An Evolutionary Ensemble-Based Method for Rule Extraction with Distributed Data*

Diego M. Escalante, Miguel Angel Rodriguez, and Antonio Peregrin

Dept. of Information Technologies
University of Huelva
{diego.escalante,miguel.rodriguez,peregrin}@dti.uhu.es

Abstract. This paper presents a methodology for knowledge discovery from inherently distributed data without moving it from its original location, completely or partially, to other locations for legal or competition issues. It is based on a novel technique that performs in two stages: first, discovering the knowledge locally and second, merging the distributed knowledge acquired in every location in a common privacy aware maximizing the global accuracy by using evolutionary models. The knowledge obtained in this way improves the one achieved in the local stores, thus it is of interest for the concerned organizations.

1 Introduction

Information technologies peak has produced a huge amount of data, and mining them is one of the most successful areas of research in computer science.

Every now and then, data may be geographically distributed, and habitual data mining techniques need to centralize it, or to get benefits from the distributed computing, using distributed algorithms that move knowledge and training data [13] and [8].

Nevertheless, it is not always possible to move the data in a distributed system because competition or legal issues. For example, banking entities may be interested in global knowledge benefits to avoid credit card fraud, but they have to safeguard their clients data. Another example concerns the medical field, where global knowledge for diagnosis or research studies is desired considering that some pathologies may be different depending on geographical information, but the privacy of patients data must be guaranteed due to legal reasons. Also, in other cases it is not possible to merge all the data in a single system due to computational resources limitations.

On the other hand, model combination is the core idea behind classical machine learning methods such as Bagging [4], Boosting [5] or Stacking [6]. Models can be seen as experts and classificacion may be better if several experts opinions are combined. As far as we know, all these methods were designed to work in non

* This work has been supported by the Spanish Ministry of Innovation and Science under grant No. TIN2008-06681-C06-06, and the Andalusian government under grant No. P07-TIC-03179.

E. Corchado et al. (Eds.): HAIS 2009, LNAI 5572, pp. 638–645, 2009.

distributed environments. They have got the whole dataset at the beginning of the classification process, so they are not directly applicable to solve inherently distributed data problems.

Therefore, merging distributed knowledge without moving the data is an interesting research area. This paper proposes a novel method to learn from inherently distributed data without sending any of them from one place to another. It is based on making the classifiers locally, and then ensembling them in a single final one using an evolutionary methodology where the population of candidate classifiers is concurrently evaluated in a distributed way in each local node.

This document is organized as follows. Section 2 provides the concepts behind the solution presented. Section 3 focuses the proposed method describing all its components. Section 4 shows the experimental study developed and finally, we present some concluding remarks in Section 5.

2 Preliminaries

This section describes the theoretical concepts in which the proposed method is based on. First, an introduction to metalearning techniques is shown and finally genetic algorithms (GA) are introduced as a learning tool.

2.1 Metalearning

Metalearning [1] is a strategy that makes easier independent models combination and supports the data mining applications big scalability, so in some cases we can avoid the data movement issue by combining models instead of raw data. Two main policies are related in the literature [3] to perform model combination:

- Multiple Communication Round: Methods of this kind require a significant synchronization amount. They usually use a voting system, so sending examples throught the system is often necessary.
- Centralized Ensemble-based: This kind of algorithms can work generating the local classifiers first and combining them at a central site later. This is the one we use in our proposal.

2.2 Genetic Learning

GAs have achieved reputation of robustness in rule induction in common problems associated to real world mining (noise, outliers, incomplete data, etc). Initially, GA were not designed as machine learning algorithms but they can be easily dedicated to this task [9]. Typically the search space is seen as the entire possible hypothesis rule base that covers the data. The goodness can be related to a coverage function over a number of learning examples [8][13].

Regarding the representation of the solutions, the proposals in the specialized literature usually use two approaches in order to encode rules within a population of individuals:

- The "Chromosome = Set of rules", also called the Pittsburgh approach, in which each individual represents a rule set [10]. In this case, a chromosome evolves a complete rule set and they compete among them along the evolutionary process.
- The "Chromosome = Rule" approach, in which each individual codifies a single rule, and the whole rule set is provided by combining several individuals in a population (rule cooperation) or via different evolutionary runs (rule competition). In turn, within the "Chromosome = Rule" approach, there are three generic proposals:
  - The Michigan approach, in which each individual encodes a single rule. These kinds of systems are usually called learning classifier systems [11]. They are rule-based, message-passing systems that employ reinforcement learning and a GA to learn rules that guide their performance in a given environment. The GA is used for detecting new rules that replace the bad ones via the competition between the chromosomes in the evolutionary process.
  - The IRL (Iterative Rule Learning) approach, in which each chromosome represents a rule. Chromosomes compete in every GA run, choosing the best rule per run. The global solution is formed by the best rules obtained when the algorithm is run multiple times. SIA [12] is a proposal that follows this approach.
  - The GCCL (Genetic Cooperative-Competitive Learning) approach, in which the complete population or a subset of it encodes the rule base, In this model the chromosomes compete and cooperate simultaneously, [14] is an example of this approach.

3 Proposed Method

In this work we present an Evolutionary eNsemble-based method for Rule Extraction with Distributed Data (ENREDD). The ensemble-based process shares only the local models being a reasonable solution to privacy constraints. Also, it uses low bandwidth due to the low amount of data transmited (classifiers are sent instead of raw data).

Centralized ensemble-based metalearning processes are usually divided in two stages:

- Creating local classifiers from the distributed datasets.
- Aggregate local knowledge in a central node.

The algorithm resolves the first stage using a GA based on a GCCL approach. When local models are generated, they are sent to a central node where the second stage starts. Subsection 3.1 details the local learning system.

The central or master node uses an evolutionary algorithm to merge the rules from the local models. Because data stays in each local node, the algorithm must complete the task without moving the data, so it sends the candidate classifiers to the distributed nodes to evaluate their quality. Each local node sends the

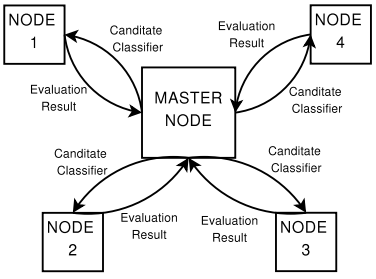

Fig. 1. Communications for candidate classifiers evaluation

accuracy obtained with its data. Once the master node has got the results, it averages the values to get a global measure of classifier quality (Fig. 1).

Master node uses a Pittsburgh approach to merge the rules from the local classifiers. A detailed description of this stage will be shown in Subsection 3.2.

3.1 Local Nodes

As was commented before, local nodes must build a classifier from the data they have got. The idea behind this method is that an acceptable rule in a local set is a candidate to compose the global classifier [2], so all the rules are at the central node. This method reduces the amount of communication and lets build an independent model generation.

ENREDD local learning process generates an initial population with a heuristic function based on local data. The chromosome has a binary representation and each gene represents a possible value for a given attribute that will be active if the value contributes to the rule.

In the example chromosome of Figure 2, the binary coding represents the rule *if* c_1 *in* (v_1,v_3) *and* c_3 *in* (v_6) *then class is* v_{10}.

c_1			c_2		c_3			Class	
V_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}
1	0	1	0	0	1	0	0	0	1

Fig. 2. Local node chromosome representation

The evaluation function balances simplicity and quality with Equation 1

$$f(r) = \left(1 + \frac{zeros(r)}{length(r)}\right)^{-1*Cases-} \tag{1}$$

where Cases- is the number of covered examples predicted as false positives, Zeros(r) is the number of zeroes in the bit string representation of the rule r and length is the chromosome length expressed in bits. In order to force the winning rules to be as accurate as possible, the fitness is exponentially measured and just when Cases- is really low the length is taken into account.

The selection is implemented with the universal suffrage operator[13] selecting the individuals that take part in recombination mixing a vote process among the train examples that takes into account the number of positive covered cases. After the selection, two crossover operators are randomly applied, two point crossover and uniform crossover. The offsprings will replace a randomly selected individual in the original population without keeping any elite population.

3.2 Master Node

The master node collects all the local classifiers received, and then, it starts the genetic optimization process that includes two phases that are described next.

Rule merging. It solves the distributed rules aggregation. The GA task is to sort the rule list to get the best order possible.

The local classifiers rules are inserted together in a table without repetition, and an integer index is asigned to each rule. Thus an integer representation has been chosen for the GA chromosome. The order inside the chromosome will determine the rule application order.

The algorithm chosen is a CHC [7] based model. The initial population is generated randomly. The Hamming distance has been considered using the number of differing integer genes in the chromosome, so once it is calculated, the half differing genes can be swapped. The parents are only crossed when Hamming distance exceeds a threshold d.

For each chromosome, the evaluation function applies the classifier in each node to evaluate it with all the available data. Next, each node sends to the master node the accuracy percent obtained, and it averages the global quality of each chromosome.

Rule reduction. It deletes the rules that never get fired with the distributed data.

It sends the final classifier to the nodes and all the rules activated with the examples they have got are marked. Then, the classifier is returned to the master node and it deletes all the rules that have not been marked.

4 Experimental Study

This section describes the experimental study developed to test the proposed method and analyzes the results obtained.

In order to compare the behaviour of the presented method we propose to modify the well known Bagging [4]. It works selecting different samples of a single dataset named bags. A classifier for each bag is created and the final classifier output is the most voted in the samples classifiers. In order to apply Bagging to distributed data, we consider bags as distributed nodes, so we use T/N samples for each node, being T the dataset size and N the number of nodes. We name this modified version MBAG.

The aim of this preliminary study is to validate the proposal without real world complexities due to data distribution like unbalanced data, heterogeneous

domain, local discretization, etc. To achieve this target the main dataset has been discretized using 10 fixed frequency values gaps with a 10 fold cross validation in a 70/30 training/test proportion.

Simulations with 5, 10 and 20 nodes have been performed. These values have been selected because higher values result in a few training examples and lower ones are no representative of a distributed configuration. For local nodes, the GA uses 250 individuals and 200 generations. The master node uses 50 individuals and 600 generations.

To compare ENREDD with MBAG we have used the Wilcoxon Signed-Ranks Test(WSRT) [15]. It is a non-parametric alternative to the paired t-test, which ranks the differences in performances of two classifiers for each dataset, ignoring the signs, and compares the ranks for the positive and the negative differences.

WSRT can reject the null hypothesys[16] (equal accuracy for compared algorithms in our study) with $\alpha = 0.05$ when parameter z is smaller than -1.96.

4.1 Results Analysis

Table 1 shows the training and test sets accuracy means for both methods. The BN columns are the trainining and test accuracy percentages for MBAG simulations with N nodes and the EN ones are the results for ENREDD.

Table 1. Average test accuracy

	TRAINING						TEST					
	B5	E5	B10	E10	B20	E20	B5	E5	B10	E10	B20	E20
Car	80.57	**97.22**	76.19	**94.49**	71.45	**92.78**	78.29	**93.97**	74.59	**90.91**	70.83	**89.89**
Cleveland	58.60	**82.83**	56.09	**79.35**	53.77	**77.92**	**52.78**	52.61	**54.00**	53.37	**54.44**	53.43
Credit	86.19	**91.72**	85.42	**94.67**	85.90	**89.94**	**85.60**	83.91	85.65	**86.38**	**86.28**	86.09
Ecoli	70.55	**89.72**	58.51	**85.25**	45.11	**79.66**	64.36	**66.84**	54.36	**63.05**	43.27	**60.36**
Glass	59.26	**85.43**	49.53	**79.85**	38.39	**87.70**	47.69	**50.54**	39.85	**44.51**	31.23	**38.23**
Haberman	74.58	**85.84**	73.93	**83.56**	73.64	**82.87**	**71.85**	68.52	**72.93**	70.35	**73.26**	69.22
House-votes	64.53	**99.19**	63.40	**98.92**	62.04	**98.30**	62.68	**96.80**	62.72	**97.12**	62.30	**96.00**
Iris	68.46	**96.11**	39.62	**98.06**	0	0	63.91	**76.09**	36.96	**68.09**	0	0
Krvskp	97.02	**98.74**	96.00	**98.50**	94.74	**98.29**	**96.94**	97.76	95.95	**97.72**	94.93	**97.57**
Monk	54.93	**84.72**	52.55	**82.07**	51.82	**75.92**	46.92	**65.50**	47.62	**63.98**	45.23	**64.27**
Mushroom	99.90	**100.0**	99.82	**99.99**	99.54	**100.0**	99.90	**100.0**	99.82	**99.95**	99.47	**99.99**
New-thyroid	88.60	**98.59**	77.13	**97.89**	70.20	**97.97**	85.69	**90.68**	74.92	**88.54**	68.77	**79.58**
Nursery	94.20	**97.99**	92.19	**98.08**	90.80	**96.96**	93.47	**96.12**	92.09	**96.53**	90.72	**96.40**
Pima	74.99	**86.59**	74.56	**83.39**	70.32	**81.28**	72.42	**79.87**	**72.03**	70.53	68.83	**71.41**
Segment	90.36	**96.23**	88.43	**93.13**	**85.67**	85.31	88.02	**92.19**	86.72	**89.04**	**83.8**	79.92
Soybean	91.66	**96.91**	88.25	**92.01**	77.31	**85.77**	91.15	**92.83**	**87.15**	87.10	75.05	**80.12**
Splice	94.22	**99.24**	93.84	**98.24**	92.75	**96.77**	93.32	**96.25**	93.25	**94.54**	91.79	**95.16**
Tic-tac-toe	88.88	**97.69**	74.13	**92.61**	70.42	**89.61**	85.35	**94.46**	70.94	**87.37**	69.03	**84.76**
Vehicle	66.44	**78.16**	63.61	**72.24**	60.57	**67.13**	55.28	**57.21**	**54.53**	53.61	**54.57**	50.25
Vote	95.59	**98.34**	95.63	**97.93**	95.39	**97.65**	**95.73**	95.16	**95.50**	94.93	**95.50**	94.00
Waveform	**79.08**	78.74	**80.65**	78.63	**80.76**	76.68	**75.88**	66.45	**78.18**	72.04	**78.93**	72.37
Wine	72.66	**96.56**	64.68	**96.35**	45.56	**100.0**	**61.85**	61.35	**57.59**	52.97	**39.63**	34.62
Wisconsin	93.26	**99.01**	89.41	**98.46**	84.96	**98.22**	91.02	**95.12**	88.39	**95.54**	84.83	**94.67**
Zoo	93.26	**99.85**	89.41	**99.68**	84.96	**99.08**	**91.02**	81.40	**88.39**	74.78	**84.83**	72.15

Table 2. Wilcoxon Signed Ranks Test

	E5 > B5	E10 > B10	E20 > B20
Negative Ranks	7	9	7
Positive Ranks	17	15	16
Ties	0	0	1
z	-2.743	-2.371	-2.403
p-value	0.006	0.018	0.016

The accuracy of both methods is lower than using standard classifing methods due to the impact of splitting datasets. Sometimes the distributed behaviour may be affected by the number of local datasets due to the fact that there is not enough representation of each class in a local node. Maybe some datasets show poor accuracy with both methods due to this fact. In the other datasets the distribution does not show any tendency regarding accuracy in ENREDD and shows a better accuracy than MBAG.

Table 2 shows the WSRT statistical test results obtained from the Table 1 data. It is shown that z is lower than -1.96 for all the cases, so we can reject the null hypothesis with $\alpha = 0.05$. For example, for a $E5 > B5$ accuracy, null hypothesis is rejected with a 99.4% confidence, because *p-value* is 0.006.

5 Conclusions and Future Work

A methodology for knowledge discovery from distributed data without moving it from its original location for legal or competition issues have been proposed. It generates local distributed classifiers using an evolutionary model, and after, it merges them using an additional evolutionary algorithm evaluating the candidate solutions with the distributed data sets in a distributed parallelized way. The knowledge discovered with this method may be of significance for some organizations interested in to collaborate to get common knowledge for some areas like security, frauds and so on.

As future work, we plan to get better the rule reduction mechanism in order to improve the interpretability of the models obtained and also, we will create synthesized data sets specifically created to simulate the geographically distributed data without the drawbacks of the generic datasets used in this work.

References

1. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning. In: Applications to Data Mining. Springer, Heidelberg (2009)
2. Provost, F., Hennessy, D.: Scaling up: Distributed machine learning with cooperation. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, pp. 74–79. AAAI Press, Menlo Park (1996)
3. Da Silva, J., Giannella, C., Bhargava, R., Kargupta, H., Klush, M.: Distributed Data Mining and Agents. Engineering Applications of Artificial Intelligence 18, 791–807 (2005)

4. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
5. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156. Morgan Kaufmann, San Francisco (1996)
6. Wolper, D.H.: Stacked generalization. Neural Networks 5, 241–259 (1992)
7. Eshelman, L.J.: The CHC Adaptative Search Algorithm: how to have safe search when engaging in nontraditional genetic recombination Foundations of Genetic Algorithms I, pp. 265–283. Morgan Kaufmann Publishers, San Mateo (1991)
8. Peregrin, A., Rodriguez, M.A.: Efficient Distributed Genetic Algorithm for Rule Extraction Eighth International Conference on Hybrid Intelligent Systems, pp. 531–536 (2008)
9. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, New York (1989)
10. Smith, S.: A learning system based on genetic algorithms. PhD Thesis, University of Pittsburgh (1980)
11. Holland, J.H., Reitman, J.S.: Cognition Systems Based on Adaptive Algorithms. In: Waterman, D.A., Hayes-Roth, F. (eds.) Pattern-Directed Inference Systems, Academic Press, New York (1978)
12. Venturini, G.: SIA: a supervised inductive algorithm with genetic search for learning attribute based concepts. In: Proceedings of European conference on machine learning, Vienna, pp. 280–296 (1993)
13. Giordana, A., Neri, F.: Search-intensive concept induction. Evolutionary Computation, 375–416 (1995)
14. Greene, D.P., Smith, S.F.: Competition-based induction of decision models from examples. Machine Learning 12(23), 229–257 (1993)
15. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics 1, 80–83 (1945)
16. Zar, J.H.: Biostatistical Analysis. Prentice Hall, Englewood Cliffs (1999)
17. Merz, C.J., Murphy, P.M.: UCI repository of machine learning databases. University of Carolina Irvine, Department of Information and Computer Science (1996), `http://kdd.ics.uci.edu`