Discovering comprehensible classification rules using Genetic Programming: a case study in a medical domain

Celia C. Bojarczuk
CEFET-PR, CPGEI.
Av. Sete de Setembro, 3165.
Curitiba - PR.
80230-901. Brazil
celia@cpgei.cefetpr.br

Heitor S. Lopes
CEFET-PR, CPGEI.
Av. Sete de Setembro, 3165.
Curitiba – PR.
80230-901. Brazil
hslopes@cpgei.cefetpr.br

Alex A. Freitas
PUC-PR, PPGIA-CCET.
Rua Imaculada Conceição, 1155
Curitiba – PR.
80.215-901. Brazil
alex@ppgia.pucpr.br
http://www.ppgia.pucpr.br/~alex

Abstract

This work it is intended to discover classification rules for diagnosing certain pathologies. These rules are capable of discriminating among 12 different pathologies, whose main symptom is chest pain. In order to discover these rules it was used genetic programming as well as some concepts of data mining, particularly the emphasis on the discovery of comprehensible knowledge.

1 INTRODUCTION

In order to classify and diagnose some pathology, one must verify which predicting attributes are most associated with that disease. In this work there are 189 predicting attributes and 12 different diseases (classes) whose main characteristic is chest pain. The predicting attributes refer to characteristics of the chest pain, other symptons reported by the patient, signals observed by the physician, details of clinical history and results of laboratory tests. The diseases are: stable angina, unstable angina, acute myocardial infarction, aortic dissection, cardiac tamponade, pulmonary embolism, pneumothorax, acute pericarditis, peptic ulcer, esophageal pain, musculoskeletal disorders, psychogenic chest pain. The goal is to predict which of those diseases a patient has, given the values of the 189 predicting attributes for the patient.

The same problem, chest pain diagnosis, was previously addressed by (*Lopes et al.*, 97) using analogical reasoning and parallel genetic algorithms. In that work the overall results were satisfactory concerning the predictive accuracy, but they had the disadvantage of not being comprehensible by users (physicians). The problem is that the genetic algorithm evolved a set of numeric weights for the attributes, and all the 189 attributes plus their numeric weights were used for classification (which was done by a kind of nearest neighbor method). Hence, the result of the system (the best individual found by the GA) involved all the attributes and too much low-level information (the attribute weights), making it very difficult for the user to understand the results.

Recently, however, there has been a growing interest in the area of data mining, where the goal is to discover knowledge that not only has a high predictive accuracy but also is *comprehensible* for users (Fayyad *et al*., 96, Freitas & Lavington, 98). Hence, the user can understand the system's results and combine them with his/her knowledge to make a well-informed decision, rather than blindly trusting in the incomprehensible output of a "black box" system.

This work addresses the above classification problem in the context of data mining. In this context, the discovered knowledge is often expressed in the form of IF-THEN prediction (classification) rules. The IF part of the rule contains a logical combination of conditions on the values of predicting attributes, and the THEN part contains the predicted pathology (class) for a patient whose clinical attributes satisfy the IF part of the rule.

The use of genetic programming (GP) for discovering comprehensible classification rules, in the spirit of data mining, is a relatively underexplored area. We believe this is a promising approach, due to the effectiveness of GP in searching very large spaces (Koza, 92; Koza, 94) and its ability to perform a more open-ended, autonomous search for logical combinations of predicting attribute values.

Unlike most data mining algorithms, GP can naturally discover rules expressed in a kind of first-order logic of the form $<Att_i\ Op\ Att_j>$, where Att_i and Att_j are predicting attributes of the same data type, $i \neq j$, and Op is a relational operator (typically "=", "≤", or ">"). First-order logic has more expressive power than propositional logic, since this latter allows only the discovery of rule conditions of the form $<Att\ Op\ Val>$, where Att is a predicting attribute, Op is a relational operator and Val is a value belonging to the domain of Att. To see the advantage of the greater expressive power of first-order logic, note that, if the predicting attributes being compared are real-valued, a first-order condition of the form $Att_i <$

Att_j (e.g. *Income < Expenses*) cannot be expressed in a finite way by a propositional-logic representation.

To summarize, the goal of this paper is to investigate the effectiveness of GP in discovering high-level, comprehensible rules for the application domain of chest pain diagnosis.

This paper is organized as follows. Section 2 presents a brief overview of GP. Section 3 describes in some detail the GP system proposed in this paper. Section 4 reports and analyzes the results of computational experiments. Section 5 discusses related work. Finally, section 6 concludes the paper.

2 A BRIEF OVERVIEW OF GENETIC PROGRAMMING

Genetic programming is a powerful search method inspired by natural selection. The basic idea is to evolve a population of "programs" candidate to the solution of a specific problem. A program (an individual of the population) is usually represented in the form of a tree, where the internal nodes are functions (operators) and the leaf nodes are terminal symbols. Both the function set and the terminal set must contain symbols appropriate for the target problem. For instance, the function set can contain arithmetic operators, logic operators, mathematical functions, etc; whereas the terminal set can contain the variables (attributes) of the target problem.

Each individual of the population is evaluated with respect to its ability to solve the target problem. This evaluation is performed by a fitness function. Then the individual undergoes the action of genetic operators such as reproduction and crossover. The reproduction operator selects individuals of the current population in proportion to their fitness values, so that the fitter an individual is the higher the probability that it will take part in the next generation of individuals. The crossover operator replaces a randomly selected subtree of an individual with a randomly chosen subtree from another individual.

Once reproduction and crossover have been applied according to given probabilities, the newly created generation of individuals is evaluated by the fitness function. This process is repeated iteratively, usually for a fixed number of generations. The result of genetic programming (the best solution found) is the fittest individual produced along all generations.

3 THE PROPOSED GENETIC PROGRAMMING SYSTEM

In this section we describe our proposed genetic programming (GP) system for the classification problem discussed in the Introduction.

The terminal set consists of the previously described 189 attributes. All the attributes are binary. The function set consists of three logic functions, namely AND, OR and NOT. These functions were chosen due to our desire for discovering high-level, comprehensible rules, as discussed before. Hence, an individual (program) consists of a combination of the above three logic operators applied to some predicting attributes. Each individual encodes the IF part of a rule.

For instance, the hypothetical individual: *(AND (OR dizzy fever) x_ray_not_ok)* would correspond to an IF part with the following conditions: *IF ((dizzy OR fever) AND x_ray_not_ok)).*

Note that the THEN part of the rule (the predicted class) is not encoded into the individual. The reason for this is the fact that in a given run of the GP all individuals represent rules predicting the same class, as follows.

Each run of our GP solves a two-class classification problem, where the goal is to predict whether or not the patient has a given disease. Therefore, in order to generate classification rules for the 12 classes, we need to run the GP 12 times. In the first run the GP would search for rules predicting class 1; in the second, for class 2, and so on. When the GP is searching for rules predicting a given class, all other classes are effectively merged into a large class, which can be conceptually thought of as meaning that the patient does not have the disease predicted by the rule.

The use of an individual for predicting whether or not a patient has a given disease (i.e. the disease associated with the current GP run) is straightforward. Recall that each patient is described by the values of 189 binary attributes. Therefore, the system simply checks whether the set of 189 binary values associated with a patient satisfy the IF part of the rule specified by the individual. If so, the system predicts that the patient has the corresponding disease, otherwise the system predicts that the patient does not have that disease.

Since the THEN part of the rule does not need to be encoded into the individual, henceforth we will refer to each individual as a rule, for the sake of brevity and simplicity. In the next subsection we describe the fitness function that we use for evaluating the quality of the rules associated with individuals of the GP.

3.1 FITNESS FUNCTION

The fitness function evaluates the quality of each rule (individual). This work uses the fitness function employed by (Lopes *et al.,* 97). Before we can define the fitness function, it is necessary to recall a few basic concepts on classification-rule evaluation. When using a rule for classifying an unknown case, depending on the class predicted by the rule and on the true class of the patient (his disease), four types of results can be observed for the prediction, as follows:

- true positive - the rule predicts that the patient has a given disease and the patient does have that disease;

- false positive - the rule predicts that the patient has a given disease but the patient does not have it;
- true negative - the rule predicts that the patient does not have a given disease, and indeed the patient does not have it;
- false negative - the rule predicts that the patient does not have a given disease but the patient does have it.

Let *tp*, *fp*, *tn* and *fn* denote respectively the number of true positives, false positives, true negatives and false negatives observed when a rule is used to classify a set of cases (patients). Our fitness function combines two indicators commonly used in the medical domain, namely the sensitivity (*Se*) and the specificity (*Sp*), defined as follows:

$$Se = tp / (tp + fn) \quad (1)$$

$$Sp = tn / (tn + fp) \quad (2)$$

Finally, the fitness function used by our GP is defined as the product of these two indicators, i.e.:

$$fitness = Se * Sp \quad (3)$$

Therefore, the goal of our GP is to maximize both the *Se* and the *Sp* at the same time. This is an important point, since it would be relatively trivial to maximize the value of one of these two indicators at the expense of significantly reducing the value of the other. Furthermore, the above fitness function has the advantages of being simple and returning a meaningful, normalized value in the range [0..1]. A good analysis of the above fitness function per se (independent of any evolutionary algorithm), as well as other related measures of predictive accuracy, can be found in (Hand, 97).

4 COMPUTATIONAL RESULTS

We have applied our proposed GP system to the previously described classification problem. In all the experiments reported in this paper the initial population was generated by the well-known ramped half-and-half method, which creates an equal number of trees for tree-depth values varying from 2 to the maximum depth (Koza, 92). The experiments were done using the Lil-GP system (Zongker *et al.*, 96), and all remainder parameters of the GP were left as the standard.

The data set consists of 138 examples (patients) and 189 attributes. As usual in the classification literature, the data set was randomly partitioned into two sets, a training set (with 90 examples) and a test set (with 48 examples). We used the same training/test set partition employed by (Lopes et al. 97) in order to make possible the comparision of results.

Note that, in principle, this can be considered a difficult classification problem, since the number of examples can be considered small, once we take into account the large number of attributes.

For each class we run the GP three times, varying some GP parameters. Hence, the GP was run 3 x 12 = 36 times, and each of these runs was independent of the others. More precisely, table 1 shows the parameter settings used in each of our three sets of experiments. The result of each run is the best rule (individual) found for that run and the best rule was the one with the best performance on the training set. For each class we selected the best rule out of the three rules found in the three GP experiments.

Therefore, the final result of our experiments is a set of 12 rules, one for each class. We analyzed the quality of these discovered rules in three ways, namely by evaluating the predictive accuracy of the rule set as a whole, the predictive accuracy of individual rules and the comprehensibility of the rules. These analyses are discussed in the next three subsections, respectively.

4.1 PREDICTIVE ACCURACY OF THE RULE SET AS A WHOLE

Among the several different criteria that can be used to evaluate the predictive accuracy of a discovered rule set, we have used the well-known accuracy rate, which, despite its drawbacks (Hand, 97), seems to be still the most used in the classification literature. The accuracy rate is simply the ratio of the number of correctly-classified test examples over the total number of test examples. Our GP achieved an accuracy rate of 77.08% - i.e. the discovered rule set correctly classified 37 out of 48 examples (test set).

4.2 PREDICTIVE ACCURACY OF INDIVIDUAL RULES

Although reporting the predictive accuracy of the rule set as a whole is commonplace in the literature, in practice the user will analyze or consider one rule at a time. Hence, it makes sense to report the accuracy rate of each individual rule. Recall that this is feasible, in our case, because our system discovered only 12 rules (one for each class). Such an analysis of the predictive accuracy of individual rules is less feasible in cases where the data mining algorithm discovers very many rules.

Table 2 reports the figures concerning the predictive accuracy of each of the 12 discovered rules. Each row of this table refers to the best rule found in the three GP runs performed for the corresponding class #. Again, all the figures reported in this table refer to the performance of the discovered rules on the test set.

Overall, the discovered rules have both high *Se* and high *Sp*, which leads to a high value of the product *Se x Sp*. In particular, five rules (the ones predicting classes 3, 4, 7, 9, 10)[1] have both *Se* and *Sp* (as well as their product) greater than or equal to 0.95. However, some rules are not so good. The rules predicting classes 8 and 11, despite their

[1] The class numbers refers to the diseases listed in the Introduction, in the order they appear.

high value of *Sp*, are less likely to be considered as truly interesting and useful by a human user, due to their very low *Se*. This is consistent with the results reported by (Lopes *et al.,* 97), since classes 8 and 11 are indeed the most difficult ones to predict.

4.3 COMPREHENSIBILITY OF THE DISCOVERED RULES

As mentioned in the Introduction, in this work we are interested not only in the predictive accuracy of the discovered rules, but also in the comprehensibility of the rules - in the spirit of data mining.

Rule comprehensibility is significantly more difficult to measure in an objective way, in comparison with predictive accuracy. There is a considerable subjective aspect in the former. In most of the literature, however, rule comprehensibility is associated with syntactic complexity. In general, the smaller the number of rules and the shorter (the smaller the number of conditions of) the rules the better.

Concerning the number of discovered rules, our GP system (by design) provides the best possible result, that is exactly one rule for each class. This minimization of the number of discovered rules saves the potentially precious time of the human user, who is required to analyze only the very best rule found for each class. In contrast, some data mining algorithms may easily overload the user with a large number of discovered rules, that can hardly be considered "comprehensible" knowledge.

The remaining question is: how short are the rules discovered by the GP? Some of the rules (particularly the ones predicting classes 1, 6 and 10) were quite long. However, the GP did find some short, very comprehensible rules. In particular, the shortest two rules discovered by the system were as follows:

IF (ecg_lv OR x_ray_ok_no) AND (ech_et) THEN class 5

IF (x_ray_air_in_ps) THEN class 7

In the first rule, the attributes in the IF part denote respectively "low voltage of the eletrocardiogram", "abnormality found in chest x-ray" and "evidence of tamponade in the echocardiogram". Class 5 represents the disease "cardiac tamponade". In the second rule, the attribute in the IF part denotes "air in pleural space observed in the chest x-ray". Class 7 represents the disease "pneumothorax".

It is important to note that we did not use any parsimony measure in our fitness function. Hence, the above simple two rules were discovered only due to their high value of a predictive accuracy-related measure - equation (3). Actually, the predictive accuracy of the above simple two rules is quite high, as can be seen in table 2.

5 RELATED WORK

The PADO system (Teller & Veloso, 95) is a sophisticated variant of GP that learns an entire classification algorithm for an arbitrary signal type (images, sounds, etc.). Most of the "function set" used by PADO involves mathematical functions and low-level primitives, so that the system does not discover high-level classification rules.

(Marchesi *et al.*, 97) have also used GP for solving a classification problem. Their work also used low-level features of a signal in the terminal set and mathematic functions in the function set, so that they have not discovered high-level classification rules. A common feature of the above projects, as well as other projects on the classification of low-level signals, is that they do not discover high-level IF-THEN rules, and the discovered "knowledge" (if we can call it this way, which is doubtful) typically is evaluated only with respect to predictive accuracy, not with respect to comprehensibility.

(Sherrah *et al.*, 97) propose a GP system which performs both feature selection and feature construction (using non-linear functions). The system also performs a kind of algorithm selection, in the sense that each individual has one of three classification algorithms associated with it. However, none of the three algorithms focus on the discovery of comprehensible classification rules.

We now review some GP projects somewhat more related to the data mining goal of discovering comprehensible knowledge. The GPDT system evolves variable-length programs in the form of decision-trees (Nikolaev & Slavov, 97). In the spirit of the decision-tree paradigm, the function set consists of predicting attributes. This differs from our system, where the function set consists of the logic operators AND, OR, NOT.

(Ngan *et al.*, 98) use GP for discovering comprehensible rules. Their approach also has the interesting feature that different discovered rules can predict different attributes, contributing for a greater autonomy of the system. This is a kind of generalization of the classification task, where all discovered rules predict the same attribute. However, the discovered rules are evaluated within the framework of support-confidence proposed for association rules (Agrawal *et al.*, 93). This framework, unlike classification, does not evaluate the predictive accuracy of discovered rules on an *unseen* test set. Furthermore, their approach is based on the use of a grammar to restrict the search space and ensure the syntactical correctness of the rules. Unfortunately, the grammar is entirely application domain-dependent, i.e. a grammar must be written for each application domain.

(Martin *et al.*, 98) use GP for discovering comprehensible classification rules. Like our approach, their system discovers rules expressed in first-order logic representation. Unlike our work, the main characteristic of their work was to integrate GP with relational databases systems containing multiple relations (tables) in such a way that GP individuals are associated with SQL queries.

To achieve this goal they proposed a constraint-based model (taking into account characteristics of relational databases) for individual representation and genetic operators, which is not necessary in our case. Another work whose main characteristic is the integration of GP with relational database systems, using GP individuals to represent SQL queries, is discussed in (Freitas, 97). This work addresses the task of classification and generalized rule induction, but it assumes that the data being mined is contained in a single relation.

6 CONCLUSIONS AND FUTURE WORK

Despite the small number of examples available in our application domain (taking into account the large number of attributes), the results of our experiments can be considered very promising. The discovered rules had a good performance concerning predictive accuracy, considering both the rule set as a whole and each individual rule.

Furthermore, which is more important from a data mining viewpoint, the system did discover some comprehensible rules, as discussed in section 4.3. It should be emphasized that this result was achieved even without including in the fitness function a term penalizing long rules. The comprehensibility of the discovered rules could, in principle, be improved with a proper modification of the fitness function. How much predictive accuracy would be reduced with such a modification is a question whose answer requires further research.

Another possibility for future research would be to perform a careful selection of attributes in a preprocessing step, in order to reduce the number of attributes (and the corresponding search space) given to the GP. Attribute selection is a very active research area in data mining (Liu & Motoda, 98).

Finally, future work should also include the application of the GP system proposed in this paper to other data sets, to further validate the results reported in this paper.

References

R. Agrawal, T. Imielinski and A. Swami (1993). Mining association rules between sets of items in large databases. In *Proceedings. 1993 Inernational Conference on Management of Data (SIGMOD-93)*, 207-216.

U. M. Fayyad, G. Piatetsky-Shapiro and P. Smyth (1996). From data mining to knowledge discovery: an overview. In: U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (eds.), *Advances in Knowledge Discovery & Data Mining*, 1-34, AAAI/MIT.

A. A. Freitas (1997). A genetic programming framework for two data mining tasks: classification and generalized rule induction. In *Genetic Programming 1997: Proceedings of 2nd Annual Conference.,* 96-101. Morgan Kaufmann.

A. A. Freitas and S. H. Lavington (1998). *Mining Very Large Databases with Parallel Processing.* Kluwer.

D. Hand (1997). *Construction and Assessment of Classification Rules.* John Wiley & Sons.

J. R. Koza (1992). *Genetic Programming: on the Programming of Computers by Means of Natural Selection.* MIT Press.

J. R. Koza (1994). *Genetic Programming II: Autonomous Discovery of Reusable Programs.* MIT Press.

H. Liu and H. Motoda (1998). *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer.

H. S. Lopes, M. S. Coutinho, R. Heinisch, J. M. Barreto, W. C. Lima (1997). A Knowledge-based system for decision support in chest pain diagnosis. *Medical & Biological Engineering & Computing* **35** (suppl., part I):514.

B. Marchesi, A. L. Stelle, H. S. Lopes (1997). Detection of epileptic events using genetic programming. *Proceedings of 19th International Conference. IEEE/EMBS,* 1198-1201. IEEE Press.

L. Martin, F. Moal and C. Vrain (1998). A relational data mining tool based on genetic programming. *Principles of Data Mining & Knowledge Discovery: Proceedings of. 2nd. European Symposium (LNAI)* **1510**, 130-138. Springer-Verlag.

P. S. Ngan, M. L. Wong and K. S. Leung (1998). Using grammar based genetic programming for data mining of medical knowledge. *Genetic Programming 1998*: *Proceedings of 3rd Annual Conference.,* 254-259. Morgan Kaufmann.

N. I. Nikolaev and V. Slavov (1997). Inductive genetic programming with decision trees. *Proceedings of 1997 European Conference on Machine Learning (ECML-97).*

J. R. Sherrah, R. E. Bogner and A. Bouzerdoum (1997). The evolutionary pre-processor: automatic feature extraction. *Genetic Programming 1997: Proceedings of 2nd Annual Conference,* 304-312. Morgan Kaufmann.

A. Teller and M. Veloso (1995). Program evolution for data mining. *International Journal of Expert Systems*, 3rd Quarter, 216-236.

D. Zongker, B. Punch and B. Rand (1998). *Lil-gp 1.01 - User´s Manual.* Department of Computer Science, Michigan State University.

Table 1: Parameter settings used in the experiments

EXP#	POPULATION SIZE	NUMBER OF GENERATIONS	MAXIMUM TREE SIZE	MAXIMUM TREE DEPTH	CROSSOVER PROBABILITY	REPRODUCTION PROBABILITY
1	5,000	50	65	12	0.85	0.15
2	3,500	50	65	15	0.75	0.25
3	2,500	30	60	12	0.70	0.30

Table 2: Predictive accuracy of individual rules

CLASS#	*TP*	*FP*	*FN*	*TN*	*SE*	*SP*	*SE* x *SP*
1	2	0	2	44	0,50	1,00	0,50
2	1	0	1	46	0,50	1,00	0,50
3	6	0	0	42	1,00	1,00	1,00
4	6	1	0	41	1,00	0,97	0,97
5	3	1	1	43	0,75	0,97	0,73
6	4	0	1	43	0,80	1,00	0,80
7	3	0	0	45	1,00	1,00	1,00
8	0	1	2	45	0,00	0,97	0,00
9	4	2	0	42	1,00	0,95	0,95
10	4	2	0	42	1,00	0,95	0,95
11	1	0	2	45	0,33	1,00	0,33
12	3	0	2	43	0,60	1,00	0,60
average	**3,08**	**0,58**	**0,91**	**43,41**	**0,706**	**0,984**	**0,694**