Mathematical and Computational Applications, Vol. 15, No. 3, pp. 318-324, 2010.

RULES3-EXT IMPROVEMENTS ON RULES-3 INDUCTION ALGORITHM

Hassan I. Mathkour
College of Computer and Information Sciences
Department of Computer Science
King Saud University
Riyadh 11653
Saudi Arabia
binmathkour@yahoo.com, mathkour@ksu.edu.sa

Abstract- This paper describes RULES3-EXT, a new algorithm for inductive learning. It has been developed to cope with some drawbacks of RULES-3 induction algorithm. The extra features of RULES3-EXT are (1) The number of required files to extract a knowledge base (a set of rules) is reduced to 2 from 3 (2) The repeated examples are eliminated, (3) The users are able to change the order of attributes and (4) The system is able to fire rule(s) partially if any of the extracted rules cannot fully be satisfied by an unseen example. The new algorithm has been tested on well known data sets and the efficiency found to be superior to that of RULES-3.

Key Words- Inductive Learning, Induction, Knowledge Acquisition, Machine Learning.

1. INTRODUCTION

Knowledge-based expert systems contain two main components: a knowledge base and an inference engine. Collecting knowledge to form the knowledge base is the basic task to build an expert system [1,2,3]. This is known as knowledge acquisition or extraction. The knowledge acquisition through interaction with an expert consists of a prolonged series of systematic interviews, usually extending over a long period [4]. Knowledge acquisition (and in particular machine learning) has been a major area of concern for expert systems research [5,6,7].

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 the expert 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 attainable. Indeed, the main feature of the second generation expert systems is that the knowledge acquisition process is highly automated [8].

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*

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 [10,11]. Many expert system shells support production rules. This has lead to various expert systems whose knowledge bases are formed from rules. In addition, rule-based expert systems are more natural in that they parallel the human way of expressing expertise. Such systems are also more popular with their relatively straight forward inference engines.

In this paper, we describe an extension to RULES-3 which makes it possible to produce a set of rules from a given set of examples. We demonstrate the superiority of the extended algorithm via test cases. The remaining of the paper is organized as follows: Section 2 reviews RULES-3 algorithm. Section 3 describes the features introduced to enhance the performance of RULES-3 giving rise to a new algorithm named RULES3-EXT. Section 4 describes the test cases used to demonstrate the performance of the new algorithm and discusses how it compares to RULES-3. Section 4 concludes the paper.

2. RULES-3 ALGORITHM

RULES-3 is an efficient algorithm for extracting a set of classification rules from a set of examples. Each object in the examples belongs to one of a number of predefined classes. An object has a fixed set of attributes, each with its own range of possible values which could be numerical or nominal. An attribute-value pair constitutes a condition in a rule. If the number of attributes is N_a , a rule may contain between one and N_a conditions. Only conjunction of conditions is permitted in a rule and therefore the attributes must all be different if the rule contains more than one condition.

During extraction process, RULES-3 considers one example at a time. It forms an array consisting of all attribute-value pairs associated with the object in that example, the total number of elements in the array being equal to the number of attributes of the object. The rule forming procedure may require at most N_a iterations per example. In the first iteration, rules may be produced with one element from the array. Each element is examined in turn to see if, for the complete set of examples, it appears only in objects belonging to one class. If so, a candidate rule is obtained with that element as the condition. In either case, the next element is taken and the examination repeated until all elements in the array have been considered. At this stage, if no rules have been formed, the second iteration begins with two elements of the array being examined at a time. Rules formed in the second iteration therefore have two conditions. The procedure continues until an iteration when one or more candidate rules can be extracted or the maximum number of iterations for the example is reached. In the latter case, the example itself is adopted as the rule. If more than one candidate rule is formed for an

example, the rule that classifies the highest number of examples, is selected and used to classify objects in the collection of examples. Examples of which objects are classified by the selected rule are removed from the collection. The next example remaining in the collection is then taken and rule extraction is carried out for that example. This procedure continues until there are no examples left in the collection and all objects have been classified. Figure 1 summarizes the steps involved in RULES-3 [12].

Step 1. Define ranges for the attributes which have numerical values and assign labels to those ranges
Step 2. Set the minimum number of conditions (N_{cmin}) for each rule
Step 3. Take an unclassified example
Step 4. $N_c = N_{cmin} - 1$
Step 5. If $N_c < N_a$ then $N_c = N_c + 1$
Step 6. Take all values or labels contained in the example
Step 7. Form objects which are combinations of N_c values or labels taken
from the values or labels obtained in Step 6
Step 8. If at least one of the objects belongs to a unique class then form rules with those objects;
ELSE go to Step 5
Step 9. Select the rule which classifies the highest number of examples
Step 10. Remove examples classified by the selected rule
Step 11. If there are no more unclassified examples then STOP;
ELSE go to Step 3

Figure 1. Induction procedure in RULES-3 (N_c=number of conditions, N_a=number of attributes) [12]

3. RULES3-EXT ALGORITHM

The main features of RULES-3 are: Short Training Time, The ability to Classify Unseen Examples, Specifiable Maximum Number of Rules, Ability to Deal with Incomplete Examples, Ability to Handle Attributes with Numerical Values, Ability to generate a compact set of more general rules and the option of adjusting the precision of the rules to be extracted.

RULES3-EXT was designed to have the following extra features to improve the performance on RULES-3: (1) The number of required files to extract a knowledge base (a set of rules) is reduced to 2 from 3 (2) The repeated examples are eliminated, (3) The users are able to change the order of attributes (4) The system is able to fire rule(s) partially if any of the extracted rules cannot fully be satisfied by an unseen example.

3.1. The Number of Required Files

RULES-3 requires three files: one for attributes names, one for classes names and one for the set of examples.. In the set of examples, each example contains a value of each attribute and a class name. As this file already contains the classes names,

RULES3-EXT does not need the 'classes file' because it can be formed using the set of examples. This feature will help to prevent any mismatch between the classes names given in a separate file and in the file having the set of examples. This in turn contributes to a higher degree of consistency.

3.2. The Elimination of Repeated Examples

The set of examples may contain some repeated examples. It may be coming from an automatic data generated or the user may by mistake re-input some examples. RULES-3 EXT eliminates all repeated examples before it starts the extraction process. This will help reduce the time needed to complete the extraction process. This contributes towards a more efficient (less costly) knowledge acquisition process.

3.3. Changing The Order of Attributes

RULES-3, forms rules by considering the attribute-values by considering the order they have been presented. Sometimes the user, for various reasons, may prefer some attributes to be considered before some others. This will result having preferred attributes first in the extracted rules. For example, for some rules, if the number of conditions is less than the number of attributes, these rules will have the preferred attributes. To do this, the order of attributes must be changed in the set of examples. RULES3-EXT, allows the users to change the order of attributes.

3.4. Partial Rule Firing

RULES-3 is a system that can extract a set of rules as well as use them for classifying trained and unseen examples. When an unseen example is given to the system, if there is no rule that could be satisfied by the example, the result is unknown. To avoid these, RULES3-EXT adopts a strategy that helps lessen the number of unknown results. It allows the rules to be partially satisfied. For example, if there are at most three attributes in any rule in the rule set, when three conditions are not satisfied, the system will search for two conditions to be satisfied. If the two conditions cannot be satisfied, then a search is made for one condition. If the result is given by such a rule, it will be marked as partially satisfied. If the example cannot be classified the result will be unknown.

4. TEST CASES

To demonstrate the superiority of RULES3-EXT to RULES-3, two test cases have been analyzed and the performance of the two algorithms is compared. The first case is for credit data and the second is for flowers data. The performance of the two algorithm is given below.

4.1. Test Case for Credit data

This data is obtained from UCI Machine Learning Repository [13]. UCI repository is an international machine learning database repository. It is an archive of many databases used specifically for evaluating data mining and machine learning algorithms. This repository is maintained by University of California, Irvin.

This file concerns credit card applications. It contains 15 attributes. It has two classes. All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data. This dataset is interesting because there is a good mix of attributes - continuous, nominal with small numbers of values, and nominal with larger numbers of values. There are also a few missing values. 129 examples were randomly chosen for training and 140 for test. The training parameters are given in Table 1.

Table 1. The training parameters set by the user for RULES-3 and RULES3-EXT.

	No. of quantization level for numerical attributes	No. of rules to be extracted	Min. No. of conditions for each rule
RULES-3	3	Minimum	1
RULES3-EXT	3	Minimum	1

Table 2 lists the results from both algorithms. In Table 2 it is clear that of RULES3-EXT outperforms RULES-3 for the same test (unseen) examples. Although all features contributed towards the better performance of RULES3-EXT in this case, a main contribution was due to the partial rule firing feature of RULES3-EXT.

Table 2. The number of training, test, correctly classified and misclassified examples for RULES-3 and RULES3-EXT.

	No. of training Examples	No.of test Examples	No. of correctly Classified Examples	No. of Misclassified Examples	Efficiency %
RULES-3	129	140	102	38	73
RULES3-EXT	129	140	113	27	81

4.1. Test for IRIS data

This data [13,14] is about the classification of flowers. It contains 4 attributes namely: petal length, petal width, sepal length and sepal width. It has three classes: Iris-setosa, Iris-virginica and Iris versicolor. 50 examples were randomly chosen for training and 100 for test. Table 3 shows the initial parameters set by the user for training both RULES-3 and RULES3-EXT.

Table 3. The training parameters set by the user for RULES-3 and RULES3-EXT.

	No. of quantization level for numerical attributes	No. of rules to be extracted	Min. No. of conditions for each rule
RULES-3	7	Maximum	2
RULES3-EXT	7	Maximum	2

Table 4 shows that, the efficiency of RULES3-EXT is obviously better than of RULES-3 for the same test (unseen) examples. This is also because of the partially rule firing feature of RULES3-EXT.

Table 4. The number of training, test, correctly classified and misclassified examples for RULES-3 and RULES3-EXT.

	No. of training Examples	No.of test Examples	No. of correctly Classified Examples	No. of Misclassified Examples	Efficiency %
RULES-3	50	100	88	12	88%
RULES3-EXT	50	100	94	6	94%

5. CONCLUSION

In this paper, an extended version of RULES-3 algorithm is introduced. The algorithm, RULES3-EXT, has been developed to overcome some drawbacks of RULES-3. The extra features of RULES3-EXT are (1) The number of required files to extract a knowledge base (a set of rules) is reduced to 2 from 3 (2) The repeated examples are eliminated, (3) The users are able to change the order of attributes and (4) The system is able to fire rule(s) partially if any of the extracted rules cannot fully be satisfied by an unseen example. This will make the algorithm to be more efficient and more general as it can classify more number of unclassified examples. The training time is shorter. The new algorithm has been tested on well known IRIS data and Credit Card Application data sets and the efficiency found to be better then of RULES-3.

Acknowledgement-The author would like to thank Research Center in the College of Computer and Information Sciences, King Saud University for its financial support to complete this study.

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. A. Al-Attar, Rule induction from mythology to methodology, *Research and Developments in Expert Systems VIII*, London, September, 85-103, 1991.
11. J.R. Quinlan, Generating production rules from decision trees, in proc. *Tenth IJCAI-87,* Milan, Italy, 304-307, 1987.
12. D.T. Pham and M.S. Aksoy, A new algorithm for inductive learning, *Journal of Systems Eng.,* 5,115-122, 1999.
13. A. Asuncion, and D.J. Newman, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science, 2007.
14. R.A. Fisher, The use of multiple measurements in taxonomic problems, *Ann. of Eugenics*, 7, 179-188, pp. 466-475, 1936.