Improving on Symbolic Learning System Based on Genetic Algorithm

Limei Feng Xizhao Wang

Machine Learning Center, Faculty of Mathematics and Computer Science, Hebei University, Baoding 071002, P. R. China

Abstract

This paper uses GAssist system to get symbolic rules and proposes four techniques to improve it. A new population initialization method is applied and fitness scaling is used to promote the population's convergence. It also improves the deserted hierarchical selection operator and combines it with the MDL-based fitness function to control bloat effect. Finally, a new stop criterion to GA is studied. The experimental results show that the system is further improved. Comparing with other systems GA system is roughly comparable in generalization capacity but the efficiency needs improve.

Keywords: GAssist system, Population initialization method, Fitness scaling, Hierarchical selection operator, New stop criterion

1. Introduction

Symbolic learning is to induce rule sets from symbolic datasets. The values of symbolic datasets are discrete, unordered and don't have operation relation among values. Continuous datasets can be converted to symbolic ones by discretizing algorithm. Therefore, methods of symbolic learning are very practical and could be widely used. Research on symbolic learning has very important meaning. Using GA to search rule sets is one of the most important methods for symbolic learning problems.

An example of the use of GA for symbolic learning is GABIL system [1]-[2] described by DeJong, K. A.and Spears, W. M. A chromosome represents a rule sets and the length of it is variable. GAssist system is an improved edition of GABIL system [3]. A new smart crossover operator is proposed by Jaume Bacardit [4]. The operator selects more than two parents and merges the rules of them to make the final individual have the highest training accuracy. The ensemble technique Bagging is studied to combine with GAssist system. The tests show that the combination is very useful [5].

This paper aims at symbolic learning problems, thus continuous attributes are discretized by Kohonen algorithm [6]-[7] first. In order to make it convenient to compare, the system is called GAssist-Kohonen system.

Rule representation is introduced in section 2. Section 3 describes a new population initialization method and section 4 describes the introduced fitness scaling. Section 5 introduces the original hierarchical selection operator first, and then describe the improvement on it in detail. Experimental results are shown in section 7. Finally, a conclusion and future work are given.

2. Rule representation

The use of GA to solve symbolic learning problems is to seek a IF-THEN rule set that performs best on training examples. The learning process is to search a large space of candidate rule sets to get the one having the highest fitness. The hypothesis of GA is often encoded by bit strings. The candidate hypothesis space is determined by the encoding method. The encoding method of GAssist system is as follows:

- For a single attribute, a bit string is used to describe the value of it. For example, attribute A can take on three values a_1、a_2、a_3. Three bits are needed to represent it. Setting 1 represents the attribute can take the corresponding value. Then the string 010 indicates that the value of attribute A is the second one a_2.
- Given the bit string that represents a single attribulte, conjunctions of bit strings indicate the constrains on multiple attributes.
- The values of multiple attributes can be represented by conjunctions of the corresponding bit strings.

A rule is represented by the conjunction of the rule precondition bit string and the rule postcondition bit string.

Because the hypothesis often consists of more than one rule and a chromosome represent a rule set , the length of the chromosome is variable.

3. New population initialization method

```
Procedure:New population initialization method
Input:population size (PopSize); probability based on the rule ($P_1$); the number of rules of each chromosome(InitNumRule)
Output:initial population
1.For i=1 to PopSize
2.    default_class=a class randomly selected
3.    For j=1 to InitNumRule
4.        generate a real number P randomly
5.        Tag=0;
6.        If P< $P_1$
7.            While tag==0
8.                select a training example randomly and niche is the class of it
9.                IF niche! =default_class
10.                   convert the selected example to a rule directly
11.                   Tag=1;
12.               Else
13.                   generate a rule randomly
14.               End If
15.           End While
16.       End If
17.   End For
18. End For
```

Fig. 1: New population initialization method.

The initial population is generated total randomly in GAssist system. At a later time, Jaume Bacardit proposed CWSI [8] (smart initialization operator with class-wise sampling) method. Experiments on 25 datasets indicates that the performance of CWSI is better on only one dataset. Therefore, the initial population is randomly generated in GAssist-Kohonen system that is the basic system for us to solve symbolic problems.

If the initial population is generated total randomly not considering the information of training examples the performance of the initial rule sets couldn't be very good . But if excessive information of training examples is used the generated rules may be too specific and the generalization capacity is also bad. Therefore, in this paper the rules of the initial population consist of two parts. One part is randomly generated and the other part is converted directly by exmples. The technique is different from CWSI. We use the probability P_1 to control the proportion of the introduced training examples. For each rule of each chromosome, a real number is randomly generated. If it is smaller than P_1, select an training example randomly and then convert it to a rule directly. If it is not small than P_1 , generate a rule randomly. GAssist system uses a static default rule mechanism [9], thus the class of the selected training example can't be the default class of the chromosome. The algorithm of the new initialization method is described in Figure 1.

4. Introduction of fitness scaling

In the earlier stage of evolution, if a few chromosomes has very high fitness values, they will be selected with higher probabilities. Thus, the diversity of the population is lost, which leads to premature convergence. In the later stage of evolution, the difference of chromosomes becomes smaller and the selection process is similar to randomly select. So, the evolution process becomes slow. In order to avoid the above two drawbacks, fitness scaling is introduced. Assume f is the original fitness value and *f'* is the new one [10], the formula of fitness scaling is as follows:

$$f' = af + b \tag{1}$$

where coefficients a and b is determined by the following two conditions：

$$f'_{avg} = f_{avg} \tag{2}$$

$$f'_{\max} = c \cdot f_{avg} \tag{3}$$

where c is the number of expected ones desired for the best chromosome. The new fitness value may be negative. In this case, the following formula is used.

$$f'_{\min} = 0 \tag{4}$$

The following inequation is first calculated when calculating the values of coefficients a and b:,

$$f_{\min} > \frac{c \cdot f_{avg} - f_{\max}}{c-1} \tag{5}$$

if the above inequation is satisfied, then :

$$a = \frac{f_{avg}}{f_{avg} - f_{\min}} \tag{6}$$

$$b = \frac{-f_{avg} \cdot f_{\min}}{f_{avg} - f_{\min}} f_{avg} \tag{7}$$

if the above inequation is not satisfied, then :

$$a = \frac{c-1}{f_{\max} - f_{avg}} f_{avg} \tag{8}$$

$$b = \frac{f_{\max} - c \cdot f_{avg}}{f_{\max} - f_{avg}} f_{avg} \tag{9}$$

5. Improving on hierarchical selection operator

5.1. Introduction of hierarchical selection operator

The fitness function of GABIL system only considers the training accuracy of the chromosome and ignore the complexity factor, which leads to the system to improve the training accuracy by memorizing training examples. As a result, the length of the chromosome increases without control and the generalization capacity of the rule set is bad, which is called bloat effect [11].

GAssist system experienced a developing process. In order to control bloat effect, it ever used hierarchical selection operator. If the training accuracy difference of two chromosomes is small than the threshold setted in advance, the shorter one is selected first. If it is not small than the setted theshold, the chromosome having higher fitness is selected. At a later time, the MDL-based fitness function [12] is proposed , which has better performace than the hierarchical selection operator. Therefore, the hierarchical selection operator is deserted. However, it has the following two reasonable points:

- Consistant with the Occam's razor principle

The hierarchical selection operator is integrated with the tournament selection operator. When the performance of two chromosomes is similar, the shorter one is selected first, which is consistant with the Occam's razor principle。

- Specific-to-general policy

In the earlier stage of evolution, the training accuracy difference of chromosomes is large, the chromosome with higher training accuracy is selected, the influence of the length factor is small.

However, with the evolving process, the training accuracy difference becomes smaller than the threshold. The influence of hierarchical selection operator becomes bigger and shorter chromosome is selected first. In this way, the lengths of chromosomes becomes shorter gradually. Therefore, hierarchical selection operator controls bloat effect in a reasonable specific-to-general process.

Except the above two reasonable points, the hierarchical selection operator has a drawback that the training accuracy threshold is hard to set. If it is too big, the condition is easily satisfied and short chromosomes are selected with high probability. In this case, the system attach importance to the length factor excessively, which leads to the loss of diversity. The population can't evolve and premature convergence occures. On the other hand, if it is too small, then the condition is hard to satisfy and the chromosome with high training accuracy is selected more often. In this case, the length factor of the chromosome is ignored in fact and the sytem degenerates to GABIL system.

Taking the merits and demerits of the hierarchical selection operator into consideration, we improve on it first, and then reintroduce it to our system.

5.2. Improvement on the operator

```
Procedure:tune the threshold of the improved
hierarchical selection operator
Input:Maximum GA iterations( maxIterations ) ;
      fitness difference threshhold (threshold) ;
      continuous iterations (lastIterations) ; tune
      coefficient (a)
1. i=0;
2. count=0;
3. While i < maxIterations
4.      Run GA
5.      If the performance of the best
chromosome is improved
6           count=0;
7.      Else
8.          count++;
9.      End If
10.     IF count > lastIterations
11.         threshold=threshold*a;
12.         count=0;
13.         i++;
14.     End If
15. End While
```

Fig. 2: Tune the threshold of the improved hierarchical selection operator.

GAssist system uses MDL-based fitness function that considers the accuracy and complexity and also trades off the two factors. If hierarchical selection operator is introduced into GAssist system , the meaning of the threshold is the fitness difference rather than training accuracy difference. Besides, the threshold becomes easier to tune because of the change of fitness function. We only tune it in one direction. If the threshold is too small, then the fitness difference between two chromosomes isn't small than the threshold and hierarchical selection operator has no effect and chromosome with higher fitness value is selected. Assume threshold is 0, then the system goes back to GAssist system. In this case, although the improved hierarchical selection operator can't help to control bloat effect, it doesn’t have any bad effect. Therefore, in this case the threshold could be very small and it becomes easier to tune.

In this paper,we tune the threshold by a heuristic process. First, set a threshold that is large enough to avoid premature convergence (experiments show that, 0.01 is safe [12]). With the iterating process going, if the performance of the best chromosome has not been improved after a continuous number of iterations, the threshold is tuned to smaller. This algorithm is descibed in Figure 2.

6. The new stop criterion

In general, we set the maximum number of iterations as the stop criterion, which is very simple, but inaccurate. If using generational replacement with elitism for the best chromosome, we can set the stop criterion according to the variation of the best chromosome [13]. GAssist system uses elitism selection, while using the maximum iterations as the stop criterion, so we propose that GA stops as the best chromosome has similar and stable performance through a certain number of iterations (MaximumBestDelayGA). The new stop criterion is as follows:

- The standard deviation of the accuracy for the last MaximumBestDelayGA iterations of the best chromosome is not larger than a threshold (thresh-accuracy).
- The difference of the maximum number of rules and the minimum number of rules of the best chromosome in the MaximumBest-DelayGA iterations is not larger than a threshold (thresh-rule).
- The current iteration reaches the maximum number iterations.

If both the above two conditions are satisfied, GA stops iterating. If one of the condition is not satisfied GA doesn’t stop running until it iterates the maximum number of iterations.

7. Experiments

7.1. Experimental setup

In this paper eight datasets are obtarined from UCI [14] machine learning repository. They are Iris, Glass, Hepatitis(Hepa), Hungarian(Hung), Balance-scale(Bala), Credit-screening, Tic-Tac-Toe(Tic) and Mushroom(Mush). The features of them are shown in Table 1.

Stratified ten-fold cross-validation [15] is used in experiments. In order to test all the four techniques proposed in this paper, experiments are conducted as the following analysis:

Parameter	Value
General parameters	
Population size	300
Initial Number of rules in an individual	15
rule-wise Probability	0.10
Selection Algorithm	Tournament
Tournament size	3
Fitness similarity thresh	0.005
Crossover probability	0.6
Probability of mutating an individual	0.6
Number of seeds for each experiment	15
Probability of value 1 in initialization	0.75
Number of incremental learning segments	2,3,15
MDL Weight heuristically adjusting	
Iteration of activation	25
InitialRateOfComplexity	0.075
MaximumBestDelay	10
WeightRelaxationFactor	0.9
Rule deletion operator	
Iteration of activation	5
Minimum Number of rules before disabling the operator	numClasses+3
fitness penalty	
Iteration of activation	25
Minimum No. of rules	Maximum-6
Improved Hierarchical Selection threshold adjusting	
Initial value	0.01
Iteration of activation	40
MaximumBestDelaythresh	10
ThreshRelaxationFactor	0.9
The new stop criterion	
Number of GA iterations	1000
MaximumBestDelayGA	10
thresh_accuracy	0.005
thresh_rules	2

Table 3: Parameters of GA.

First, the performance of GAssist-Kohonen System is tested. Then the first technique is tested, if it is effective the second technique is introduced. In this way , four techniques should all be tested. Finally, we get five configurations that are detailed described in Table 2 .The detailed setting of parameters is shown in Table 3.

datasets	Number of examples	Number of attributes	Number of classes
Iris	150	5	3
Hepa	155	20	2
Glass	214	10	7
Hung	294	14	2
Bala	625	5	3
Cred	690	16	2
Tic	958	10	2
Mush	8124	23	2

Table1: feartures of datasets

Code	Algorithm configuration
1	GAssist-Kohonen
2	1 +new population initialization method
3	2+fitness scaling
4	3+improved hierarchical selection operator
5	4+new stop criterion

Table 2: Algorithm configurations.

7.2. Experimental results and analysis

Experimental results of five configurations are shown in Table 4. For each configuration and dataset we show the average value of: (1) the testing accuracy (Test Acc.), (2) the number of rules (No. of rules) (4) the running time in seconds. Analysing Table 4, we can get the following conclusions:

1) Testing accuracy is increased after the new population initialization technique is used while the size of the rule set and running time are nearly the same.

2) After calculating the fitness of chromosomes, fitness scaling is applied. Observing experimental results of configuration 2 and 3, we can see that the performance of the rule set is improved (the accuracy is increased or the size of rule set is reduced). Analysing in theory, fitness scaling can keep the competition of chromosomes in the population and promote the population's convergence. Therefore, the efficiency of the system should be improved, which has not been proved in experiments. It is because the new stop criterion has not been applied yet and GA has to run the maximum number of iterations.

3) Observing results of configuration 3 and 4, we can see that both the number of rules and the run-time are reduced significantly for all the datasets with none or only a little accuracy lose. Therefore, we can get a conclusion that our original objective of using the improved hierarchical selection operator to assist the MDL-based fitness function to control bloat effect has been achieved.

4) After the new stop criterion is used, three different results are observed:

- For Iris, Glass,Balance-scale and Mushroom, run-time is reduced significantly with none accuracy loses. In this case, the population has already converged to a global optimal solution, but the number of GA iterations hasn't been reached, so the new stop criterion works by stopping the searching process in time.
- For Hepatitis and Hungarian, although the running time is longer we get better solutions (increased average accuracy and reduced average number of rules). In this case, the population can still evolve, but the number of GA iterations is reached and the learning process is forced to stop. Because the new stop criterion contains stability of the performace of the best chromosome and large maximum iterations. Therefore, the new stop criterion going on to improve the performance by running more rounds.
- For the credit-screening and Tic-tac-toe, it suffers from over-fitting. In this case, the new stop criterion is too strict and GA can't search such a good solution, so it keeps running until reaching the maximum number of iterations which is setted a large number (1000).

Observing the above results , we can conclude that the four techniques are effective.

8. Comparision with other symbolic learning methods

There are many symbolic learning methods. For example, FCV, Classification based on Multiple Association Rules (CMAR), Bayes optimal classfier (Bayes) and C4.5. We compare our system with them. Experimental resulsts are shown in Table 5 and Table6.

Observing the results we can get the following conclusions:

- Because GA is a global search algorithm., comparing with other symbolic learning methods, it has a valuable merit that performs better for most datasets. Thus, we conclude that it can be used in more extensive questions.
- The efficiency of our system should be improved .

9. Conclusions and further work

In this paper, four techniques are proposed to improve GAssist system. First a new population initialization method is proposed and then fitness scaling is introduced into the

datasets	GA	FCV	CMAR	Bayes	C4.5
Iris	98.8	83.1	93.3	74.2	94.4
Hepa	85.5	87.6	81.5	87.1	76.0
Glass	70.1	40.7	71.0	44.0	75.1
Hung	84.2	91.0	83.3	83.9	77.1
Bala	92.1	87.3	80.8	87.4	62.3
Cred	84.3	91.6	84.9	86.3	85.5
Tic	86.9	99.5	95.1	70.0	87.6
Mush	98.4	100.0	96.2	99.5	100.0

Table 5: Average accuracy of the global comparison experiments.

datasets	GA	FCV	CMAR	Bayes	C4.5
Iris	3.4	0.1	5.5	0	0
Hepa	34.8	0.1	25.7	0	0
Glass	7.7	0.1	60.1	0	0
Hung	44.3	0.1	62.0	0	0
Bala	21.71	0.2	0.1	0	0
Cred	85.4	0.2	15.4	0	0
Tic	73.4	0.2	4.5	0	0
Mush	148.7	3.8	8.4	0	0

Table 6: Average time of the global comparison experiments.

dataset	Cofiguration 1			Cofiguration 2			Cofiguration 3			Cofiguration 4			Cofiguration 5		
	Test Acc.	No. of rule	Time (s)	Test Acc.	#rule.	Time (s)	Test Acc.	No. of rules	Time (s)	Test Acc.	No. of rules	Time (s)	Test Acc.	No. of rules	Time (s)
Iris	99.2	6.1	68.8	99.4	6.0	68.3	100.0	5.9	56.2	98.8	5.6	10.2	98.8	7.3	3.4
Hepa	85.1	6.2	38.2	85.6	6.2	37.3	85.9	6.2	39.8	83.9	5.7	13.2	85.5	6.0	34.8
Glass	69.1	10.8	92.4	69.5	10.8	92.4	70.0	9.8	93.7	70.5	9.2	19.7	70.1	9.5	7.7
Hung	82.9	11.6	12.5	83.2	11.4	12.4	83.7	6.9	51.6	82.6	6.2	10.1	84.2	6.1	44.3
Bala	91.7	6.1	97.6	92.1	6.0	96.7	92.0	7.0	107.1	92.1	7.0	85.3	92.1	6.0	21.71
Cred	87.5	7.8	30.6	88.2	7.8	29.6	87.0	6.3	38.5	87.8	6.0	13.2	84.3	5.2	85.4
Tic	93.1	17.3	137.9	93.5	19.3	143.3	94.6	20.2	143.2	94.7	7.4	31.7	86.9	7.7	73.4
Mush	99.5	5.0	268.6	99.7	5.0	263.6	99.4	5.4	267.0	99.4	3.5	234.9	98.4	5.7	148.7

Table 4: Results of five algorithm configuration.

system. We also improve on the deserted hierarchical selection operator and combine it with the MDL-based fitness function to control bloat effect. Finally, a new stop criterion is applied. Experimental results show that for most datasets generalization capacity of the rule set and the efficiency are improved significantly. However, for a few datasets, the new stop criterion seems too strict, which leads to over-fitting. Therefore, the new stop criterion still needs improvement. Comparing with other symbolic learning methods, it is proved that GA has better performance for more extensive problems but the efficiency is still slow.

Therefore, as further work, we should study the way to fix the weakness of the new stop criterion. Besides, we should study new techniques to improve the efficiency of the system.

Acknowledgement

This work is financially supported by National Science Fund of China (60473045) and Hebei Province High Technology project (04213533).

References

[1] K.A. DeJong, W.M. Spears, Learning concept classifi-cation rules using genetic algorithms. *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 651-656, 1991.

[2] K. A.DeJong, William M.Spears, Diana F.Gordon, Using genetic algorithms for concept learning. *Machine Learning*, 13(3):161-188, 1993.

[3] J. Bacardit, Pittsburgh genetics-based machine learning in the data mining era: representations, generalization, and run-time. *Doctoral disertation*, Ramon Llull University, Barcelona, Catalonia, Spain, 2004.

[4] J. Bacardit, N. Krasnogor, Smart crossover operator with multiple parents for a pittsburgh learning classifier system. *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 1441-1448, 2006.

[5] J. Bacardit, N. Krasnogor, Empirical evaluation of ensemble techniques for a pittsburgh learning classifier system. *Ninth International Workshop on Learning Classifier Systems,IWLCS2006, Lecture Notes in Artificial Intelligenge (to appear), Springer* , 2006.

[6] T. Kohonen, *Self-organization and associative memory*. Springer, Berlin, 1988.

[7] C.T. Lin, C.S.G. Lee, Neural-network-based fuzzy logic control and decision system. *IEEE Trans. Comput*, 12: 1320-1336, 1991.

[8] J. Bacardit, Analysis of the initialization stage of a pittsburgh approach learning classifier system. *Genetic and Evolutionary Computation Conference*. Washington DC. pp. 1843-1850, 2005.

[9] J. Bacardit, David E. Goldberg, Martin V. Butz, Improving the performance of a pittsburgh

learning classifier system using a default rule. *Seventh International Workshop on Learning Classifier Systems*, 2004.

[10] T. Kvan, Collaborative design: what is it?. *AUTOMATION IN CON2 STRACTION*, 9(4): 409-415, 2000.

[11] W. B.Langdon, , Fitness causes bloat in variable sizerepresentations. *Workshop on Evolutionary Co-mputation with Variable Size Representation at ICGA-97*, May 14 1997.

[12] J. Bacardit., J. M. Garrell, Bloat control and generalization pressure using the minimum description length principle for a pittsburgh approach learning classifier system. *Sixth International Workshop on Learning Classifier Systems Chicago*, July 2003.

[13] M.Q. Li, D. Lin, *Genetic algorithm's theory and application*, Science Press, 2002.

[14] UCI repository of machine learning databases and domain theories, FTP address: ftp : // ftp. ics. uci. edu / pub / machine - learning - databases.

[15] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection. *In International Joint Conference on Artificial Intelligence,* pp.1137-1145, 1995.