GATree: Genetically Evolved Decision Trees

Athanassios Papagelis, Dimitrios Kalles
Computer Technology Institute
PO Box 1122, 261 10, Patras, Greece
{kalles,papagel}@cti.gr

Abstract

We use genetic algorithms to evolve classification decision trees. The performance of the system is measured on a set of standard discretized concept learning problems and compared (very favorably) with the performance of two known algorithms (C4.5, OneR).

1. Introduction

Genetic Algorithms (GAs) have been widely used as an effective search technique. Rather than search from general-to-specific or from simple-to-complex hypotheses, GAs generate successor hypotheses by repeatedly mutating and recombining parts of the best currently known hypotheses.

Decision tree induction has been successfully applied to a broad range of tasks from learning to diagnose medical cases to learning to assess credit risk of loan applicants. The construction of optimal decision trees has been proven to be NP-complete [1]. This led to the development of several heuristics. Current inductive learning algorithms use variants of impurity functions like information gain, gain ratio [2], gini-index [3], distance measure [4], *j*-measure [5] to guide the search. Fayyad [6] discusses several deficiencies of impurity measures, pointing out that they are insensitive to inter-class separation and intra-class fragmentation, as well as insensitive to permutations of the class probability distribution.

This work attempts to overcome the use of greedy heuristics and search the decision tree space in a more natural way. Specifically, we use GAs to directly evolve binary decision trees, without using binary string representations. We couple our objective with a simplification motivation. We use GAs to evolve *accurate* as well as *simple* decision trees.

Although GAs have been used extensively for classification and concept learning tasks [7-12], there is little work on their utility as a tool to evolve decision trees. The closest relative of this work comes from Koza [13], who speculates on the suitability of the tree genome for decision tree building. Most often GAs are related with Decision Trees (and other pattern classification algorithms) as a preprocessor for the problem of feature selection [14-17].

Since Schaffer [18] introduced the concept of different levels of suitability for *learner biases*, the idea that there is no universally better algorithm is fast maturing on the machine learning community. Here we focus on and distinguish between *preference* and *procedural* bias. A preference bias is based on the learner's behavior while a procedural bias is based on the learner's design. For example, C4.5 is biased towards accurate, small trees (preference bias) and uses the gain-ratio metric and minimum-error pruning (different procedural bias). A preference bias is most often desirable since it determines the characteristics of the produced tree. On the other hand, an inadequate procedural bias may severely affect the quality of the output. The proposed search imposes a new *weak* procedural bias, one that employs global metrics of tree quality. We thus shift from "how" to induce a tree (standard, impurity-based induction) to "what criteria an induced tree must satisfy".

There is an active debate on the machine learning community on whether less greedy heuristics can improve the quality of the produced trees. Garey and Graham showed that greedy algorithms using information theoretic splitting criteria can be made to perform arbitrarily worse than the optimal [19]. Norton showed that exhaustive lookahead applied to ID3 reduced tree sizes on average and produced small gains in accuracy, but could be expensive [20]. Ragavan and Rendell showed that their LFC algorithm that performed both lookahead and constructive induction can perform well on tasks involving feature interaction [21]. On the other hand, Murthy and Salzberg found that one-level lookahead yields larger, less accurate trees on many tasks [22]. Quinlan and Cameron-Jones reported similar findings and hypothesized that lookahead can yield "fluke theories" that fit the training data but have poor predictive accuracy [23].

The rest of this paper is organized in three sections. First, we we elaborate on the GATree system and the

1082-3409/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: University of Leeds. Downloaded on June 09,2020 at 14:10:14 UTC from IEEE Xplore. Restrictions apply.

modifications to the standard genetic operators. We then validate our concept via an experimental session and conclude by identifying promising lines of research.

2. The GATree system

Traditional binary strings representations do not appear well suited for representing symbolic concept descriptions of varying length and complexity. One can resolve this issue by changing the fundamental GA operators so as to work with the complex non-string objects, or by constructing string representations of solutions that minimize any changes to the basic GA philosophy.

We stuck with the first approach for three fundamental reasons. First, it is natural to use a tree structure to represent decision trees and the mutation-crossover operators can be efficiently altered to match this structure. Second, it is not trivial to alter the basic genetic operators to be used with string representatives of decision trees. Finally, emerging libraries of GA's components allow alternative representations and substantially decrease the overhead of tuning GA operators. For this work we have used GALIB [24], a library of GA components, to build a population of minimal binary decision trees.

We build decision trees that have one decision node that leads to two leaves. Every decision node has a random chosen value as a test. First we choose a random attribute. Then, if that attribute is nominal we randomly choose one of its possible values; if it is continuous we randomly pick an integer value belonging to its min-max range. This approach reduces the size of search space and it is straightforward. Still, it has problems with real-valued attributes; for this work we concentrated on nominal attributes. Leaves are populated using the same line of thought; we just pick a random class from the ones available.

Mutation chooses a random node and replaces that node's test-value with a new random value (Figure 1).

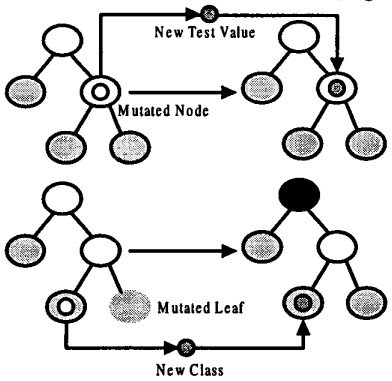

Figure 1. Mutation examples.

The crossover operator chooses two random nodes and swaps those nodes' sub-trees. Since predicted values rest only on leaves, the crossover operator does not affect the decision tree's coherence (Figure 2).

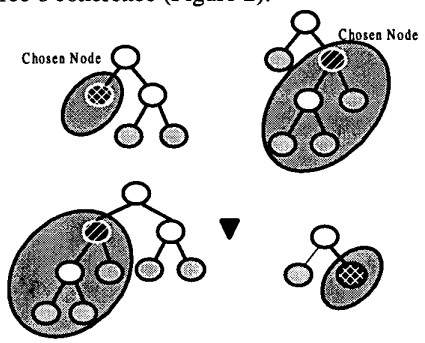

Figure 2. Crossover examples.

For the payoff (objective) function that assigns utility to candidate solutions, we balanced accuracy and size:

$$payoff\ (tree\ i) = CorrectClassified_i^{\ 2} * \frac{x}{size_i^{\ 2} + x} \qquad \textbf{(Eq. 1)}$$

The second part of the formula (the size factor) includes a factor x which has to be set to an arbitrary big number. This way, the payoff is greater for smaller trees. The size factor can be altered to match individual needs.

To reduce the overcrowding problem [8] we used a scaled payoff function, which aimed at reducing the similarity of decision trees on the population. When there were many decision trees with similar characteristics, we reduced their payoff function (similarity is estimated using a simple formula based only on the differences between the number of nodes and tree levels).

To speed up evolution we also implemented an altered version of Limited Error Fitness (LEF) [25]. This technique introduces an error limit. If the number of errors of an individual, during the process of evolution, is higher than the error limit, all remaining cases are treated as errors. This means that poor individuals will not be evaluated on the entire training set, saving CPU time. With moderate usage of the error limit we were able to produce insignificant accuracy loses and significant CPU time savings.

3. Experiments

WEKA (v 3.1.6) [26] implementations for C4.5 [27] and OneR [28] were used, with the default parameters supplied by WEKA. A 5-fold cross-validation was adopted. GA parameters are shown in Table 1.

Experiments were conducted using datasets from the UCI Repository [29]. Continuous attributes were

Authorized licensed use limited to: University of Leeds. Downloaded on June 09,2020 at 14:10:14 UTC from IEEE Xplore. Restrictions apply.

discretized using WEKA's unsupervised equal-frequency binning method, using the entropy minimization criterion. Table 2 and Table 3 present the results.

Table 1. Experiments parameters

Evolution Type	Generational
Initial Population	200
Generations	200
Generation Gap	25 %
Mutation Probability	0.005
Crossover Probability	0.93
x in Size Factor	1000
Random Seed	123456789

Table 2. Classification accuracy

	C4.5	OneR	GATree
Colic	83.84±3.41	81.37±5.36	85.01±4.55
Heart-Statlog	74.44±3.56	76.3±3.04	77.48±3.07
Diabetes	66.27±3.71	63,27±2.59	63,97±3.71
Credit	83.77±2.93	86.81±4.45	86.81±4
Hepatitis	77.42±6.84	84.52±6.2	80.46±5.39
Iris	92±2.98	94.67±3.8	93.8±4.02
Labor	85.26±7.98	72.73±14.37	87.27±7.24
Lymph	65.52±14.63	74.14±7.18	75.24±10.69
Breast-Cancer	71.93±5.11	68.17±7.93	71.03±8.34
Zoo	90±7.91	43.8±10.47	82.4±4.02
Vote	96.09±3.86	95.63±4.33	95.63±4.33
Glass	55.24±7.49	43.19±4.33	53.48±4.33
Balance-Scale	78.24±4.4	59.68±4.4	71.15±6.47
AVERAGES	78.46	72.64	78.75

Table 3. Average tree sizes

	C4.5 (pruned)	GATree
Colic	27.4	5.84
Heart-Statlog	39.4	8.28
Diabetes	140.6	6.6
Credit	57.8	3
Hepatitis	19.8	5.56
Iris	9.6	7.48
Labor	8.6	8.72
Lymph	28.2	7.96
Breast-Cancer	35.4	6.68
Zoo	17	10.12
Vote	11	3
Glass	60.2	8.98
Balance-Scale	106.6	8.92
AVERAGES	43.2	7.01

GATree was able to produce very accurate results, though the difference with C4.5 is not significant. However, it is most important that those results were accompanied by extremely small decision trees (C4.5 produced seven times bigger trees on average). Another significant point is that, even though there are datasets where the accuracy difference between C4.5 and OneR was big (Labor, Lymph, Zoo, Balance-Scale, Labor) GATree managed to be close to (or better than) the most accurate scheme.

OneR does exceed C4.5 in accuracy in several noisy datasets, but performs substantially worse on average. On the other hand, C4.5 produces unnecessarily big trees. Pruning consistently under-prunes the resulted trees. Contrary to greedy induction, GATree produces a dynamic, small-biased, accuracy/size based tree optimisation. This procedure is potentially superior than the (treated as uncorrelated) build-prune procedure of greedy heuristics. Nevertheless, GATree's "pruning capabilities" is just a side effect of its design. Possibly, there are better ways to achieve its effect using more precise global metrics of tree quality.

4. Discussion

GATree can be substantially improved by the dynamic tuning of parameters. One can estimate the problem's space-size and the convergence characteristics (by a bootstrap testing procedure). We intend to investigate the effect of those two parameters on initial algorithm characteristics to obtain optimal results with less generations and smaller initial population.

A basic drawback of GAs, compared with greedy heuristics, is speed. In order to evolve 500 decision trees for 500 generations with 25% generation gap we have to create and test 62875 decision trees. Although those trees are cheap to create and use, the time burden is substantially bigger than that of other heuristics (like information gain).

A possible solution is based on the fact that the *control problem* (a major issue when the knowledge is represented with rules) is implicitly solved in decision trees. The crossover/mutation operators change the tree from a node downwards. Instead of classifying every instance using the changed tree (in order to assign it some score), we can classify only the instances that belong to the changed-node's subtree. That can result in substantial timesavings when the crossover is near the tree's fringe. The extra burden is additional structures that keep track of every instance passing from some node, together with node statistics (how many instances pass from it, how many of them were correctly classified).

In this work we have explored how GAs can be used to directly evolve decision trees. The whole approach is based on conceptual simplicity, adopting only necessary extensions to basic GAs and small *a priori* bias. The experiments have indicated that GAs have substantial advantages over other greedy induction heuristics.

Authorized licensed use limited to: University of Leeds. Downloaded on June 09,2020 at 14:10:14 UTC from IEEE Xplore. Restrictions apply.

5. References

[1] Murthy S., K (1998). Automatic construction of decision trees from data: A multidisciplinary survey. *Data Mining and Knowledge Discovery.*

[2] Quinlan, R. (1986). *Induction of decision trees*, Machine Learning, 1:81-106,1986

[3] Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J. (1984) *Classification and Regression Trees.* Wadsworh International Group.

[4] de Mantaras., R.S. (1989). ID3 Revisited: A distance based criterion for attribute selection, Proceedings of Int. Symp. Methodologies for Intelligent Systems, Charlotte, North Carolina, USA.

[5] Smyth, P. Goodman, R.M. (1990) Rule Induction using information theory, *Knowledge Discovery in Databases,* MIT Press.

[6] Fayyad, M.U. (1991). *On the Induction of Decision Trees for Multiple Concept Learning*, Doctoral dissertation, Department of Electrical Engineering and Computer Science, Univercity of Michigan.

[7] Wilson, S.W. (1986). *Classifier system learning of a boolean function* (Research Memo RIS-27r). Cambridge, MA: Rowland Institute for Science.

[8] Goldberg D. (1989). *Genetic Algorithms in Search, Optimization & Machine Learning.* Addison-Wesley.

[9] Booker L.B., D.E. Goldberg & J.H. Holland (1989). Classifier Systems and Genetic Algorithms, *Artificial Intelligence*, 40, 2, 235-282.

[10] DeJong, K.A., Spears, W. M., & Gordon, D.F. (1993). Using genetic algorithms for concept learning. *Machine Learning,* 13, 161-188.

[11] Janikow, C., Z. (1993) A knowledge-intensive genetic algorithm for supervised learning, *Machine Learning*, 13,189-228.

[12] Congdon, C.,B.. (1995). *A comparison of genetic algorithms and other machine learning systems o a complex classification task from common disease research,* Doctoral dissertation, Department of Electrical Engineering and Computer Science, Univercity of Michigan.

[13] Koza, J. (1992). *Genetic Programming: On the programming of computers by means of natural selection.* MA:MIT Press.

[14] Punch, W.F., Goodman E.D., Pei Min, Chia-Shun Lai, Hovland P. & Enbody R. (1993). Further Research on Feature Selection and Classification Using Genetic Algorithms. *Proceedings of ICGA93,* 557-564.

[15] Turney, D.P (1995). Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm. *Journal of Artificial Intelligence Research,* 2, 369-409.

[16] Vafaie, H., DeJong, K. (1992). Genetic Algorithms as a Tool for Feature Selection in Machine Learning. *IEEE Computer Society Press, Los Alamos, CA,* 200-203.

[17] Bala, J., Huang, J., Vafaie, H., DeJong, K., Wechsler, H. (1995). Hybrid Learning Using Genetic Algorithms and Decision Trees for Pattern Classification. *Proceedings of IJCAI95, Montreal.*

[18] Schaffer, C. (1993). Overfitting avoidance as bias, *Machine Learning,* 10, 153-178

[19] Garey R., M, and Graham L.,R (1974) Performance bounds on the splitting algorithm for binary testing. *Acta Informatica,* 3(Fasc. 4):347--355.

[20] Norton, S., W. (1989). Generating Better Decision Trees. *In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* 800-815.

[21] Ragavan, H. and L. Rendell (1993), Lookahead Feature Construction for Learning Hard Concepts, *Proceedings of the Tenth International Conference on Machine Learning*, Amherst, MA, pp. 252--259 (Morgan Kaufmann, San Francisco, CA).

[22] Murthy, S. & Salzberg, S. (1995), Lookahead and pathology in decision tree induction, *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1025-1031

[23] Quinlan, J. R. and Cameron-Jones, R. M.(1995) Oversearching and layered search in empirical learning. *In Proceedings of the 14th International Joint Conference on Artificial Intelligence*,1019-1024, Montreal, Canada.

[24] Wall, M. (1996). *GAlib: A C++ Library of Genetic Algorithm Components.* M.I.T.

[25] Gathercole, C., Ross, P., (1997) Tackling the Boolean even N parity problem with genetic programming and limited-error fitness. *Genetic Programming 1997: Procceedings of the Second Annual Conference*, 119-127.

[26] Witten, I., Frank, E. (2000) *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann Publishers, San Mateo, CA.

[27] Quinlan, R. (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA.

[28] Holte, R.C. (1993). Very simple classification rules perform well on most commonly used datasets, *Machine Learning* 11,63-91.

[29] Blake, C., Keogh, E., & Merz, J. (2000) UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Authorized licensed use limited to: University of Leeds. Downloaded on June 09,2020 at 14:10:14 UTC from IEEE Xplore. Restrictions apply.