropriate manner.

8. Conclusions and future research

A major direction for future research is extending the current RISE framework to include analytical as well as empirical learning. This will allow us 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. Other directions for research include: studying in greater detail the behavior of RISE with windowing, and determining its applicability to large databases; extending it to use the k nearest rules to classify a new example; experimenting with methods for generalizing a symbolic antecedent that stop short of deleting it (e.g., through the introduction of internal disjunctions); experimenting with alternative methods for processing missing values; and performing an average-case analysis of the algorithm. Finally, we plan to make available a user-friendly version of RISE.

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. A simplification of the knowledge engineer's task is now possible, since she can use a single algorithm (RISE) instead of an IBL and a rule induction one, with reasonable confidence that accuracy will not be reduced, and may in fact be improved.

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.

Notes

1. This phrase will be used in this article to refer to a family of learning approaches that also includes, or is also known as, exemplar-based, memory-based, case-based, experience-based, kernel-based, nearest-neighbor, local, and lazy learning.
2. The computations in this reference are only for the basic step of the algorithms, which is embedded in loops that may run $O(ea)$ in the worst case, leading to the bound shown.
3. BC: Ljubljana dataset; CH: king-rook-vs-king-pawn dataset; EC: class is 2nd feature, features 1 and 10-13 deleted, example with unknown class deleted; HD: Cleveland dataset, last feature deleted to yield a two-class problem; HO: class is 24th feature, features 3 and 25-28 deleted; LI: 100 examples, seed = 1, 10% noise; PO: pseudo-discretized values reconverted to numeric; SF: 1st feature used as class; SO: small dataset.
4. Clark & Boswell (1991) observed Laplace accuracy to outperform entropy in the context of CN2, but conducted no comparison of Laplace accuracy with uncorrected accuracy.

References

Aha, D. W. (1990). *A study of instance-based learning algorithms for supervised learning tasks: Mathematical, empirical, and psychological evaluations* (Technical Report 90-42). Irvine, CA: University of California at Irvine, Department of Information and Computer Science.

Aha, D. W. (Ed.) (in press). Special issue on lazy learning. *Artificial Intelligence Review*.

Aha, D. W., & Bankert, R. L. (1994). Feature selection for case-based classification of cloud types: An empirical comparison. *Proceedings of the 1994 AAAI Workshop on Case-Based Reasoning* (pp. 106–112). Seattle, WA: AAAI.

Aha, D. W., & Goldstone, R. L. (1992). Concept learning and flexible weighting. *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (pp. 534–539). Bloomington, IN: Lawrence Erlbaum.

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

Atkeson, C. G., Moore, A. W., & Schaal, S. (in press). Locally weighted learning. *Artificial Intelligence Review*.

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.

Biberman, Y. (1994). A context similarity measure. *Proceedings of the Ninth European Conference on Machine Learning* (pp. 49–63). Catania, Italy: Springer-Verlag.

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

Brodley, C. E. (1995). Recursive automatic bias selection for classifier construction. *Machine Learning, 20*, 63–94.

Buntine, W. (1989). Learning classification rules using Bayes. *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 94–98). Ithaca, NY: Morgan Kaufmann.

Cameron-Jones, R. M. (1992). Minimum description length instance-based learning. *Proceedings of the Fifth Australian Joint Conference on Artificial Intelligence* (pp. 368–373). Hobart, Australia: World Scientific.

Catlett, J. (1991). Megainduction: A test flight. *Proceedings of the Eighth International Conference on Machine Learning* (pp. 589–604). Evanston, IL: 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). Porto, Portugal: Springer-Verlag.

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

Cohen, W. W. (1995). Fast effective rule induction. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 115–123). Tahoe City, CA: Morgan Kaufmann.

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. *Proceedings of the Sixth IEEE International Conference on Tools with Artificial Intelligence* (pp. 704–707). New Orleans, LA: IEEE Computer Society Press.

Domingos, P. (1995a). Rule induction and instance-based learning: A unified approach. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1226–1232). Montreal, Canada: Morgan Kaufmann.

Domingos, P. (1995b). *The RISE 2.0 system: A case study in multistrategy learning* (Technical Report 95-2). Irvine, CA: University of California at Irvine, Department of Information and Computer Science.

Domingos, P. (1995c). Two-way induction. *Proceedings of the Seventh IEEE International Conference on Tools with Artificial Intelligence* (pp. 182–189). Herndon, VA: IEEE Computer Society Press.

Domingos, P. (in press). Context-sensitive feature selection for lazy learners. *Artificial Intelligence Review*.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York, NY: 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). Anaheim, CA: AAAI Press.

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

Holte, R. C., Acker, L. E., & Porter, B. W. (1989). Concept learning and the problem of small disjuncts. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 813–818). Detroit, MI: Morgan Kaufmann.

Kelly, J. D., & Davis, L. (1991). A hybrid genetic algorithm for classification. *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence* (pp. 645–650). Sydney, Australia: Morgan Kaufmann.

Kohavi, R., & Li, C. (1995). Oblivious decision trees, graphs, and top-down pruning. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1071–1077). Montreal, Canada: Morgan Kaufmann.

Kolodner, J. (1993). *Case-based reasoning*. 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). Philadelphia, PA: AAAI Press.

Michalski, R. S., & Tecuci, G. (Eds.) (1993). *Proceedings of the Second International Workshop on Multistrategy Learning*. Harpers Ferry, VA: Office of Naval Research/George Mason University.

Michalski, R. S., & Tecuci, G. (Eds.) (1994). *Machine learning: A multistrategy approach*. San Mateo, CA: Morgan Kaufmann.

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

Mohri, T., and Tanaka, H. (1994). An optimal weighting criterion of case indexing for both numeric and symbolic attributes. *Proceedings of the 1994 AAAI Workshop on Case-Based Reasoning* (pp. 123–127), Seattle, WA: AAAI.

Murphy, P. M., & Aha, D. W. (1995). *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). Bled, Yugoslavia: Sigma.

Oliveira, A. L., and Sangiovanni-Vincentelli, A. (1995). Inferring reduced ordered decision graphs of minimum description length. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 421–429). Tahoe City, CA: Morgan Kaufmann.

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. (1987). Generating production rules from decision trees. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence* (pp. 304–307). Milan, Italy: Morgan Kaufmann.

Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning, 5*, 239–266.

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). Amherst, MA: Morgan Kaufmann.

Rao, R. B., Gordon, D., & Spears, W. (1995). For every generalization action, is there really an equal and opposite reaction? Analysis of the conservation law for generalization performance. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 471–479). Tahoe City, CA: Morgan Kaufmann.

Rendell, L. (1986). A general framework for induction and a study of selective induction. *Machine Learning, 1*, 177–226.

Riesbeck, C. K., & Schank, R. C. (1989). *Inside case-based reasoning*. Hillsdale, NJ: Lawrence Erlbaum.

Rivest, R. L. (1987). Learning decision lists. *Machine Learning, 2*, 229–246.

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. (1994a). Cross-validation, stacking, and bi-level stacking: Meta-methods for classification learning. In P. Cheeseman & R. W. Oldford (Eds.), *Selecting models from data: Artificial intelligence and statistics IV*. New York, NY: Springer-Verlag.

Schaffer, C. (1994b). A conservation law for generalization performance. *Proceedings of the Eleventh International Conference on Machine Learning* (pp. 259–265). New Brunswick, NJ: Morgan Kaufmann.

Scott, P. D., & Sage, K. H. (1992). Why generalize? Hybrid representations and instance-based learning. *Proceedings of the Tenth European Conference on Artificial Intelligence* (pp. 484–486). Vienna, Austria: Wiley.

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). Stockholm, Sweden: Springer-Verlag.

Smyth, P., Gray, A., & Fayyad, U. (1995). Retrofitting decision tree classifiers using kernel density estimation. *Proceedings of the Twelfth International Conference on Machine Learning* (pp. 506-514). Tahoe City, CA: Morgan Kaufmann.

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

Ting, K. M. (1994). *Discretization of continuous-valued attributes and instance-based learning* (Technical Report 491). Sydney, Australia: Basser Department of Computer Science, University of Sydney.

Towell, G. G., & Shavlik, J. W. (1994). Knowledge-based artificial neural networks. *Artificial Intelligence, 70*, 119–165.

Townsend-Weber, T., & Kibler, D. (1994). Instance-based prediction of continuous values. *Proceedings of the 1994 AAAI Workshop on Case-Based Reasoning* (pp. 30–35). Seattle, WA: AAAI.

Utgoff, P. E. (1989a). Incremental induction of decision trees. *Machine Learning, 4*, 161–186.

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

Wettschereck, D. (1994). A hybrid nearest-neighbor and nearest-hyperrectangle algorithm. *Proceedings of the Ninth European Conference on Machine Learning* (pp. 323–335). Catania, Italy: Springer-Verlag.

Wettschereck, D., & Dietterich, T. (1995). An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms. *Machine Learning, 19*, 5–27.

Wnek, J., & R. S. Michalski (1994). Hypothesis-driven constructive induction in AQ17-HCI: A method and experiments. *Machine Learning, 14*, 139–168.

Zhang, J. (1990). A method that combines inductive learning with exemplar-based learning. *Proceedings of the Second IEEE International Conference on Tools for Artificial Intelligence* (pp. 31–37). San Jose, CA: IEEE Computer Society Press.

Received June 22, 1994
Accepted October 30, 1995
Final Manuscript December 5, 1995