Knowledge and Information Systems, Vol.3, No.2, pp.146-167,
2001

A Description Length Based Decision Criterion for Default Knowledge in the Ripple Down Rules Method

Takuya Wada, Tadashi Horiuchi, Hiroshi Motoda, Takashi Washio

Institute of Scientific and Industrial Research, Osaka University, Japan

Abstract. The "Ripple Down Rules (RDR)" Method is a promising approach to directly acquiring and encoding knowledge from human experts. It requires data to be supplied incrementally to the knowledge base being constructed, each new piece of knowledge being added as an exception to the existing knowledge base. Because of this patching principle, the knowledge acquired depends strongly on what is given as the default knowledge, used as an implicit outcome when inference fails.

Keywords: Knowledge Acquisition; Ripple Down Rules Method; Default Knowledge; Description Length

1. Introduction

An implicit assumption of the development of knowledge-based systems is that the expert knowledge is stable and that it is worth investing into knowledge acquisition (KA) from human experts. However, the rapid innovation in technology in recent years makes existing knowledge out-of-date very soon and requires frequent updates. We should now think of expertise not as static but as dynamic in many real world problems. In addition, the advancement of worldwide networks such as the Internet has changed computer usage drastically. It is now possible that many users have access to a single knowledge-based system through a network [11], and further, the need arises for multiple experts to supply knowledge continuously through the same network without knowing exactly how the knowledge base has evolved.

2. Ripple Down Rules Revisited

Ripple Down Rules (RDR) is a knowledge acquisition technique that challenges the *KA bottleneck* problem by allowing rapid development of knowledge bases

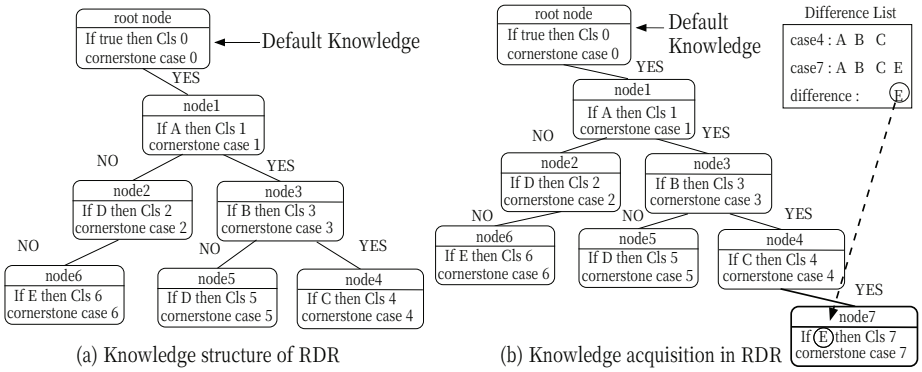


Fig. 1. Knowledge structure of Ripple Down Rules

by human experts without the need for analysis or intervention by a knowledge engineer. From long experience of knowledge-based systems development [2], it is clear that human experts are not good at providing information on how they reach conclusions, rather they can justify that their conclusions are correct [3, 5]. The basis of RDR is the maintenance and retrieval of cases. In this sense, RDR can be thought of as a special instance of Case-Based Reasoning (CBR) [12], which is normally regarded as a method of machine learning.

The tree structure of an RDR Knowledge Base (RDR KB) is shown in Figure 1.(a). Each node in the binary tree is a rule with a desired conclusion. Each node has a cornerstone case associated with it, that is, the case that prompted the inclusion of the rule. The classification (conclusion) comes from the last rule whose condition was satisfied by the current case. The knowledge acquisition process in RDR is illustrated in Figure 1.(b). To add a new rule into the tree structure, it is necessary to identify the conditions of the rule as well as its location.

3. Default Knowledge in the Ripple Down Rules method

The rule at the root node in an RDR KB, which is represented by a binary tree, is special. Its condition part is empty (this means that it is satisfied by any case) and its consequence is a special conclusion called the "default class". Here, the default class means a tacit conclusion for a current case when no other conclusion of the case is derived from the knowledge base.

4. Experimental Design

4.1. Objective

In previous research on RDR, it was common for the majority class in the domain to be assumed as the default class [10]. The majority class in the domain means the class of the most frequent cases in the data set. This looks natural because we only need to acquire rules (knowledge) for other (minority) classes as exceptions to the default class and subsequent refinement rules for the default class. The

Table 1. Summary of 17 data sets

Name	# of Cases	# of Classes	Attribute type	# of Attributes
Car	1728	4	Nom.*	6
Tic-Tac-Toe	958	2	Nom.	9
Nursery	12960	5	Nom.	8
Connect-4	16889†	3	Nom.	42
Mushroom	8124	2	Nom.	22
Titanic	2201	2	Nom.	3
Ann-thyroid	7200	3	Mixed***	15/6
Pendigits	10992	10	Num.**	16
Iris	150	3	Num.	4
Page-block	5473	5	Num.	10
Optdigits	5620	10	Num.	64
Yeast	1484	10	Num.	8
Waveform	5000	3	Num.	21
Image	2310	7	Num.	19
Cmc	1473	3	Mixed	7/2
German	1000	2	Mixed	13/7
Shuttle	14500	7	Num.	9

* Nominal attribute. ** Numerical attribute. *** This data set has two kinds of attributes: nominal attribute / numerical attribute. † This data set consists of 25% cases which are selected form the whole 67557 cases by random sampling.

size of the knowledge base is expected to be compact. But is this really the case? The objective of this paper is to answer this question. The fact that a class is the majority class doesn't necessarily mean that it is easy to describe and characterize that class.

4.2. Experimental Setting

We have carefully designed experiments and conducted parametric studies using different data sets. RDR requires human expertise. Since we don't have expertise on the data sets we use, we instead use a simulated expert [6] as a replacement. A simulated expert has a knowledge base constructed by a computer program and is one kind of expert system. For each default class used, we evaluate the classification accuracy and the knowledge base size with different noise levels and different training data size. The results are compared with respect to different values of these parameters and with a machine learning method.

1) Data Sets We have selected 16 data sets from University of California Irvine Data Repository [1] and 1 data set from University of Toronto Data Repository [23]. Of these, 7 data sets have all nominal attributes, 7 data sets all numerical attributes and 3 data sets mixed attributes (see Table 1).

2) Machine learning method The RDR KB is compared with the knowledge base built separately using the standard machine learning method C4.5 [18].

3) Simulated expert As a simulated expert, we also use C4.5 [18]. We run C4.5 with the default setting, using the entire data for each noise level and for each data set. This is the maximum attainable performance that can be induced from the given data and we use this as the knowledge (expertise) of an expert.

Here again, we use the induction rule set (c4.5rules). C4.5 derives the rule set from the decision tree in the following way.

A human expert checks the output of the RDR system, and when the conclusion is wrong, she generates a new rule by selecting conditions from the difference list. The simulated expert acts in exactly the same way and selects conditions from the difference list. To imitate human expert behavior, all the elements in the intersection between the conditions of the rule for which the case applies (expert knowledge) and the difference list are selected, although it is true that RDR eventually constructs a correct tree by selecting only one condition from the intersection. It should not be expected, however, that the simulated expert would perform better than a real human expert. When there is noise, the simulated expert may misclassify some cases. In this case we use the incorrect classification by the simulated expert as the conclusion of a human expert. It is also noted that the simulated expert has a slight error rate even when the data is noise-free. This is because the pruning is made in C4.5.

4) Noise handling Noise handling is different for nominal attributes and numerical attributes. For nominal attributes, the feature values are changed randomly to other alternatives with a specified fraction (noise level).

5) Data preparation Since RDR accepts data incrementally, the order of incoming data affects the performance. The whole data for each domain is reordered by random sampling and ten different data sets are generated to cancel out this ordering effect. For each data set of a fixed order, the first 75% of the data is taken as training data and the remaining 25% of the data as test data.

6) Accuracy The predicted error rate is used as a measure of accuracy. It is the average of the error rates for the remaining 25% data of the ten data sets, each with a different ordering.

7) Size of knowledge base The size of the knowledge base is defined as the sum of nodes in RDR binary tree and the number of induction rules in C4.5. This is in favor of C4.5 because the number of conditions in one rule of C4.5 is normally larger than the number of conditions in one node of RDR tree.

4.3. Methods to calculate Description Length

DL reflects both induction rules and cases misclassified by the simulated expert for each class. The value of this DL is calculated as the sum of the coding cost for induction rules and that for misclassified cases. We devised three coding methods that are explained below. Each of them is based on a slightly different view but has a clear meaning. They are entirely different from the coding method used in C4.5, which seems to use a coding based on empirical evidence.

1) DL by permutation based coding

The number of bits required to specify the induction rules for a certain class is calculated by

$$RuleBits = \sum_{i=1}^{N} \left\{ \log_2(AttNum) + \sum_{j=1}^{M_i} CondBits(i,j) \right\}, \tag{1}$$

where N is the total number of the induction rules, M_i is the number of conditions in the i-th induction rule, $CondBits(i,j)$ is the number of bits required to specify the j-th condition of the i-th induction rule and $AttNum$ is the number of attributes in the problem domain. The term $\log_2(AttNum)$ represents the number of bits required to specify the number of conditions (M_i) of the i-th induction rule. $CondBits(i,j)$ is calculated separately depending on the type of attributes.

a) Discrete attributes

$$CondBits(i,j) = log_2(AttNum - j + 1) + log_2(ValueNum(i,j)), \quad (2)$$

where $ValueNum(i,j)$ is the number of discrete attribute values of the j-th condition of the i-th induction rule. The first term represents the number of bits required to specify the attribute of the j-th condition (the first condition has $AttNum$ choices, the second $AttNum-1$, thus the j-th $AttNum-j+1$). The second term represents the number of bits required to specify the attribute value of the j-th condition (there are $ValueNum(i,j)$ choices).

b) Continuous attributes

We assume that the inequalities used in the conditions of the continuous attributes are restricted to only two types, because the induction rules given from the decision tree in C4.5 have only two inequalities ("$\leq$", "$>$").

$$CondBits(i,j) = log_2(AttNum-j+1)+log_2(2)+log_2(CutOff(i,j)),(3)$$

where $CutOff(i,j)$ represents the number of boundaries for discretization of the continuous attribute of the j-th condition of the i-th induction rule. The first term is the number of bits required to specify the attribute of the condition. The second term represents the number of bits required to specify the type of the inequalities and the third term represents the number of bits required to specify the boundary of the continuous attribute of the condition.

The number of bits required to specify the misclassified cases is

$$ExceptionBits =$$
$$\log_2\left(\frac{r}{2}+1\right)+\log_2\left(\begin{pmatrix} r \\ fp \end{pmatrix}\right)+\log_2\left(\frac{n-r}{2}+1\right)+\log_2\left(\begin{pmatrix} n-r \\ fn \end{pmatrix}\right),(4)$$

where the induction rules cover r cases out of the total n cases with fp false positives and fn false negatives. The first term represents the number of bits required to describe f_p (the number of errors is less than half of r). The second term represents the number of bits required to specify the false positive cases among r cases covered by the induction rules. The third term represents the number of bits required to describe f_n and the fourth term the number of bits required to specify the false negative cases among $n-r$ cases which are not covered by the induction rules.

It is difficult to judge which encoding method is the best of the three for deciding the default class. Note that it is always possible to devise a longer encoding. All of the above three give a good and efficient coding length and the results shown later indicate that all are good candidates.

Table 2. Car Evaluation Database

Class		*unacc*	*acc*	*good*	*vgood*
Simulated Expert	Number of cases	1210	384	69	65
	Number of induction rules	11	53	15	15
	Number of misclassified cases	33	56	33	15
	Description length (bits)**	254.9	688.4	265.9	226.6
RDR	Accuracy*	3.5	2.9	3.8	3.7
	Acquisition speed*	11.5	6.4	11.3	12.5
	Size of knowledge base*	89.1	41.9	84.3	97.3

* Both the accuracy (%) and the size are evaluated at 75% of the total data seen. The Acquisition speed is the accuracy at 20% of the total data seen.
** The calculation of this DL is based on "3)DL by combination-based coding" in section 4.3

5. Experimental Results

5.1. Characterization of Default Class

We show only the detailed results of the data set "Car" out of the 17 data sets and the summary results of the whole data sets. The "Car" data set gives a typical result.

Table 2 summarizes the data characteristics and the main results. This data set has 6 nominal attributes and 4 classifications which are *unacc* (unacceptable), *acc* (acceptable), *good* and *vgood* (very good). The first three rows are 1) the number of cases for each class (total of 1728 cases showing that *unacc* is the majority class), 2) the number of induction rules for each class that was derived by c4.5rules [1], and 3) the number of cases misclassified by the simulated expert C4.5 for each class.

5.2. A Decision Criterion for Default Knowledge

From the observations in section 5.1, we can conjecture that the more induction rules are required to describe a certain class and the more cases are misclassified by the simulated expert for the class, the faster the acquisition speed of RDR is and the smaller the size of RDR knowledge base is when this class is used as the default class. By having a class that requires more induction rules to describe itself as the default class, we can avoid adding these rules later in the knowledge acquisition process. Likewise by having a class for which there are more cases misclassified by the simulated expert, the refinement rules within the other classes can be more error free.

Although the results in Tables 3 and **??** are not as good as we hoped (only 71%), in terms of the difference of different coding methods all three are good enough and there is no clear difference among the three. We thus adopt method **3) DL by combination-based coding** to evaluate the best default class.

Figures **??** and **??** show the relation between the accuracy and the description lengths of the four RDRs for three different noise levels: 0%,10% and 20% at the point where the first 20% and 50% of data are seen respectively in the "Car" data set. This is to see the effect of noise both at an earlier stage of knowledge

[1] This corresponds to the simulated expert knowledge for the class.

Table 3. Comparison of the Results by the three Coding Methods (without noise)

Coding Method	Accuracy		Speed		KB size		Summary
	MaxDL	≤30%	MaxDL	≤30%	MaxDL	≤30%	
"Permutation"	14/17	14/17	15/17	15/17	11/17	12/17	11/17
"Power"	13/17	13/17	15/17	15/17	12/17	13/17	11/17
"Combination"	14/17	14/17	15/17	15/17	12/17	13/17	12/17

"MaxDL" : The numerator shows the number of data sets for which Max. DL gives the best performance for the item in the top row (*i.e.*, Accuracy, Speed and KB size). "≤30%" : The numerator shows the number of data sets for which Max. DL class ranks within the top 30% for the item in the top row. "Summary" : The numerator shows the number of data sets for which Max. DL class ranks within the top 30% for all the items in the top row.

acquisition where the system has not yet seen enough data and a later stage where the system's performance is approaching to the equilibrium. It is observed that the error rates get smaller as more data are seen for all noise levels and for all default classes, and the error rates of all four RDRs become higher as the data set becomes noisier. However, RDR which has the default class with the maximum description length has the best accuracy of the four RDRs for most of the different noise levels. The same results are obtained for the other data sets. We can conclude that RDR with the default class which has the maximum description length has good accuracy even when there is noise in the data set.

How good is the majority class as a default? For the results of noiseless data sets, there are 8 data sets for which the Max. DL principle results in the majority default class. There are 6 data sets for which the overall performance is best if the majority class is used as the default. The corresponding number is 12 if the Max. DL principle is used to select the default class. There are only 6 data sets among these 12 for which the best default matches the majority class. Table **??** compares the performance difference between the Max. DL default class and the majority default class. For these data sets for which the Max. DL class is not the same as the majority class, the average error rate of the former is about 80% smaller than that of the latter. It is evident that the majority class is not necessarily the best default class.

So far we have assumed that it is possible to calculate DL. In order to calculate DL, we need to have a rule set and its error rate, which requires enough data to have been accumulated. Thus, we have conducted additional evaluation where we estimated the description length (by the combination based coding) when only the first 10% of the data is seen. The result indicates that the relative values of the description length can be estimated well by using a small fraction of data. The number of cases where the correct default class was predicted was 11 out of 17 data sets. However, the default classes for these 11 data sets are not necessarily a subset of those for the 12 data sets for which DL is evaluated when 100% data is seen. Two data sets that were "Yes" in summary with 100% data became "No" (the number of data for these two data sets is small (10% is less than 100) to estimate DL) and one data set that was "No" became "Yes" (this is by chance). This means that we only need wait until a small amount of data is accumulated before initiating knowledge acquisition using the RDR method to construct a high performance knowledge base.

5.3. Confirming the Generality of the Decision Criterion

In the previous sections we made the assumption that the knowledge of the expert has a small description length, *i.e.*, following Occam's razor, or the minimum description length principle, and showed that the *maximum description length principle* (Max. DLP) is a good criterion for selecting the best default knowledge in RDR. We confirm that our criterion is indeed a good criterion, even when the knowledge of the expert is poor (*i.e.*, it's description length is not minimum), by performing additional experiments. In order to simulate this factor, we take a rule set that has a maximum DL instead of the minimum DL in C4.5. This normally selects a rule set that is unbalanced between the coding length of rules and misclassified cases, with more emphasis on the latter because of the weighting factor. We repeated the experiment for the same noiseless data sets mentioned in section 4.2 using poor expert knowledge. DL that was used to select the best default class was evaluated by combination based coding.

Results of this experiment are shown in Table **??**. The Max. DL default class is the best default class for 10 out of 16 data sets in overall evaluation. Interestingly, it coincides with the best default class for all of the 16 data sets in terms of "Accuracy" and "Speed". It should be noted that "Yes" here is relative to the alternative default classes and the accuracy attained is of course lower than, in cases where good expert knowledge is used.

6. Discussions

Application of Max. DL criterion to human expert knowledge

We have evaluated RDR using a simulated expert that is a computer program based on a machine learning technique instead of a human expert. Therefore, the application of Max. DL principle is straightforward. However, it is not self-evident how to apply this principle to human-expert knowledge. As RDR is a methodology for acquiring useful knowledge from the vague and disordered knowledge of a human expert by forcing him/her to be placed in a specific context, it may not be the case that the expert can estimate the coding length of the knowledge and the number of misclassified cases in advance. However, there must be some difference in certainty or confidence for the knowledge which the expert has for each class, and the result in the previous section shows that it is better to use the least easily describable class, and the class the expert is least confident about, as the default class in RDR. This doesn't necessarily mean using the majority class.

Minimum Description Length Principle

Occam's razor, or minimizing the description length, is the normal practice for selecting the most plausible of many alternatives. What we propose here is its reverse and may look contradictory, but it is not. Our Max. DL principle applies to choosing the best default class, and by choosing the best default class, the total description length of the RDR becomes the smallest. How small it is depends on how good the expert knowledge is.

7. Conclusion

This paper describes experimental results about the effect which the selection of default knowledge and the amount of noise in data have on the performance of RDR, which is a knowledge acquisition technique which dispenses with the need for analysis or intervention by a knowledge engineer. Because of the patching principle in RDR, the knowledge acquired strongly depends on what is given as the default knowledge.

The above characteristics of RDR will be useful in building a knowledge-based system which assumes a network environment, where multiple users and experts interact with each other through the computer network. Acquiring or updating knowledge from multiple experts in a network environment is a new challenge.

Acknowledgement

The authors have benefited greatly from discussions with Dr. B. H. Kang of the University of Tasmania (Australia).

References

[1] C. Blake, E. Keogh, C. J. Merz. UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.
[2] P. Compton, K. Horn, J. R. Quinlan, L. Lazarus. Maintaining an Expert System. In: J. R. Quinlan (ed.), *Application of Expert Systems*, Addison Wesley, 1989, pp. 366–385.
[3] P. Compton, R. Jansen. A Philosophical Basis for Knowledge Acquisition, *Knowledge Acquisition* **2**, 241–257, 1990.
[4] P. Compton, G. Edwards, B. H. Kang et al. Ripple Down Rules: Possibilities and Limitations. In: *Proc. of the 5th Knowledge Acquisition for Knowledge Based Systems Workshop*, 1991.
[5] P. Compton, G. Edwards, G. Srinivasan et al. Ripple Down Rules: Turning Knowledge Acquisition into Knowledge Maintenance, *Artificial Intelligence in Medicine* **4**, 47–59, 1992.
[6] P. Compton, P. Preston, B. H. Kang. The Use of Simulated Experts in Evaluating Knowledge Acquisition. In: *Proc. of the 9th Knowledge Acquisition for Knowledge Based Systems Workshop*, 1995.
[7] B. Gaines, M. Shaw. Cognitive and Logical Foundations of Knowledge Acquisition. In: *Proc. of the 5th Knowledge Acquisition for Knowledge Based Systems Workshop*, 1990.
[8] D. K. Gary, J. H. Trevor. Optimal Network Construction by Minimum Description Length, *Neural Computation* **5**, 210–212, 1993.
[9] B. H. Kang, P. Compton. Knowledge Acquisition in Context: Multiple Classification Problem. In: *Proc. of the Second Pacific Rim International Conference on Artificial Intelligence 2*, 1992, pp. 847–853.
[10] B. H. Kang. *Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules*, PhD Thesis, Department of Electrical Engineering, University of New South Wales, Australia, 1996.
[11] B. H. Kang, K. Yosida, H. Motoda, P. Compton. Help Desk System with Intelligent Interface, *Applied Artificial Intelligence* **11**, 611–631, 1997.
[12] J. L. Kolodner. *Case-Based Reasoning*, Morgan Kauffmann, San Francisco, 1993.
[13] J. L. Kolodner. A Process Model of Case-Based Reasoning in Problem Solving. In: *Proc. of the 9th International Joint Conferences on Artificial Intelligence 1*, 1985, pp. 284–290.
[14] Y. Mansuri, J. G. Kim, P. Compton, C. Sammut. An Evaluation of Ripple Down Rules. In: *Proc. of the First Australian Knowledge Acquisition for Knowledge-Based Systems Workshop*, 1991.

[15]K. Morik, S. Wrobel, J. Kietz, W. Emde. *Knowledge Acquisition and Machine Learning: Theory, Methods, and Applications*, Academic Press, 1993.
[16]J. R. Quinlan, R. L. Rivest. Inferring Decision Trees Using the Minimum Description Length Principle, *Information and Computation* **80**, 227–248, 1989.
[17]J. R. Quinlan. Induction of decision trees, *Machine Learning* **1**, 81–106, 1986.
[18]J. R. Quinlan. *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
[19]Z. Ramadan, P. Compton, P. Preston, T. Le-Gia, V. Chellen, M. Mulholland, D. B. Hibbert, P. R. Haddad, B. H. Kang. From Multiple Classification RDR to Configuration RDR. In: *Proc. of the 10th Knowledge Acquisition for Knowledge Based System Workshop*, 1997.
[20]J. Rissanen. Modeling by Shortest Data Description, *Automatica* **14**, 465–471, 1978.
[21]R. C. Schank. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*, Cambridge University Press, 1982.
[22]J. Suzuki. A Construction of Bayesian Networks from Databases on an MDL Principle. In: *Proc. of the 9th Conference on Uncertainty in Artificial Intelligence*, 1993, pp. 266–273.
[23]The University of Toronto. Data for Evaluating Learning Valid Experiments. http://www.cs.utoronto.ca/~delve/data/datasets.html, 1998.
[24]T. Wada, T. Horiuchi, H. Motoda, T. Washio. Characterization of Default Knowledge in Ripple Down Rules Method. In: *Proc. of the Third Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD'99*, Beijing, China, April, 1999. Lecture Notes in Computer Science Series, Springer-Verlag, 1999, pp. 284–295.
[25]C. S. Wallace, J. D. Patrick. Coding Decision Trees, *Machine Learning* **11**, 7–22, 1993.
[26]S. Wrobel. *Concept Formation and Knowledge Revision*, Kluwer Academic Publishers, 1994.

A. Description length used in C4.5

Calculate the frequency p_{ij} of the j-th attribute value of the i-th attribute Att_i for all cases, and then calculate the following equation for each attribute $Att_i (i = 1, ..., M)$.

$$BranchBits(Att_i) = \sum_{j=1}^{Num\,of\,Att_i\,values} \{-p_{ij} \log_2 p_{ij}\} \tag{5}$$

Sum them over all attributes.

$$SumBranches = \sum_{i=1}^{M} BranchBits(Att_i) \tag{6}$$

Calculate the value called $AttTestBits$.

$$AttTestBits = -\sum_{i=1}^{M} \left\{ \frac{BranchBits(Att_i)}{SumBranches} \log_2 \frac{BranchBits(Att_i)}{SumBranches} \right\} \tag{7}$$

The number of bits required to encode the j-th condition of the r-th induction rule is calculated by

$$CondBits(r, j) = \frac{AttTestBits}{REDUNDANCY} + BranchBits(Att(r, j)), \tag{8}$$

where $Att(r, j)$ represents the attribute of the j-th condition of the r-th induction rule and $REDUNDANCY$ is the inverse ratio of the actually used attributes. The number of bits required to encode the condition part of the r-th induction rule is calculated by

$$Bits(r) = \sum_{j=1}^{M_r} CondBits(r, j) + \log_2(M_r) - \log_2(M_r!), \tag{9}$$

where M_r is the number of conditions in the r-th induction rule. Then, the number of bits required to encode N induction rules for a certain class is calculated by

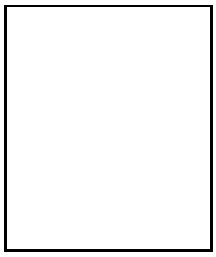
$$RuleBits = \sum_{r=1}^{N} \left\{ \sum_{j=1}^{M_r} CondBits(r, j) + \log_2(M_r) - \log_2(M_r!) \right\} - \log_2(N!). \tag{10}$$

The number of bits required to encode the misclassified cases is

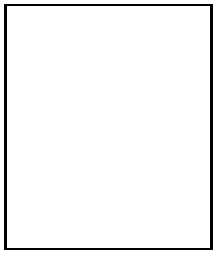
$$ExceptionBits = \log_2 \left(\left(\begin{array}{c} r \\ fp \end{array} \right) \right) + \log_2 \left(\left(\begin{array}{c} n-r \\ fn \end{array} \right) \right), \tag{11}$$

where the rules cover r of the n training cases with fp false positives and fn false negatives. Here, a weighting factor of 0.5 is used for the former because it is known that the rule-coding length as defined in the above equation is overestimated. This value is the default setting in C4.5.

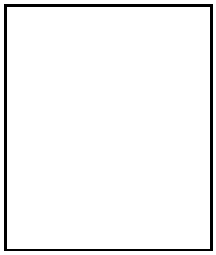
Author Biographies



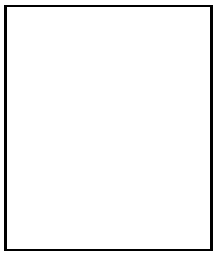
Takuya Wada received his master's degree in Communication Engineering in 1999 from Osaka University, Japan. He is currently a PhD candidate at the Institute of Scientific and Industrial Research of Osaka University . His current research interests include knowledge acquisition and machine learning, in particular integration of these two; *i.e.* KA from human experts and one from data. He is a member of Japanese Society for Artificial Intelligence.



Tadashi Horiuchi is a research associate in the division of Intelligent Systems Science at the Institute of Scientific and Industrial Research of Osaka University. His current research interests include machine learning, knowledge acquisition, knowledge discovery, data mining and soft computing. He received his Bs (1992), Ms (1994) and Ph.D. (1999) degrees in precision engineering from Kyoto University. He is a member of IEEE Computer Society, JSAI, IPSJ, SICE and SOFT.



Hiroshi Motoda is a professor in the division of Intelligent Systems Science at the Institute of Scientific and Industrial Research of Osaka University. Before joining the university, he had been with Hitachi since 1967 and reached the position of a senior chief research scientist at the Advanced Research Laboratory where he headed an AI group and conducted research on machine learning, knowledge acquisition, diagrammatic reasoning and information filtering. His current research interests includes, in addition to these, scientific knowledge discovery and data mining. He received his Bs (1965), Ms (1967) and PhD (1972) degrees in nuclear engineering from university of Tokyo. He is now on the editorial board of Artificial Intelligence in Engineering, International Journal of Human-Computer Studies and Knowledge and Information Systems: An International Journal. He received the outstanding achievement award from JSAI (1999). He is a member of AAAI, IEEE Computer Society, JSAI, JSSST, IPSJ and JCSS.



Takashi Washio graduated at Dept. of Nuclear Eng., Tohoku Univ. in 1983, and took his Ms.E. and Ph.D. in the same department in 1985 and 1988 respectively. He was a visiting researcher in Nuclear Reactor Lab. of Massachusetts Institute of Technology (MIT) from 1988 to 1990, and was a senior researcher in Mitsubishi Research Institute, Inc. in Tokyo, Japan from 1990 to 1996. He has researched on the techniques of qualitative reasoning, diagnosis theory and risk analysis for large-scale plants. He became an associate professor of Institute for the Scientific and Industrial Research (ISIR), Osaka Univ. in 1996. His current research interests are automated scientific law discovery and industrial data mining techniques. He is the member of AAAI, Japanese Society for Artificial Intelligence, Society of Instrument and Control Engineers, Information Processing Society of Japan and Japan Society for Fuzzy Theory and Systems.

correspondence and offprint requests to: Takuya Wada, Institute for the Scientific and Industrial Research, University of Osaka, 8-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan. Email:wada@ar.sanken.osaka-u.ac.jp