Mathematical and Computational Applications, Vol. 13, No. 1, pp. 51-60, 2008.

A REVIEW OF RULES FAMILY OF ALGORITHMS

Mehmet Sabih Aksoy
College of Computer and Information Sciences
King Saud University, P.O. Box 51178
Riyadh 11543 Saudi Arabia
msaksoy@ccis.ksu.edu.sa
sabihaksoy@yahoo.com

Abstract- In recent years, there has been a growing amount of research on inductive learning. Out of this research a number of promising algorithms have surfaced. In the paper after a brief description of knowledge acquisition, induction and inductive learning; RULES family of inductive learning algorithms, their strengths as well as weaknesses are explained and discussed. The applications of inductive learning and particularly the applications of RULES family of algorithms are overviewed.

Key words- Induction, Inductive Learning, Knowledge Acquisition, Expert Systems

1. KNOWLEDGE ACQUISITION

Knowledge-based expert systems consist of two main components: a knowledge base and an inference mechanism. Collecting knowledge to form the knowledge base is the main task in the process of building an expert system [1,2,3].

The process of acquiring knowledge through interaction with an expert consists of a prolonged series of intense, systematic interviews, usually extending over a long period [4]. Human experts are capable of using their knowledge in their daily work, but they usually cannot summarize and generalize their knowledge explicitly in a form which is sufficiently systematic, correct and complete for machine representation and application [1]. Expert systems require large amounts of knowledge to achieve high levels of performance, yet the acquisition of knowledge is slow and expensive [5]. The shortage of trained knowledge engineers to interview experts and capture their knowledge is another problem of knowledge acquisition [6].

The aforementioned problems are not just difficulties of the early days of the technology, but are still acknowledged today as paramount problems. Knowledge acquisition (and in particular machine learning) has become a major area of concern for expert systems research [5,7].

An alternative method of knowledge acquisition exists in which knowledge is learned, or induced, from examples. While it is very difficult for an expert to articulate his knowledge, it is relatively easy to document case studies of the expert's skills at work [5]. Instead of asking an expert to summarize and articulate his knowledge, the main idea of automatic induction is to have him provide a basic structure of his discipline. The knowledge itself will be induced from examples expressed in this structure. Recent developments have proved that this method of knowledge acquisition is entirely possible. Indeed, the main feature of the second generation expert systems is that the knowledge acquisition process is highly automated [8].

2. INDUCTION AND INDUCTIVE LEARNING

In recent years, there has been a growing amount of research on inductive learning [9]. In its broadest sense, induction (or inductive inference) is *a method of moving from the particular to the general - from specific examples to general rules* [5,10,11]. Induction can be considered the process of generalizing a procedural description from presented or observed examples [12,13,14].

The purpose of inductive learning is to perform a synthesis of new knowledge, and this is independent of the form given to the input information [15]. In order to form a knowledge base using inductive learning, the first task is to collect a set of representative examples of expert decisions. Each example belongs to a known class and is described in terms of a number of attributes. These examples may be specified by an expert as a good tutorial set, or may come from some neutral source such as an archive. The induction process will attempt to find a method of classifying an example, again expressed as a function of the attributes, that explains the training examples and that may also be used to classify previously unseen cases [5]. The outcome of an induction algorithm is either a decision tree or a set of rules. Production rules can easily be extracted from decision trees [16,17].

3. RULES FAMILY OF ALGORITHMS

3.1. Rules-1

Pham and Aksoy have developed RULES-1 (RULe Extraction System-1) [18] for extracting a set of classification rules for a collection of objects belonging to a given set of classes. An object must be described in terms of a fixed set of attributes, each with its own set of possible values. For example "Weather" and "Temperature" might be attributes with sets of possible values {rainy, sunny, snowy} and {low, average, high} respectively.

An attribute-value pair constitutes a condition. If the number of attributes is N_a, a rule may contain between one and N_a conditions, each of which must be a different attribute-value pair. The conjunction of conditions only is permitted in a rule and therefore the attributes must all be different if the rule comprises more than one condition. The attributes and the values associated with them in a collection of objects form an array of attributes and values. The total number of elements of the array is the total number of all possible values. For example if there are four attributes with 3,4,2 and 5 values respectively, the total number of elements is 14.

The rule forming procedure may require at most N_a iterations. The first iteration produces rules with one condition and the second iteration results in rules with two conditions, etc. In the first iteration, each element of the array of attributes and values is examined to decide whether it can form a rule with that element as the condition. For the whole set of examples if a given element applies to only one class, then it is a candidate for forming a rule. If it pertains to more than one class, it is passed over and the next element is examined. When all elements of the array have been looked at, the whole set of examples is checked for any example that cannot be classified by the

candidate rules. If there are no unclassified examples the procedure terminates. Otherwise, a new array is constructed which comprises attributes and values contained in all the unclassified examples. In the second iteration elements of the array are examined in pairs to determine whether they apply to only one class in the whole set of examples. As before, for those pairs of elements that pertain to unique classes, candidate rules are obtained. If there are still unclassified examples at the end of this iteration, a new array is formed and the next iteration is initiated. This procedure continues until all examples are correctly classified or the number of iterations (the number of conditions, N_c) is equal to N_a . In the latter case, all remaining unclassified examples are taken as rules. For each iteration after the first, candidate rules extracted in the current iteration are checked against previously obtained rules. Candidate rules that do not contain irrelevant conditions are added to the rule set and the others are ignored. This check is not required for the first iteration as each rule can only have one condition.

RULES-1 does not suffer from the irrelevant-condition problem. As it does not require all examples to be kept in the main memory of the computer at once, it is economical in memory space while not even needing to use windowing.

A disadvantage of RULES-1 is that, the training time is long especially when dealing with the problems having big number of attributes and values. This is because of the fact that, RULES-1 uses an array of values of all unclassified examples in each iteration to extract rules. Especially when the number of conditions is big it may take too much time to extract the rule set. Another disadvantage of RULES-1 is that, the number of selected rules is too big because there is no control over that. Finally the algorithm cannot deal with numerical values and incomplete examples.

3.2. Rules-2

Pham and Aksoy have improved RULES-1 and produced RULES-2 [19]. The rule forming strategy of RULES-2 is almost the same as that of RULES-1. The difference is that instead of considering the values of all unclassified examples, in each iteration, only the values of one unclassified example are used to produce rules for classifying that example. Compared to RULES-1, RULES-2 is generally faster as it requires fewer rule searching operations in its induction process. Furthermore, it allows the user to specify the number of rules to be extracted, is able to deal with incomplete examples and can handle attributes with numerical as well as nominal values.

As only one unclassified example is used in each iteration to obtain rules, it is simple to control the upper limit for the number of rules to be extracted. Three options are available in RULES-2 enabling the user to set that limit as the minimum possible number (which is 1), the number of all extractable rules, or an intermediate number, *n*. The induction time will be lowest for the first option and the accuracy or generalization ability, highest for the second option. Unlike the former algorithm, RULES-2 automatically avoids irrelevant conditions without requiring an additional step during the induction process to eliminate them.

In many real problems, there could be incomplete examples, that is examples in which the values of some attributes are unknown. RULES-1 would not be able to handle those problems. In RULES-2, attributes for which values are not available are

assigned a "NULL" value. During the induction process, such values are simply ignored and therefore do not interfere with the operation of the algorithm.

Engineering problems generally involve attributes with numerical values in addition to attributes with nominal values. RULES-2 can also deal with attributes having numerical values by quantising them. The ranges of values of these attributes and the number of quantization levels for each range are specified by the user. From the given set of examples, RULES-2 constructs a new set for which the values of all numerical attributes are represented by appropriate quantization levels. Induction is then carried out with the new set of examples, the quantization levels being treated as any other nominal values.

3.3. Rules-3

The third version of RULES family of automatic rule extraction systems is RULES-3 [20]. RULES-3 inherits all advantageous features of its predecessors. In addition it has two new features: (1) it provides the user with the option of adjusting the precision of the extracted rules and (2) it generates a compact set of more general rules.

RULES-3 allows the user to specify the minimum number of conditions for each rule. The benefits of doing this are: (i) it will result in a more precise set of rules and (ii) it will help to decrease the number of searching operations (the training time) required to find the rule set particularly when there are many attributes. The default value for the minimum number of conditions is one.

If the number of extracted rules for the example being processed is higher than one then each rule is checked to find how many examples it can classify. The rule which classifies the most examples is chosen and the others are discarded. This will result a more compact set of general rules.

3.4. Rules-3 Plus

Compared to RULES-3, RULES-3 Plus has two new important features. First, it uses a more efficient rule searching procedure instead of the exhaustive search conducted in RULES-3. Second, it uses a simple metric for sorting and selecting candidate rules according to their accuracy and generality. The algorithm is explained by Pham and Dimov as follows [21].

A SET of Attributes and Values (SETAV) is formed using attribute-value pairs used in the example being considered. The total number of elements is the number of attributes in the example. In the first iteration, each element of the set is tested to decide whether it can form a rule. If an element in the set applies to only one class, for the whole set of examples, then it is a candidate to form a rule otherwise it is added to another set called *Partial Rules SET* (PRSET). The maximum number of expressions in PRSET which determines the number of alternatives to be examined in the next pass, can be specified by the user. Each expression in PRSET is specialized by appending to it a single condition from SETAV. The appended condition is discarded if it is already included in the expression. RULES-3 Plus uses a metric called the *H measure* to evaluate the information content of the expressions during the rule forming process. For a given expression, *H measure* is defined as:

$$H = \sqrt{\frac{E^c}{E}}.\left[2 - 2\sqrt{\frac{E_i^c E_i}{E_c E}} - 2\sqrt{\left(1 - \frac{E_i^c}{E_c}\right)\left(1 - \frac{E_i}{E}\right)}\right] \quad (1)$$

where E^c: is the number of examples covered by the expression (the total number of examples correctly classified and misclassified by a given rule), E is the total number of examples, E_i^c is the number of examples covered by the expression and belonging to the target class i (the number of examples correctly classified by a given rule), and E_i is the number of examples in the training set belonging to the target class i. In equation (1), the first term

$$G = \sqrt{\frac{E^c}{E}} \quad (2)$$

relates to the generality of the rule and the second term,

$$A = 2 - 2\sqrt{\frac{E_i^c E_i}{E_c E}} - 2\sqrt{\left(1 - \frac{E_i^c}{E_c}\right)\left(1 - \frac{E_i}{E}\right)} \quad (3)$$

relates to its accuracy. During the rule forming process, the expressions in PRSET are ordered according to their *H measures*. If the number of extracted rules in a given iteration is greater than one, then the rule that has the highest H measure is chosen and the others are ignored. When using the rule set formed by RULES-3 Plus to classify a new example, there could be three outcomes. They are:

• only one rule covers the new example and the example is correctly classified by that rule;

• more than one rule covers the example, in this case, the rule with the largest *H measure* is chosen to classify the example;

• no rules can classify the example or it is misclassified. In this case, the example is added to the training set and the induction process re-initiated.

In RULES-3 Plus, only the expressions in PRSET (there are m_{PRSET} such expressions) are specialized. In RULES-3 all possible combinations of conditions in each iteration are considered. This is the main difference between RULES-3 and RULES-Plus. The PRSET contains the expressions that have the highest information content among all partial rules formed in each iteration. This new rule forming procedure reduces the search space significantly. In RULES-3 and RULES-2, in the worst case, for each example, the total number of all possible combinations is given by:

$$N_{\max} = \sum_{i=1}^{n_a} \frac{n_a!}{(n_a - i)!.i!} \quad (4)$$

Where n_a is the number attributes.

The corresponding number for RULES-3 Plus, is:

$$N'_{\max} = n_a + m_{PRSET} \sum_{i=1}^{n_a - 1} (n_a - i) \quad (5)$$

$N'_{\max}$ is generally smaller than N_{max} . It means RULES-3 Plus considers fewer partial rules than RULES-3 and its predecessors during the rule forming process.

3.5. Rules-4

RULES-4 is the first incremental learning algorithm in the family. It can update and refine the acquired knowledge for new coming examples. RULES-4 uses the same rule forming procedure with RULES-3 Plus. The incremental induction procedure of RULES-4 is given by Pham and Dimov as follows [22]:

Input: Short-Term Memory (STM), Long-Term Memory, ranges of values for numerical attributes, frequency distribution of examples among classes, one new example.
Output: STM, LTM, updated ranges of values for numerical attributes, updated frequency distribution of examples among classes.

Step 1. Update the frequency distribution of examples among classes.
Step 2. Test, whether the numerical attributes are within their existing ranges and if not update the ranges.
Step 3. Test whether there are rules in the LTM that classify or misclassify the new example and simultaneously update their accuracy measures (A measures) and H measures.
Step 4. Prune the LTM by removing the rules for which the A measure is lower than a given prespecified level (threshold).
Step 5. IF the number of examples in the STM is less than a prespecified limit
THEN add the example to the STM
ELSE
IF there are no rules in the LTM that classify the new example
THEN replace an example from the STM that belongs to the class with the largest number of presentatives in the STM by the new example.
Step 6. For each example in the STM that is not covered by the LTM, form a new rule by applying the rule forming procedure of Rules-3 Plus. Add the rule to the LTM. Repeat this step until there are no examples uncovered by the LTM.

STM is initially empty and contains a set of examined examples. LTM is used to have the set of previously extracted rules. It may also contain some initial rules defined by the user.

The noise threshold mentioned in Step 4 is defined as:

$$T = 2 - 2\sqrt{(1-NL)\frac{E_i}{E}} - 2\sqrt{NL\left(1-\frac{E_i}{E}\right)} \quad (6)$$

here NL is a prespecified noise level. When the accuracy measure A for a rule is less than the threshold T, the rule will be removed from the LTM.

3.6. Rules-5

RULES-5 was developed to overcome some of the deficiencies of RULES-3 Plus. It employs a method to handle continuous attributes. In this case there is no need to do quantization in order to process numerical attribute-values. Pham and Bigot explains the algorithm as follows [23].

The condition selection and continuous attribute handling procedure which is the core procedure used for RULES-5 is explained by the authors as follows: The rule formation procedure of RULES-5, takes account only the conditions excluding the closest example (CE) that does not belong to the target class and covered by the extracted rule so far. To obtain CE, a measure is used to find the distance between any two examples. The data set may contain continuous as well as discrete attributes, this measure is able to handle both types. The distance measure between example E1 and E2 is defined as follows:

$$Distance\_E1-E2 = \sqrt{\sum_c \left(\frac{V^i_{E1}-V^i_{E2}}{V^i_{\max}-V^i_{\min}}\right)^2 + \sum_d d\_distance} \quad (7)$$

where $\sum_c$ is the sum for continuous attributes, $\sum_d$ is the sum for discrete attributes. V^i_{E1} is the value of the i^{th} attribute in the example E1, V^i_{E2} is the value of the ith attribute in example E2, $V^i_{\max}$ is the maximum known value of the ith continuous attribute, $V^i_{\min}$ is the minimum known value of the i^{th} continuous attribute and d_distance is defined for each discrete attribute by applying the following rule:

If $V^i_{E1} = V^i_{E2}$ Then d_distance =0
Else d_distance=1

This will reduce the search space because not all conditions need to be tested.

Compared to its predecessors RULES-5 generates fewer rules, takes shorter training time and extracts more accurate rule sets.

4. APPLICATIONS OF INDUCTIVE LEARNING

Inductive learning algorithms are domain independent. In principle, they can be used in any task involving classification or pattern recognition [1]. There have been several successful applications of inductive learning systems. Medical applications such as lymphography, prognosis of breast cancer recurrence, location of primary tumour and thyroid problem diagnosis have been reported [5,24,25]. Other applications include

investment appraisal [26], forensic classification of glass fragment evidence [27], extraction of decision rules for analysis of test data for the space shuttle main engine [28], experimental generation of decision rules for the conceptual design of steel members under bending [10], soil classification [29], stock control [30], software resource analysis [31], assessing credit card applications [32], military decision making [33], dynamic system identification [34,35], engine fault diagnosis [36] and identification of the mass-spectra of complex materials [37,38,39].

RULES family of algorithms particularly, have many applications. Some of them are, Industrial Visual Inspection, Banknote Recognition, Signature Verification, Number plate Recognition, Inspection of ceramic tiles, Barcode Recognition, Dynamic System Identification,-The classification of well-known IRIS data, Parameters estimation in wastewater treatment and Application of RULES-3 to DNA Sequence. The details can be found in [40].

5. CONCLUSION

The induction of decision trees and rules from empirical data is a useful technique for automatic knowledge acquisition. It offers a modularized, clearly explained format for decision making which is compatible with human reasoning procedures. Also, the resulting rules are suitable for use in expert systems. Also it is reported that induction algorithms have been incorporated into a number of commercial systems including Expert-Ease, RuleMaster and ACLS [41].

In this paper the RULES family of algorithms is reviewed. The family so far has six versions. Each version has some extra new features to overcome some problems that cannot be coped with using previous versions. The algorithms have been used for many application which shows their good performance. However, there are still some features that the family should have in order to produce better set of rules. For example, the family can produce only discrete rules. It would be better if it can produce fuzzy rules. Especially for problems having numerical values, it will improve the efficiency of the algorithm.

6. REFERENCES

1. W.Z. Liu and A.P. White, A review of inductive learning, in proc. *Research and Development in Expert Systems VIII.*, Cambridge, 112-126, 1991.
2. A. Mrozek, A new method for discovering rules from examples in expert systems, *Int. J. Man-Machine Studies, 36*, 127-143, 1992
3. A. Hart, *Knowledge acquisition for expert systems*, Chapman and Hall, London, 1989.
4. D.A. Waterman, *A guide to expert systems*, Addision-Wesley, California, 1986.
5. J.R. Quinlan, Induction, knowledge and expert systems, in *Artificial Intelligence Developments and Applications*, Eds J.S. Gero and R. Stanton, Amsterdam, North-Holland, 253-271, 1988.
6. S.M. Weiss and C.A. Kulikowski, *Computer systems that learn*, Morgan Kaufmann, San Mateo, California, 1991.

7. G.J. Williams, Combining decision trees, initial results from MIL algorithm, in *Artificial Intelligence Developments and Applications,* Eds: J.S. Gero and R. Stanton, 273-289, 1988.
8. V. Devedzic and D. Velasevic, Features of second generation expert systems, an extended overview, *Eng. Appl. of AI,* 3, 255-270, 1990.
9. Y. Nakakuki Y., Y. Koseki and M. Tanaka, Inductive learning in probabilistic domain, in proc. *Eighth National Conf. on AI*, Boston, July 29, August 3, 809-814, 1990.
10. R. Forsyth, *Machine Learning principles and techniques*, Ed: R. Forsyth, Chapman and Hall, London, 1989.
11. P.J. Hancox, W.J. Mills and B.J. Reid, *Artificial intelligence / expert systems*, Ergosyst Associates, Lawrence, Kansas, 1990.
12. E. Charniak, and D. McDermott, *Introduction to artificial intelligence*, Addison-Wesley, California, 1985.
13. S.L. Tanimoto, *The elements of artificial intelligence*, Computer Science Press, Maryland, USA, 1987.
14. S.H. Rubin, Expert systems for knowledge acquisition, in proc. *First World Congress on Expert Systems , Vol 3*, Orlando, Florida, December 16-19, 1793-1799. 1991.
15. Y. Kodratoff, *Introduction to machine learning*, Pitman Publishing, London, 1988.
16. A. Al-Attar, Rule induction from mythology to methodology, *Research and Developments in Expert Systems VIII*, London, September, 85-103, 1991.
17. J.R. Quinlan, Generating production rules from decision trees, in proc. *Tenth IJCAI-87,* Milan, Italy, 304-307, 1987.
18. D.T. Pham and M.S. Aksoy, RULES: A simple rule extraction system, *Expert Systems with Applications*, 8, 59-65, 1995.
19. D.T. Pham and M.S. Aksoy, An algorithm for automatic rule induction, *Artificial Intelligence in Engineering,* 8, 277-282, 1993.
20. D.T. Pham and M.S. Aksoy, A new algorithm for inductive learning, *Journal of Systems Eng.,* 5,115-122, 1995
21. D.T. Pham and S.S. Dimov, An Efficient Algorithm For Automatic Knowledge Acquisition, *Pattern Recognition*, 30, 1137-1143, 996.
22. D.T. Pham and S.S. Dimov, An algorithm for incremental inductive learning, *Proc. Instn Mech. Engrs, Part B: J. Engineering Manufacture*, 211(B3), 239-249, 1997.
23. D.T. Pham & S. Bigot, Rules-5: A Rule Induction Algorithm for Classification Problems Involving Continuous Attributes, *Proc. Inst. Mechanical Engineers*, Part C, Vol. 217, 2003, 1273-1285, 2003.
24. R.S. Michalski et al., The multi-purpose incremental learning system AQ15 and its testing application to three medical domains, in proc. *National Conf. on AI*, Philadelphia, PA., August, 1041-1044, 1996.
25. J.R. Quinlan, Inductive knowledge acquisition: a case study, in *Applications of Expert Systems,* Ed: J.R. Quinlan, Turing Institute Press, 157-173, 1987.
26. P.R. Race and R.C. Thomas, Rule induction in investment appraisal, *Journal of the Operational Research Society, 38*, 1113-1123, 1988.
27. E.J. Spiehler, Application of machine learning to classification of glass fragment evidence in forensic science, *Seminar Materials, Machine Learning Seminar,*

Learned Information Ltd., Oxford, and Machine Learning Research Ltd., Nottingham, London, 1987.
28. K.L. Modesitt, Space shuttle main engine anomaly data and inductive knowledge-based systems: automated corporate expertise, in proc. *Conf. Artificial Intelligence for Space Applications*, Huntsville, Alabama, November, 1-8, 1987.
29. M.B. Dale, A.B. McBratney and J.S. Russell, On the role of expert systems and numerical taxonomy in soil classification, *Journal of Soil Science*, 40, 223-234, 1989.
30. J.C. Thorpe, A. Marr and R.S. Slack, Using an expert system to monitor an automatic stock control system, *Journal of the Operational Research Society, 40*, 945-952, 1989.
31. R.W. Selby and A.A. Porter, Learning from examples: generation and evaluation of decision trees for software resource analysis, *IEEE Transactions on Software Engineering, 14*, 1743-1756, 1988.
32. C. Carter and J. Catlett, Assessing credit card applications using machine learning, *IEEE Expert Intelligent Systems and Their Applications*, *FALL,* 71-79, 1987.
33. Y. Lirov, E.Y. Rodin and B.K. Ghosh, Automated learning by tactical decision systems in air combat, *Computer and Mathematics with Application, 18*, 151-160,1989.
34. C. Batur, A. Srinivasan and C.C. Chan, Automated rule-based model generation for uncertain complex dynamic systems, *Eng. Appl. of AI, 4(5),* 359-366, 1991.
35. M.S. Aksoy, Dynamic System Modeling Using Rules3 Induction Algorithm, *Mathematical & Computational Applications An International Journal,* 10, 121-132, 2005.
36. M. Ke and M. Ali, A learning representation and diagnostic methodology for engine fault diagnosis, in proc. *Second Int. Conf. on Industrial and Engineering Applications of AI and Expert Systems, 2*, June 6-9, 824-830, 1989.
37. P.D. Harrington, T.E. Street and K.J. Voorhees, Rule building expert systems for classification of mass spectra", *Analytical Chemistry, 61*, 715-719, 1989.
38. P.D. Harrington and K.J. Voorhees, Multivariate rule-building expert system, *Analytical Chemistry,* 62, 729-734, 1990.
39. D.R. Scott, Classification and identification of mass spectra of toxic compounds with an inductive rule building expert system and information theory, *Analytical Chimica Acta, 223*, 105-121, 1989.
40. M.S. Aksoy, Applications of RULES-3 Induction System, IPROMS2005 virtual conference, Cardiff, U.K., 2005.
41. X. Wu, Inductive learning: Algorithms and Frontiers, *Artificial Intelligence Review 7,* 93-108, 1993.