Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science
http://pic.sagepub.com/

A rule merging technique for handling noise in inductive learning

D T Pham, S Bigot and S S Dimov

Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 2004
218: 1255
DOI: 10.1243/0954406042369017

The online version of this article can be found at:
http://pic.sagepub.com/content/218/10/1255

Published by:
SAGE
http://www.sagepublications.com

On behalf of:
Institution of Mechanical Engineers
Institution of Mechanical Engineers

Additional services and information for *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* can be found at:

Email Alerts: http://pic.sagepub.com/cgi/alerts

Subscriptions: http://pic.sagepub.com/subscriptions

Reprints: http://www.sagepub.com/journalsReprints.nav

Permissions: http://www.sagepub.com/journalsPermissions.nav

>> Version of Record - Oct 1, 2004

What is This?

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

A rule merging technique for handling noise in inductive learning

D T Pham*, **S Bigot** and **S S Dimov**
Intelligent Systems, Laboratory, School of Engineering, Cardiff University, Cardiff, Wales, UK

Abstract: Inductive learning algorithms are used for extracting IF–THEN rules from examples. The main weakness of most existing algorithms is their poor ability to handle data containing noise. This problem is even more severe when inductive learning techniques are applied to real engineering data. The paper presents a new pruning technique that improves significantly the performance of the RULES family of inductive learning algorithms. The technique is designed for RULES-5, the latest algorithm in the family, but could readily be applied to rule sets created by other algorithms.

Keywords: machine learning, rule induction, pruning, noise handling

NOTATION

A^i	ith attribute in an example
Best_rule	best rule created by RULES-5
BPP	basic post-pruning
C_E	class value in example E
CE	example closest to *SE* not belonging to the target class
$Cond^i_R$	condition in rule R for the ith attribute
Final Rset	rule set resulting from the pruning
IPP	incremental post-pruning
IREP	incremental reduced error pruning
m	number of attributes in an example
New_Rset	temporary rule set
NL	noise level
REP	reduced error pruning
Rset	rule set being pruned
R2M	rule to be merged
SE	seed example
Th	noise threshold
TC_R	target class value in rule R
V^i_E	value of the ith attribute in example E
$V^i_{\max}$	maximum known value of the ith continuous attribute
$V^i_{\min}$	minimum known value of the ith continuous attribute
$V^i_{\max R}$	lower bound employed in rule R to form a condition on the ith continuous attribute
$V^i_{\min R}$	lower bound employed in rule R to form a condition on the ith continuous attribute
V^i_R	discrete value employed in rule R to form a condition on the ith discrete attribute

The MS was received on 5 February 2004 and was accepted after revision for publication on 18 June 2004.
* *Corresponding author: Intelligent Systems Laboratory, Manufacturing Engineering Centre, School of Engineering, Cardiff University, PO Box 925, Cardiff CF24 0YF, Wales, UK. E-mail: PhamDT@cf.ac.uk.*

1 INTRODUCTION

The last decade has seen a steep increase in data storage capabilities. Many organizations now hold large amounts of historical data that contain useful but non-apparent or hidden knowledge. This has led to the emergence of a new research field, knowledge discovery in databases (KDD), that aims to develop techniques for 'mining' useful knowledge from data. A typical KDD process comprises the selection of informative data from the database, the pre-processing of that data (for example, to correct errors and deal with missing details), the application of data mining techniques to obtain specific patterns and the interpretation of those patterns to extract the targeted knowledge. The key part of this process is the data mining operation. Data mining, defined by Mitchell as 'using historical data to discover regularities and improve future decisions' [**1**], involves applying specific algorithms for pattern extraction. Many types of data mining algorithms exist, depending on the kind of knowledge targeted. These algorithms are based on a range of tools, from statistics to machine learning algorithms. They include neural network training algorithms, instance-based algorithms, genetic algorithms and algorithms for inductive learning and association rules learning. These algorithms allow the creation of different types of model that describe the

Downloaded from pic.sagepub.com at Cardiff University on March 2012

patterns found in the data. Data mining has many potential applications in mechanical and manufacturing engineering, enabling the development of knowledge-based systems for a variety of selection and classification tasks [2].

This paper concentrates on one type of data mining algorithms, namely inductive learning algorithms. An important characteristic of inductive learning algorithms is that their model structure is readily understood by people. Most inductive learning methods employ the 'concept learning' approach [3] and can be categorized [4] into divide-and-conquer methods and covering methods.

Divide-and-conquer algorithms, such as ID3 [5], C4.5 [6] and the commercially available C5 algorithm [7], construct sets of hypotheses in the form of decision trees. In contrast, covering methods, such as CN2 [8, 9], RIPPER [10], AQ [11, 12] and its most recent version AQ19 [13], represent classification knowledge in the form of a set of rules to describe each class. More information about covering methods can be found in references [14] and [15].

An important issue for all inductive learning algorithms is how well they handle noisy data. Most algorithms attempt to form rule sets that correctly classify all examples in the training set. In the presence of noisy data, this leads to the generation of overspecialized rules, where algorithms overfit the training data to avoid generating inconsistent rules. This is a common problem when inductive learning methods are applied to data resulting from engineering experiments. In such data sets there are many kinds of noise, for instance measurement, data entry and data transfer errors.

A solution to this problem is to apply pruning methods to make the rules more general. This is usually achieved by tolerating some inconsistency in the generated rule sets. There are two main types of pruning method [16]:

1. *Post-pruning methods* (*rule truncation*). These methods deal with the noise after the learning process is completed, by post-processing the generated fully consistent rule sets. The pruning is carried out by examining each rule and discarding conditions that are created due to the presence of noisy examples. Reduced error pruning (REP) [17] and grow [18] are two typical post-pruning algorithms.
2. *Pre-pruning methods* (*stopping criterion*). These methods deal with the noise during the learning process by employing heuristics to terminate the rule specialization process. All rules are generated in a single pass and some of them may not be consistent.

Post-pruning methods generally yield more accurate rule sets than obtainable with pre-pruning methods. Unfortunately, post-pruning is also computationally expensive. This is because, unlike pre-pruning methods, computational effort is wasted to create overspecialized rules that later have to be pruned. Attempts have been made to combine these two techniques by initially applying pre-pruning techniques to reduce the overspecialization of the rule sets and then post-processing them to complete the process. An example of such an implementation is the top-down pruning algorithm [16]. This hybrid approach offers a good balance between the processing speed of pre-pruning techniques and the accuracy of rule sets obtained by post-processing.

A survey of the most well-known pruning techniques has been carried out by Breslow and Aha [19]. Most pruning techniques were originally developed for decision-tree-based algorithms, because this representation of the classification knowledge facilitates the process. In this paper, the use of pruning methods with covering algorithms is discussed. The aim of the research is to develop an appropriate pruning technique that could be implemented in RULES-5 [20], the latest member of the RULES family of covering algorithms [21, 22].

RULES-5 is an induction algorithm with a performance exceeding that of some of the best available algorithms in problems where the data are relatively free from noise. When noise is present, RULES-5 can generate large numbers of overspecialized rules. An effective pruning technique will enable RULES-5 to handle noisy data and generate compact and accurate rule sets.

The remainder of the paper will summarize the rule formation procedure of RULES-5, review existing pruning methods, describe the proposed pruning technique and two modes of implementing it, give a step-by-step example of its application to RULES-5 and present the results of tests on benchmark data sets to demonstrate the improvements achieved.

2 RULES-5

RULES-5 employs simple and efficient techniques for handling continuous attributes and extracting IF–THEN rules from examples. Data are presented to RULES-5 in the form of a collection of objects, each belonging to one of a number of given classes. These objects, together with their associated classes, constitute a set of training examples from which the algorithm induces a model. Each example E is described by its class value C_E and by a vector of m attributes $(A^1, \ldots, A^i, \ldots, A^m)$. Each attribute value V_E^i is either discrete or continuous. In the case of a continuous attribute, $V_{\min}^i \leqslant V_E^i \leqslant V_{\max}^i$, where $V_{\min}^i$ is the minimum known value for the ith attribute and $V_{\max}^i$ its maximum known value. An example E is therefore

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

RULES-5 Rule-Forming Procedure

Take one example uncovered by the rule set formed so far (*SE*)
Initialise *PRSET* (empty list)
Form an initial rule with no conditions to classify *SE*
Store this rule in *PRSET* and copy it into *best_rule*

WHILE *best_rule* is not consistent **DO**
  Initialise *T_PRSET* (empty list)
  FOR each rule in *PRSET* **DO**
    rule_to specialise = the rule taken from *PRSET*
    CE = an example misclassified by *rule_to specialise* and the closest to *SE*
    FOR $i = 1$ to $i = m$ **DO**
      IF $V^i_{SE} \neq V^i_{CE}$ **THEN**
        new_rule = *rule_to_specialise*
        IF continuous attribute **THEN**
          IF $V^i_{SE} < V^i_{CE}$ **THEN**
            Append the condition $[A^i < V^i_{CE}]$ to *new_rule*
          ELSE
            Append the condition $[A^i > V^i_{CE}]$ to *new_rule*
        ELSE
            Append the condition $[A^i = V^i_{SE}]$ to *new_rule*
        IF *new_rule* is consistent
        AND *new_rule* covers more uncovered examples than *best_rule* **THEN**
            Replace *best_rule* with *new_rule*
        IF *new_rule* is not consistent **THEN**
            IF number of rules in *T_PRSET* < *PRSET*_size **THEN**
              Store *new_rule* into *T_PRSET*
            ELSE
            IF the *new_rule* H measure is higher than the H measure of any rule in *T_PRSET* **THEN**
              Replace the rule with the lowest H measure in *T_PRSET* with *new_rule*
      END FOR
    END FOR
  copy *T_PRSET* into *PRSET*
END WHILE

IF *best_rule* contains continuous attribute conditions **THEN**
  Constrain their coverage to training examples

Fig. 1 RULES-5 rule-forming procedure

formally defined as follows:

$$E = (A^1 = V^1_E, \ldots, A^i = V^i_E, \ldots, A^m = V^m_E, \text{Class} = C_E)$$

RULES-5 forms a new rule by starting from an example not covered by previously created rules, the seed example. The algorithm employs a specialization process that searches for consistent rules that are as general as possible. The result is a rule set that correctly classifies all or most of the training examples.

A rule set is a list of IF–THEN rules. Each rule R is described by a conjunction of conditions on each attribute ($Cond^i_R$) and by a target class value (TC_R). A rule R can be formally defined as $Cond^1_R \wedge \cdots \wedge Cond^i_R \wedge \cdots \wedge Cond^m_R \rightarrow TC_R$. If $Cond^i_R$ exists, it could be an attribute–value pair ($A^i = V^i_R$) or a range of values ($V^i_{\min R} \leqslant A^i \leqslant V^i_{\max R}$) for discrete and continuous attributes respectively, where V^i_R is a discrete value and $V^i_{\min R}$ and $V^i_{\max R}$ are real numbers in the ith continuous attribute range ($V^i_{\min}, V^i_{\max}$).

The complete rule-forming procedure of RULES-5 is given in Fig. 1. Thus, RULES-5 produces rule sets that do not contain any inconsistent rules and cover fully the training examples. Another specific aspect of the RULES-5 search mechanism is the use of information about the distribution of examples in order to reduce the dependence of the concept formation process on the heuristic measures employed. Consequently, less variability is achieved in the performance of the algorithm with respect to different data sets. This feature, along with the knowledge representation structure of RULES-5, has to be taken into account when designing a pruning technique for the algorithm.

3 EXISTING PRUNING METHODS

As already mentioned, there are two main types of pruning techniques: 'post-pruning' and 'pre-pruning'. Most of these techniques have been designed for decision tree structures, especially the pre-pruning

Downloaded from pic.sagepub.com at Cardiff University on March 1, 2012

techniques based on the use of a stopping criterion. In order to adopt these methods for covering algorithms, the algorithms themselves should be modified. Such an approach is applied in the CN2 algorithm [**10**, **11**], which is essentially a covering algorithm (AQ) that has been adapted to allow a decision tree pre-pruning technique to be employed.

Research carried out by Frank [**23**] shows that pre-pruning techniques are generally faster but less accurate than post-pruning techniques. Frank also came to the conclusion [**23**] that pre-pruning techniques, and especially those based on a stopping criterion, tend to terminate the specialization process before all branches are optimized. In general, post-pruning methods have the potential to achieve a higher level of rule set refinement. Unfortunately, most post-pruning techniques applied to decision trees cannot be used directly for rule set processing, because their pruning strategies are specifically designed for the node/leaf structures of decision trees. Thus, few existing post-pruning techniques have been adopted for covering algorithms.

Some researchers even claim that existing pruning techniques are not applicable to rule sets obtained using conventional covering algorithms. For instance, Fürnkranz [**16**] states that the 'pruning of branches in a decision tree will never affect the neighbouring branches, whereas pruning of conditions of a rule will affect all subsequent rules'. This is due to the fact that most covering methods remove examples from the training sets when they are covered by a newly formed rule. As a result, the heuristics employed to guide the specialization process take into account only the remaining examples. Therefore, the newly formed rules depend on the rules formed so far. This leads to the creation of rules that are dependent on one another, and pruning one of them affects the performance of the whole rule set.

The RULES family differs from other covering algorithms. The examples covered by previously formed rules are only marked in order to avoid the creation of unnecessary rules. These examples are used to assess the accuracy and generality of each newly formed rule. As a result, all rules are independent and each can be pruned without affecting the rest of the rule set. It should be noted, however, that the reasons for other algorithms removing examples from the training set is to direct the search process towards forming rules that cover a maximum number of uncovered examples. This leads to more compact rule sets and minimizes the overlapping between rules. For the RULES family, a specific heuristic has been developed [**24**] in order to provide guidance to the search process. The heuristic takes into account already covered examples without the need to remove them from the training set. This enables the RULES family of algorithms to create independent rules as well as more compact rule sets.

Two of the most popular post-pruning techniques are reduced error pruning (REP) [**25**] and incremental reduced error pruning (IREP) [**26**]. REP was originally applied only to decision trees but after modification it was used for pruning rule sets. The modified version of this technique has been implemented in C4.5 [**6**], where the decision tree is converted into a set of rules that are then pruned using REP. Algorithms using the REP technique split the training set into two subsets: a growing set (usually two-thirds of the training set) and a pruning set. The algorithms use the growing set to form the initial rule sets and then the effect of the pruning is evaluated on the second set of examples. These algorithms process the rule sets step by step until no further improvement can be achieved without sacrificing accuracy. This post-pruning technique has been improved in IREP, where instead of creating the entire rule set and then pruning it, the algo-rithm prunes the individual rules immediately after their creation. In this way the generation of overspecialized rules that need to be pruned later on can be avoided.

The main deficiency of both techniques is the necessity to divide the training set into two subsets and the need for the pruning subset to contain 'at least one example of each disjunctive clause' [**17**]. These requirements might be difficult to meet in cases where only a small number of examples are available. Furthermore, useful information that is contained in the pruning subset of examples would not be utilized during the rule-forming process. This could lead to less generic rule sets that do not cover fully the example space. As a consequence, the resulting rule sets would be highly dependent on the adopted approach for splitting the training data. In addition, these post-pruning techniques could lead to overfitting of the pruning sets [**27**]. Alternative methods exist that do not require the training set to be split. With these methods, the removal of each rule or condition is evaluated by applying heuristic measures. For instance, a number of methods have been developed recently based on the minimum description length principle in order to evaluate rule sets resulting from the pruning process [**28**–**30**].

An important weakness of these methods is that it is computationally expensive to identify rules or conditions to be removed. In addition, they only rely on heuristic measures to evaluate whether an acceptable level of pruning has been reached. These measures stop the pruning process when a specific criterion is satisfied. They are employed in an arbitrary manner. Their effect on the learning process is not fully understood and their performance often varies depending on the application domain. This makes the performance of the pruning process very dependent on the quality of the selected measures.

The pruning technique proposed in this paper does not require the data set to be split and allows the user to control the level of pruning. This considerably reduces the dependence of process performance on the heuristic measures used.

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

4 DESCRIPTION OF THE PROPOSED PRUNING TECHNIQUE

To understand the proposed technique, a simple example will be used to demonstrate the effect of noisy data on the rule-forming process. Figure 2 provides a graphical representation of a training set containing noisy data. The rules generated by RULES-5 are given in Fig. 3. As can be seen in this figure, the noisy data prevents the creation of a more general rule for class + and forces RULES-5 to create three more specific rules instead.

A key component of the proposed pruning technique is an operation that merges some rules in order to create more general ones. This is done by making the assumption that these rules would have been formed by RULES-5 if noisy examples were not present.

This section describes the procedure applied to merge two rules. The rules resulting from these mergers might not be consistent according to the adopted consistency measure [**24**], but this inconsistency could be acceptable up to a certain limit.

This limit is represented by the noise threshold *Th*, which is defined as the required level of consistency of a rule, and is given by the user, based on his or her domain knowledge regarding the amount of noise in the data. The user specifies this information in the form of a parameter called noise level. The noise level is defined as the percentage of noise (examples with incorrect values) expected in the data set. For example, if the noise in the training data is considered to be 10 per cent then $NL = 0.10$ and, using *NL*, *Th* can be computed as $Th = 1 - NL = 0.9$.

The main objective of the proposed pruning procedure is to merge specific rules in order to create new, more-general rules with a consistency level equal to or higher than a specified *Th*. The most general rules are kept, to form a new rule set, and at the same time the rules that classify only examples already covered by those more general rules are removed. This leads to the creation of more compact rule sets that could handle noisy examples with greater efficiency.

The proposed pruning technique creates a new rule

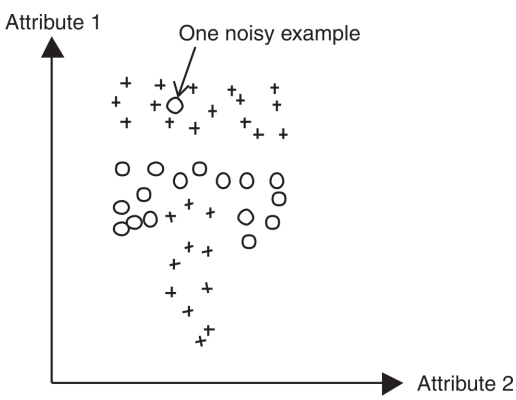

Fig. 2 Training data containing noise

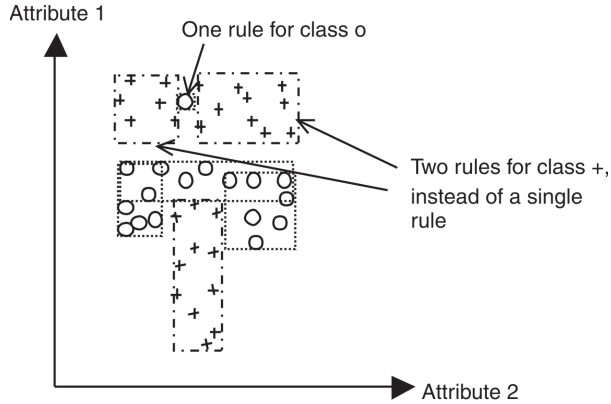

Fig. 3 Rules generated by RULES-5

by merging two existing rules *R*1 and *R*2 covering examples from the same class. This newly generated rule (*new_rule*) contains new conditions formed by combining the conditions of the two existing rules. The merging of conditions between *R*1 and *R*2 is performed by carrying out the following operations:

1. *Continuous attribute*. If a condition exists for a particular attribute in both rules ($V^i_{\min R1} \leqslant A^i \leqslant V^i_{\max R1}$ and $V^i_{\min R2} \leqslant A^i \leqslant V^i_{\max R2}$), a new condition is created, by forming a new attribute range that includes the ranges of both rules. Otherwise, no condition will be formed for this attribute in the newly formed rule. The new condition in the new rule has the following form:

$$V^i_{\min new\_rule} \leqslant A^i \leqslant V^i_{\max new\_rule}$$

with $V^i_{\min new\_rule} = V^i_{\min R1}$ if $V^i_{\min R1} \leqslant V^i_{\min R2}$ and $V^i_{\min new\_rule} = V^i_{\min R2}$ otherwise. Also, $V^i_{\max new\_rule} = V^i_{\max R2}$ if $V^i_{\max R1} \leqslant V^i_{\max R2}$ and $V^i_{\max new\_rule} = V^i_{\max R1}$ otherwise.

2. *Discrete attribute*. If a condition exists for a particular attribute in both rules and if the attribute values in the conditions of both rules are the same ($V^i_{R1} = V^i_{R2}$), then the same attribute value is used to form a condition in the newly formed rule ($A^i = V^i_{R1}$). Otherwise, no condition will be formed for this attribute in the newly formed rule.

An example of merging two rules is given below:

Rule 1: IF A1 = yes And A2 = blue And A3 = no And 7 < A4 < 12 Then Class 1.
Rule 2: IF A1 = yes And A2 = green And 2 < A4 < 6 Then Class 1.
Merged rule: IF A1 = yes And 2 < A4 < 12 Then Class 1.

The new rule is more general and classifies all examples covered by the two initial rules. It could also classify other examples, including potentially noisy ones. Therefore, the accuracy of the resulting rules from this operation must be checked against a predefined *Th* to

Downloaded from pic.sagepub.com at Cardiff University on March 1, 2012

decide if the inconsistency of the new rule is within acceptable limits.

5 RULE PRUNING PROCESS

This section describes two ways of applying the proposed pruning technique. The first approach, called basic post-pruning (BPP), is similar to REP and is applied after a rule set is already formed. This method can be computationally expensive and therefore an incremental method, called incremental reduced error pruning (IREP), was proposed [**16**]. With this method, the pruning is carried out incrementally during the rule set formation whenever a new rule is created. By applying this method, more general rules are formed as early as possible during the learning process, and fewer computations are required to form a rule set. Based on this approach a second pruning method is developed, called incremental post-pruning (IPP). Both methods employ the same technique for merging rules.

5.1 Basic post-pruning

This technique is applied to the entire rule set (*RSet*). The algorithm starts by taking one rule at a time, from *RSet*, called the rule-to-be-merged (*R2M*). This rule is then merged with each of the other rules for the same class within *RSet*. If the consistency measure of the best resulting rule (the one with the highest consistency) from these mergers is equal to or higher than *Th*, then it is added to *RSet* and the rules used for its formation are removed from *RSet*. Otherwise, if the consistency of the best rule is lower than *Th*, which means that the generality of *R2M* cannot be improved further without sacrificing its accuracy, the algorithm stores *R2M* in the new rule set (*New_RSet*) and removes it from *RSet*. If there are still rules within *RSet* that are not processed, the algorithm takes one of them as *R2M* and repeats the procedure.

This iterative process leads to the creation of *New_RSet*, which is more general than *RSet*. However, at the same time it could contain rules that classify only examples already covered by other more general rules in *RSet*. Therefore, *New_RSet* has to be further processed to create a new set called the final rule set (*Final_RSet*). This is carried out by selecting the most general rule within *New_RSet*, the rule that classifies the largest number of examples not covered by *Final_RSet* formed so far, and transferring it into *Final_RSet*. The procedure is repeated until there are no examples uncovered by *Final_RSet*. The complete BPP procedure is presented in Fig. 4.

The Basic Post-Pruning Procedure

Take a rule set to be pruned (*RSet*)

STEP 1
- Initialise *New-RSet* (empty list)

WHILE there is a rule in *RSet* **DO**
  STEP 2
  - *R2M* = the first rule in *RSet*
  - Merge *R2M* with each rule for the same class in *RSet*
  - *new_rule* = the rule with the highest consistency measure resulting from the mergers
  STEP 3
  - **IF** *new_rule* consistency measure $\geqslant$ *Th* **THEN**
    - Remove all rules used for its formation from *RSet*
    - Add *new_rule* into *RSet*
  - **ELSE**
    - Add *R2M* into *New-RSet*
    - Remove *R2M* from *RSet*

END WHILE

STEP 4
- Initialise *Final-RSet* (empty list)

WHILE there are examples not classified by *Final-RSet* **DO**
  STEP 5
  - *new_rule* = the rule in *New-RSet* covering the largest number of examples not covered by *Final-RSet* formed so far
  - Add *new_rule* into *Final-RSet*

END WHILE

Where: *RSet* is the rule set being pruned; *New-RSet* is a temporary rule set; *R2M* is the rule to be merged; *new_rule* is a temporary rule; *Final-RSet* is the resulting rule set.

Fig. 4 The basic post-pruning (BPP) procedure

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

The Incremental Post-Pruning Procedure

WHILE there is an uncovered example **DO**
best_rule = a rule formed by the Dyna rule forming procedure
IF *Th* < 1 **THEN**
- *R2M* = *Best_Rule*
STEP 1
- Merge *R2M* with each rule for the same class in *RSet*
- *new_rule* = the rule with the highest consistency measure resulting from the mergers
- **IF** *new_rule* consistency measure ⩾ *Th* **THEN**
  - Remove all rules used for its formation from *RSet*
  - Add *new_rule* into *RSet*
  - *R2M* = *new_rule*
  - go back to *STEP 1*

END WHILE

- Initialise *Final-RSet* (empty list)
WHILE there are examples uncovered by the *Final-RSet* **DO**
- *new_rule* = the rule in *RSet* covering the largest number of examples still not covered by *Final-RSet* formed so far
- Add *new_rule* into *Final-RSet*

END WHILE

Where: *RSet* is the rule set being pruned; *Best_Rule* is the rule created by Dyna; *R2M* is the rule to be merged; *new_rule* is a temporary rule; *Final-RSet* is the resulting rule set.

Fig. 5 The incremental post-pruning (IPP) procedure

5.2 Incremental post-pruning

In contrast to BPP, IPP is applied in parallel to the rule-forming process. Whenever a new rule is formed, the IPP method merges this new rule (*R2M*) with each rule for the same class stored in *RSet*. If the consistency measure of the best resulting rule (*Best_Rule*) is equal to or higher than *Th*, then the rules used for its formation are removed from *RSet*, *Best_Rule* is added to *RSet* and a new iteration starts with this rule as *R2M*. Otherwise, if the consistency of *Best_Rule* is lower than *Th*, the algorithm stores *R2M* into *RSet* and continues the rule-forming process.

As was the case with BPP, at the end of the rule set forming process, *RSet* could contain rules that classify only examples already covered by more general rules in *RSet*. Therefore, *RSet* is processed further in the same way as *New_RSet* in BPP. The complete procedure is presented in Fig. 5.

6 ILLUSTRATIVE PROBLEM

To illustrate the pruning process carried out by the algorithms described in the previous paragraph, the BPP procedure is applied to the rule set (*RSet*) shown in Fig. 6. The application of the IPP procedure on the same *RSet* is not described because the merging process is identical to BPP. The noise level is set to 10 per cent ($Th = 0.9$) for this illustrative problem.

Step 1. A new empty rule set (*New_RSet*) is initialized *New_RSet* = { }. *RSet* contains seven rules:

$$RSet = \{\text{Rule 1, Rule 2, Rule 3, Rule 4, Rule 5, Rule 6, Rule 7}\} \text{(Fig. 6)}$$

Step 2. The first *R2M* is Rule 1 covering examples belonging to class +. *R2M* is merged with other rules for class +. The results of merging *R2M* with Rule 3 (Rule 1-3) and Rule 2 (Rule 1-2) are shown in Figs 7 and 8 respectively. The best rule with the highest consistency measure resulting from this merger is Rule 1-2; *new_rule* = Rule 1-2, consistency = 16/17 = 0.94.

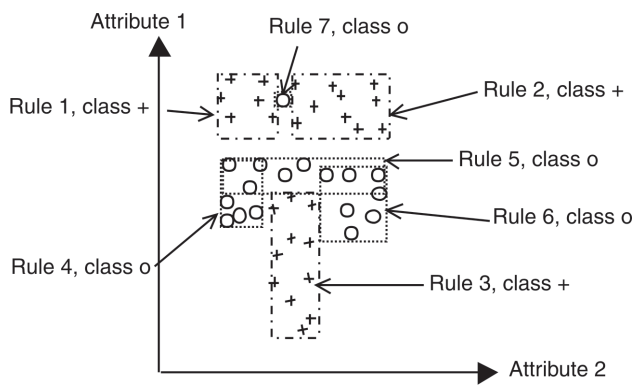

Fig. 6 Before pruning—seven rules

Downloaded from pic.sagepub.com at Cardiff University on March 9, 2012

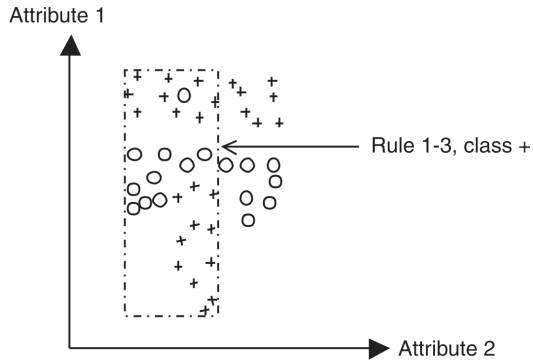

Fig. 7 Rule 1 merged with Rule 3, consistency = 0.67

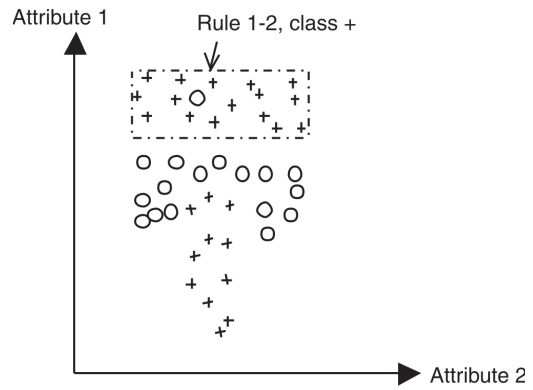

Fig. 8 Rule 1 merged with Rule 2, consistency = 0.94

Step 3. The consistency of *new_rule* is higher than *Th*. Therefore Rule 1 and Rule 2 are removed from *RSet* and *new_rule* is added to *RSet*:

$$RSet = \{\text{Rule 1-2, Rule 3, Rule 4, Rule 5, Rule 6, Rule 7}\}$$

There are still rules in *RSet* and the algorithm goes back to Step 2:

Step 2. *R2M* = Rule 1-2 can be merged only with Rule 3. The result of this merger (Rule 1-2-3) is shown in Fig. 9; *new_rule* = Rule 1-2-3, consistency = 27/44 = 0.61.

Step 3. The consistency of *new_rule* is lower than *Th* and therefore *R2M* is stored into *New-RSet* and at the same time Rule 1-2 is removed from *RSet*:

$$RSet = \{\text{Rule 3, Rule 4, Rule 5, Rule 6, Rule 7}\}$$
$$New\text{-}RSet = \{\text{Rule 1-2}\}$$

There are still rules in *RSet* and the algorithm goes back to Step 2.

Steps 2 and 3. By repeating these two steps until *RSet* is empty, *New-RSet* is formed (Fig. 10):

$$New\text{-}RSet = \{\text{Rule 1-2, Rule 3, Rule 4, Rule 5, Rule 6, Rule 7}\}$$

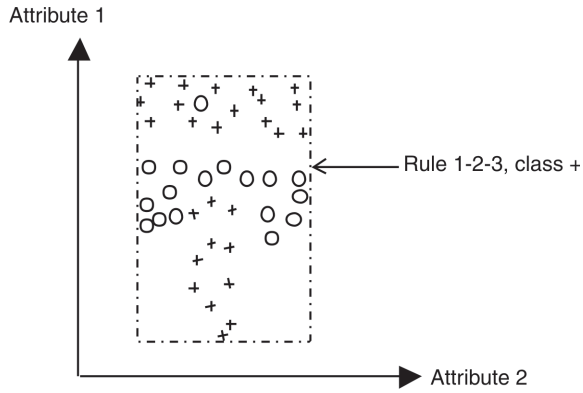

Fig. 9 Rule 1-2 merged with Rule 3, consistency = 0.61

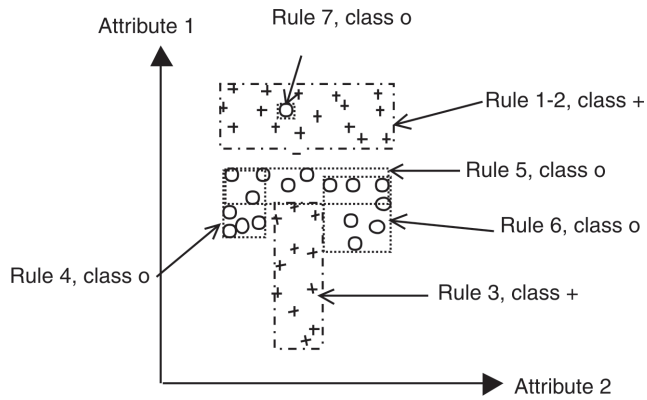

Fig. 10 *New-RSet*—six rules

Step 4. *Final-RSet* is initialized. No examples are covered by *Final-RSet*; *Final-RSet* = {}.

Step 5. The rule covering the highest number of uncovered examples is selected and stored in *Final-RSet*:

$$Final\text{-}RSet = \{\text{Rule 1-2}\}$$

There are still uncovered examples and therefore the procedure returns to Step 5. This is repeated until all examples are covered by *Final-RSet*. Rule 6 classifies the last set of examples not covered by *Final-RSet* so far and therefore Rule 7 is not added to the rule set.

At the end of the process *Final-RSet* includes the following rules (Fig. 11):

$$Final\text{-}RSet = \{\text{Rule 1-2, Rule 3, Rule 4, Rule 5, Rule 6}\}$$

If a higher *NL* is specified (e.g. 20 per cent), this will lead to a different rule set. The pruning result with *Th* = 0.8 (corresponding to the specified *NL* value of 20 per cent) is shown in Fig. 12. In this case the number of rules required to cover the whole data set is reduced to only three.

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

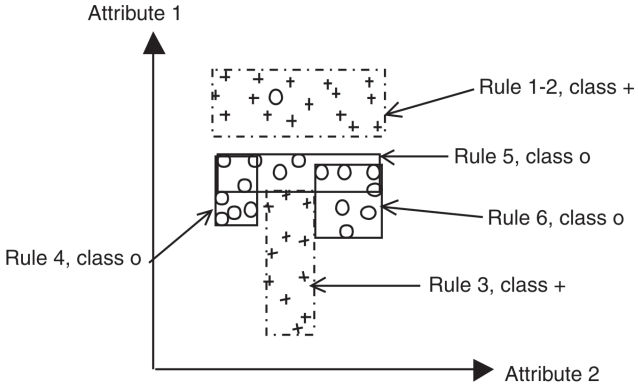

Fig. 11 *Final-RSet*, with specified $NL = 10$ per cent ($Th = 0.9$)

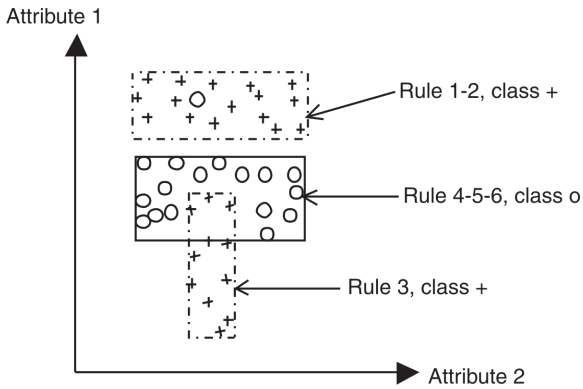

Fig. 12 *Final-RSet*, with specified $NL = 20$ per cent ($Th = 0.8$)

7 TESTS AND ANALYSIS OF RESULTS

A question arises as to how to compare the performance of the two new rule pruning procedures with those obtained by applying other existing methods. The main problem is that the performance of BPP or IPP depends on the noise level specified by the user, while in REP or IREP the level of pruning is set automatically; hence, it is not a straightforward task to carry out this comparison. Also, even though the main principles of these techniques are described in the literature, some issues remain open, such as the way the training set is split into growing and pruning subsets.

Because of the above problems, it was decided to compare the performance of RULES-5 enhanced with BPP or IPP against the well-known divide-and-conquer inductive learning algorithm C5 [**7**]. The two new rule pruning procedures were implemented in RULES-5. Tests were carried out on 15 data sets [**31**] commonly used to benchmark inductive learning algorithms (*balance_scale*, *breast_cancer*, *wdbc*, *wpbc*, *car*, *credit screening*, *cylinder-band*, *dermatology*, *diabetes*, *ecoli*, *glass*, *haberman*, *iris*, *liver* and *tic-tac-toe*).

Initially, the performance of the two new pruning techniques are compared. Then, the rule sets generated by RULES-5 and pruned with the best of these two techniques are evaluated against the rule sets obtained using only RULES-5 or C5.

As mentioned previously, one of the main features of the two new pruning techniques is that the noise level has to be specified by the user, based on his or her domain knowledge. Unfortunately, such expert knowledge is not available for the benchmarking data employed. Three arbitrary noise levels of 0.1, 0.2 and 0.3 are therefore used in this study.

7.1 Comparisons between BPP and IPP

The tests (Table 1) show that the results obtained applying IPP and BPP depend on the training data but on average are similar in terms of the number of formed rules and their accuracy. IPP creates on average approximately the same number of rules as BPP with a similar test accuracy. Because IPP is computationally cheaper, this technique was implemented in RULES-5. However, BPP could be applied as a post-processing technique for rule sets created by other algorithms such as RULES-3 Plus, for which it also gave good results (Table 2).

7.2 Comparison with RULES-5

The performance of RULES-5 with IPP is compared against that of RULES-5 without any pruning. The test results presented in Table 3 show a significant reduction in the number of rules generated when pruning was performed.

The use of $NL = 0.3$ results in performance improvements with a majority of the data sets:

1. RULES-5 + IPP gives higher test accuracies than RULES-5 for 3 of the 15 data sets, with a large decrease in the number of rules.
2. RULES-5 + IPP gives the same test accuracies as RULES-5 for 6 of the 15 data sets, with a significant decrease in the number of rules.
3. RULES-5 + IPP gives lower test accuracies than RULES-5 for 6 of the 15 data sets (*balance_scale*, *breast_cancer*, *wdbc*, *car*, *credit screening* and *ecoli*).

Thus, with $NL = 0.3$, RULES-5 + IPP outperforms RULES-5 for 9 of the 15 data sets. The results obtained with the six remaining data sets show the effect of overpruning. The number of rules is significantly reduced, but in most cases this also results in a decrease

Downloaded from pic.sagepub.com at Cardiff University on March 1, 2012

Table 1 Comparison between BPP and IPP

		RULES-5 with BPP			RULES-5 with IPP		
Data set name	*NL*	Rules	Training %	Test %	Rules	Training %	Test %
balance_scale	0.10	30	90.60	83.07	24	88.99	84.13
	0.20	14	83.72	82.54	16	85.32	78.84
	0.30	6	76.61	71.96	7	76.15	74.60
breast_cancer	0.10	8	93.85	89.10	4	92.62	93.36
	0.20	6	89.96	85.78	6	94.26	92.89
	0.30	5	90.57	87.20	3	90.78	88.15
wdbc	0.10	6	91.18	91.86	8	93.70	94.19
	0.20	5	90.93	91.28	4	92.95	94.19
	0.30	5	88.67	91.86	6	92.95	94.19
wpbc	0.10	23	97.81	77.05	23	97.08	72.13
	0.20	9	89.05	77.05	10	89.78	68.85
	0.30	1	76.64	75.41	1	76.64	75.41
car	0.10	206	100.00	100.00	206	100.00	100.00
	0.20	150	90.45	91.59	150	90.45	91.59
	0.30	1	70.02	59.00	1	70.02	59.00
credit screening	0.10	21	90.04	86.54	19	89.83	85.10
	0.20	8	86.51	85.10	4	86.10	84.62
	0.30	4	85.89	84.62	2	85.89	84.62
cylinder-band	0.10	224	100.00	66.87	224	100.00	66.87
	0.20	224	100.00	66.87	224	100.00	66.87
	0.30	224	100.00	66.87	224	100.00	66.87
dermatology	0.10	42	100.00	94.69	42	100.00	94.69
	0.20	42	100.00	94.69	42	100.00	94.69
	0.30	42	100.00	94.69	42	100.00	94.69
diabetes	0.10	59	93.28	73.71	61	94.59	74.57
	0.20	21	81.90	77.59	24	85.45	74.14
	0.30	4	75.19	74.57	7	79.48	76.29
ecoli	0.10	25	95.24	82.86	23	94.37	82.86
	0.20	14	90.48	81.90	17	90.48	77.14
	0.30	12	85.28	81.90	16	89.18	76.19
glass	0.10	7	100.00	94.03	7	100.00	94.03
	0.20	7	100.00	94.03	7	100.00	94.03
	0.30	7	100.00	94.03	7	100.00	94.03
haberman	0.10	47	95.77	69.89	48	95.77	68.82
	0.20	11	81.22	76.34	11	80.75	77.42
	0.30	1	73.71	73.12	1	73.71	73.12
iris	0.10	5	100.00	95.83	5	100.00	95.83
	0.20	5	100.00	95.83	5	100.00	95.83
	0.30	6	100.00	93.75	5	100.00	95.83
liver	0.10	47	99.03	61.59	49	100.00	57.97
	0.20	24	89.37	61.59	22	86.96	68.12
	0.30	13	71.01	62.32	17	85.51	63.04
tic-tac-toe	0.10	26	100.00	100.00	26	100.00	100.00
	0.20	26	100.00	100.00	26	100.00	100.00
	0.30	12.00	96.57	95.49	11	97.61	100.00

in test accuracy. This is due to the use of too high a level of pruning. It is assumed that, with knowledge about the data, a more appropriate level of pruning can be selected. For instance, with $NL = 0.1$:

1. RULES-5 + IPP gives higher test accuracies than RULES-5 for two of the six data sets (*car* and *ecoli*), with a large reduction in the number of rules.
2. RULES-5 + IPP gives lower test accuracies (2 per cent reduction in average) than RULES-5 for four of the six data sets (*balance_scale*, *breast_cancer*, *wdbc* and *credit screening*).

For those four data sets, a lower noise level could be selected to try to avoid reducing the test accuracy. However, such a reduction can be considered acceptable for many classification applications because the resulting rule sets are much more compact and easier for human experts to understand/verify. For instance, for the *credit screening* data set, the number of rules obtained is 19 instead of 117, with a reduction in test accuracy of only 1.92 per cent (from 87.02 to 85.10 per cent).

Overall, the tests showed that RULES-5 benefits significantly from the introduction of IPP.

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

Table 2 Comparison between RULES-3 Plus and RULES-3 Plus with BPP

	Rules 3 Plus			Rules 3 Plus with BPP			
Data set name	Rules	Training %	Test %	*NL*	Rules	Training %	Test %
balance_scale	216	100.00	82.54	0.10	41	91.06	83.07
				0.20	14	84.40	84.13
				0.30	5	73.39	70.90
breast_cancer	43	100.00	95.73	0.10	7	93.03	88.15
				0.20	6	93.85	90.52
				0.30	6	93.85	90.52
wdbc	53	100.00	97.67	0.10	12	94.46	95.93
				0.20	8	93.20	95.35
				0.30	4	85.39	88.95
wpbc	46	100.00	60.66	0.10	40	99.27	63.93
				0.20	13	86.86	80.33
				0.30	1	76.64	75.41
car	275	100.00	100.00	0.10	206	100.00	100.00
				0.20	150	90.10	91.30
				0.30	1	70.02	59.00
credit screening	148	98.76	81.73	0.10	40	91.08	83.65
				0.20	6	86.10	84.62
				0.30	3	85.89	84.62
cylinder-band	218	100.00	62.58	0.10	211	100.00	63.19
				0.20	211	100.00	63.19
				0.30	211	100.00	63.19
dermatology	48	100.00	95.58	0.10	39	100.00	94.69
				0.20	39	100.00	94.69
				0.30	39	100.00	94.69
diabetes	224	93.84	65.09	0.10	134	89.93	67.24
				0.20	68	85.45	73.28
				0.30	5	72.57	71.55
ecoli	83	95.24	77.14	0.10	46	92.64	79.05
				0.20	29	89.18	80.95
				0.30	22	87.88	79.05
glass	26	100.00	94.03	0.10	17	99.32	92.54
				0.20	13	97.96	92.54
				0.30	3	81.63	79.10
haberman	53	80.75	77.42	0.10	38	80.28	77.42
				0.20	19	78.40	76.34
				0.30	1	73.71	73.12
iris	12	98.04	91.67	0.10	4	96.08	93.75
				0.20	4	96.08	93.75
				0.30	4	96.08	93.75
liver	86	81.64	55.80	0.10	68	81.16	53.62
				0.20	54	78.74	52.17
				0.30	23	73.43	52.17
tic-tac-toe	140	100.00	95.14	0.10	32	100.00	100.00
				0.20	32	98.66	100.00
				0.30	12	79.55	79.51

7.3 Comparison with C5

The performance of the new algorithm (RULES-5 with IPP) is compared against C5, one of the most efficient inductive learning algorithms currently available. C5 is a divide-and-conquer algorithm and therefore creates decision trees. For the purpose of this comparison, the decision trees have been converted into rule sets. Table 4 shows the results obtained by both algorithms. For this work, it is assumed that an optimal *NL* has been found.

For 7 out of the 15 data sets, namely *balance_scale* ($NL = 0.1$ and 0.2), *breast_cancer* ($NL = 0.1$ and 0.2), *wdbc* ($NL = 0.2$), *credit screening* ($NL = 0.1$, 0.2 and 0.3), *diabetes* ($NL = 0.3$), *ecoli* ($NL = 0.3$) and *tic-tac-toe* ($NL = 0.1, 0.2$ and 0.3), RULES-5 with IPP clearly outperforms C5, creating fewer rules but with the same or higher test accuracies.

With five of the data sets (*wpbc*, *cylinder band*, *dermatology*, *glass* and *iris*) C5 outperforms RULES-5 with IPP when considering both the number of rules created and the test accuracy obtained. However, for some of these data sets, the results for RULES-5 with IPP are still acceptable. For instance, for the *iris* data set the only difference with C5 is one rule.

For two other data sets, *car* ($NL = 0.1$) and *liver* ($NL = 0.2$), the comparison is more difficult because C5 produces smaller rule sets than RULES-5 with IPP but also with *substantially* reduced test accuracies. Thus, the advantage of using one set of rules or the other will depend on the application.

Downloaded from pic.sagepub.com at Cardiff University on March 1, 2012

Table 3 Comparison between RULES-5 and RULES-5 with IPP

	RULES-5			RULES-5 with IPP			
Data set name	Rules	Training %	Test %	*NL*	Rules	Training %	Test %
balance_scale	93	100.00	86.24	0.10	24	88.99	84.13
				0.20	16	85.32	78.84
				0.30	7	76.15	74.60
breast_cancer	33	100.00	96.21	0.10	4	92.62	93.36
				0.20	6	94.26	92.89
				0.30	3	90.78	88.15
wdbc	35	100.00	95.35	0.10	8	93.70	94.19
				0.20	4	92.95	94.19
				0.30	6	92.95	94.19
wpbc	35	100.00	73.77	0.10	23	97.08	72.13
				0.20	10	89.78	68.85
				0.30	1	76.64	75.41
car	246	100.00	100.00	0.10	206	100.00	100.00
				0.20	150	90.45	91.59
				0.30	1	70.02	59.00
credit screening	117	100.00	87.02	0.10	19	89.83	85.10
				0.20	4	86.10	84.62
				0.30	2	85.89	84.62
cylinder-band	230	100.00	66.87	0.10	224	100.00	66.87
				0.20	224	100.00	66.87
				0.30	224	100.00	66.87
dermatology	43	100.00	94.69	0.10	42	100.00	94.69
				0.20	42	100.00	94.69
				0.30	42	100.00	94.69
diabetes	160	100.00	76.29	0.10	61	94.59	74.57
				0.20	24	85.45	74.14
				0.30	7	79.48	76.29
ecoli	53	100.00	80.00	0.10	23	94.37	82.86
				0.20	17	90.48	77.14
				0.30	16	89.18	76.19
glass	17	100.00	94.03	0.10	7	100.00	94.03
				0.20	7	100.00	94.03
				0.30	7	100.00	94.03
haberman	72	99.53	72.04	0.10	48	95.77	68.82
				0.20	11	80.75	77.42
				0.30	1	73.71	73.12
iris	8	100.00	95.83	0.10	5	100.00	95.83
				0.20	5	100.00	95.83
				0.30	5	100.00	95.83
liver	61	100.00	60.87	0.10	49	100.00	57.97
				0.20	22	86.96	68.12
				0.30	17	85.51	63.04
tic-tac-toe	32	100.00	100.00	0.10	26	100.00	100.00
				0.20	26	100.00	100.00
				0.30	11	97.61	100.00

Finally, the performance of C5 on the remaining data set (*haberman*) should be analysed further. When C5 is applied on the *haberman* data set, a rule set is created containing only one rule (IF anything THEN survival). RULES-5 with IPP gives the same result when *NL* is set to 0.3. This rule set fails to represent any interesting pattern contained within the data set because it ignores all but one class. As a result, the rule set is too general. On this particular data set, RULES-5 with IPP demonstrates one advantage over C5. By allowing the user to decide the level of pruning, such overgeneralization can be avoided. For instance, if *NL* is set to 0.2 for this data set, RULES-5 with IPP creates a more accurate rule set that contains 11 more meaningful rules. The same problem occurs with the *cylinder band* data set, for which C5 creates only two rules.

8 CONCLUSION

The proposed new pruning technique significantly improves the performance of the RULES-5 algorithm. The results show that RULES-5, in combination with IPP, outperforms C5. Compared to other pruning techniques, the dependence of process performance on heuristic measures is reduced. This is achieved by employing a new rule merging technique. The user controls the rule merging process by specifying a

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012

Table 4 Comparison between C5 and RULES-5 with IPP

	C5			RULES-5 with IPP			
Data set name	Rules	Training %	Test %	*NL*	Rules	Training %	Test %
balance_scale	34	90.37	79.89	0.10	24	88.99	85.19
				0.20	16	87.61	80.95
				0.30	7	77.75	76.72
breast_cancer	11	97.95	91.94	0.10	4	92.62	93.36
				0.20	6	94.26	92.89
				0.30	3	90.78	88.15
wdbc	10	97.98	95.35	0.10	8	94.21	95.35
				0.20	4	92.95	97.09
				0.30	6	92.95	94.77
wpbc	13	90.51	77.05	0.10	23	97.08	75.41
				0.20	10	89.05	70.49
				0.30	1	76.64	75.41
car	131	96.30	94.10	0.10	206	100.00	100.00
				0.20	150	90.45	91.59
				0.30	1	70.02	59.00
credit screening	43	90.46	79.22	0.10	19	89.83	85.10
				0.20	4	86.10	84.62
				0.30	2	85.89	84.62
cylinder-band	2	71.54	66.87	0.10	224	100.00	66.87
				0.20	224	100.00	66.87
				0.30	224	100.00	66.87
dermatology	8	98.42	96.46	0.10	42	100.00	95.58
				0.20	42	100.00	95.58
				0.30	42	100.00	95.58
diabetes	17	81.34	75.43	0.10	61	92.72	77.59
				0.20	24	83.40	74.57
				0.30	7	79.29	75.43
ecoli	16	17.32	15.24	0.10	23	94.37	82.86
				0.20	17	91.34	76.19
				0.30	16	89.18	76.19
glass	6	100.00	100.00	0.10	7	100.00	94.03
				0.20	7	100.00	94.03
				0.30	7	100.00	94.03
haberman	1	73.71	73.12	0.10	48	95.77	67.74
				0.20	11	80.75	77.42
				0.30	1	73.71	73.12
iris	4	99.02	95.83	0.10	5	100.00	95.83
				0.20	5	100.00	95.83
				0.30	5	100.00	95.83
liver	11	80.68	62.32	0.10	49	100.00	59.42
				0.20	22	89.37	69.57
				0.30	17	86.96	57.25
tic-tac-toe	75	93.73	85.76	0.10	26	100.00	99.31
				0.20	26	100.00	99.31
				0.30	11	88.51	90.63

parameter, *NL*, based on his or her assessment of the noise level present in the data set. Allowing domain experts to contribute to the rule extraction process may prove to be important when real data are analysed. Also, the proposed pruning technique could be regarded as a tool for analysing data sets for the presence of noise. This analysis could be carried out by assessing the changes in the size of rule sets generated and the variation of their accuracy when *NL* varies.

ACKNOWLEDGEMENTS

This work was carried out within the ESPRIT Project 29114 'INFOMAN', the ERDF Project 'Innovation in Manufacturing Centre' and the ERDF (Objective 1) Project 'Supporting Innovative Product Engineering and Responsive Manufacturing (SUPERMAN)'. The authors would like to thank Dr Z. Salem and Dr Z. Cai for their help in the literature survey and technical discussions.

REFERENCES

1 **Mitchell, T. M.** Machine learning and data mining. *Communs ACM*, 1999, **42**(11), 30–36.
2 **Teti, R.** (Ed). Intelligent computation in manufacturing engineering. In Proceedings of the 3rd CIRP International Seminar, Ischia, Italy, 2002.

Downloaded from pic.sagepub.com at Cardiff University on March 1, 2012

3 **Mitchell, T. M.** *Machine Learning*, 1997 (McGraw-Hill, New York).
4 **Quinlan, J. R.** Learning efficient classification procedures and their applications to chess-end-games. In *Machine Learning—An Artificial Intelligence Approach*, Los Altos, California, 1983, pp. 463–482.
5 **Quinlan, J. R.** Induction of decision trees. In *Machine Learning*, 1986, Vol. 1, pp. 81–106 (Kluwer, Boston, Massachusetts).
6 **Quinlan, J. R.** *C4.5: Programs for Machine Learning*, 1993 (Morgan Kaufmann, San Mateo, California).
7 Rulequest Research, *Data Mining Tools See5 and C5*. Available from http://www.rulequest.com/see5-info.html (Accessed 1 February 2003).
8 **Clark, P.** and **Niblett, T.** The CN2 induction algorithm. *Mach. Learning*, 1989, **3**, 261–284.
9 **Clark, P.** and **Boswell, R.** Rule induction with CN2: some recent improvements. In Proceedings of the 5th European Working Session on *Learning*, Porto, Portugal, 1991, pp. 151–163.
10 **Cohen, W. W.** Fast effective rule induction. In Proceedings of the 12th International Conference on *Machine Learning*, Tahoe City, California, 1995, pp. 115–123.
11 **Michalski, R. S.** On the quasi-minimal solution of the general covering problem. In Proceedings of the 5th International Symposium on *Information Processing* (*FCIP 69*), Bled, Yugoslavia, 1969, Vol. A3, pp. 125–128.
12 **Michalski, R. S.** Pattern recognition as rule-guided inductive inference. *IEEE Trans. Pattern Analysis and Mach. Intell.*, 1980, **2**, 349–361.
13 **Michalski, R. S.** and **Kaufman, K. A.** The AQ19 system for machine learning and pattern discovery: a general description and user guide. Reports of the Machine Learning and Inference Laboratory, MLI 01-2, George Mason University, Fairfax, Virginia, 2001.
14 **Fürnkranz, J.** Separate-and-conquer rule learning. *Artif. Intell. Rev.*, 1999, **13**(1), 3–54.
15 **Pham, D. T., Afify, A. A.** and **Dimov, S. S.** Machine learning in manufacturing. In Proceedings of the 3rd CIRP International Seminar on *Intelligent Computation in Manufacturing Engineering* (*ICME 2002*), Ischia, Italy, 2002, pp. III–XII.
16 **Fürnkranz, J.** Pruning algorithms for rule learning. *Mach. Learning*, 1996, **27**(2), 139–171.
17 **Brunk, C. A.** and **Pazzani, M. J.** An investigation of noise-tolerant relational concept learning algorithms. In Proceedings of the 8th International Workshop on *Machine Learning*, Evanston, Illinois, 1991, pp. 389–393.
18 **Cohen, W. W.** Efficient pruning methods for separate-and-conquer rule learning systems. In Proceedings of the 13th International Joint Conference on *Artificial Intelligence*, Chambery, France, 1993, pp. 988–994.
19 **Breslow, A.** and **Aha, D. W.** Simplifying decision trees: a survey. *Knowledge Engng Rev.*, 1996, **12**, 1–40.
20 **Pham, D. T., Bigot, S.** and **Dimov S. S.** RULES-5: a rule induction algorithm for problems involving continuous attributes. *Proc. Instn Mech. Engrs, Part C: J. Mechanical Engineering Science*, 2003, **217**(C12), 1273–1286.
21 **Pham, D. T.** and **Dimov, S. S.** An efficient algorithm for automatic knowledge acquisition. *Pattern Recognition*, 1996, **30**(7), 1137–1143.
22 **Pham, D. T.** and **Dimov, S. S.** An algorithm for incremental inductive learning. *Proc. Instn Mech. Engrs, Part B: J. Engineering Manufacture*, 1997, **211**(B3), 239–249.
23 **Frank, E.** Pruning decision trees and lists. PhD thesis, University of Aikato, Hamilton, New Zealand, 2000.
24 **Bigot, S.** New techniques for handling continuous values in inductive learning. PhD thesis, Cardiff University, Cardiff, Wales, 2002.
25 **Quinlan, J. R.** Simplifying decision trees. *Int. J. Man–Mach. Studies*, 1987, **27**, 221–234.
26 **Fürnkranz, J.** and **Widmer, J.** Incremental reduced error pruning. In Proceedings of the 11th International Machine Learning Conference, New Brunswick, New Jersey, 1994, pp. 70–77.
27 **Oates, T.** and **Jensen, D.** The effects of training set size on decision tree complexity. In Proceedings of the 14th International Conference on *Machine Learning*, San Francisco, California, 1997, pp. 254–262.
28 **Pfahringer, B.** Compression-based pruning of decision lists. In Proceedings of the European Conference on *Machine Learning*, Prague, Czech Republic, 1997, pp. 199–212.
29 **Quinlan, J. R.** Induction of logic programs: FOIL and related systems. *New Generation Computing*, 1995, **13**, 287–312.
30 **Robnik-Sikonja, M.** and **Kononenko, I.** Pruning regression trees with MDL. In Proceedings of the 13th European Conference on *Artificial Intelligence*, Brighton, 1998, pp. 455–459.
31 UCI, *UCI Machine Learning Repository*. Available from http://www1.ics.uci.edu/~mlearn/MLRepository.html (Accessed 1 February 2003).

Downloaded from pic.sagepub.com at Cardiff University on April 4, 2012