An Effective Microarray Data Classifier based on Gene Expression Programming*

Lei Duan, Changjie Tang, Liang Tang, Jie Zuo and Tianqing Zhang

School of Computer Science, Sichuan University
Chengdu 610065, China
{duanlei, tangchangjie}@cs.scu.edu.cn

Abstract

Applying data mining algorithms to microarray data analysis is an interesting and promising work. Gene Expression Programming (GEP) is a new development of evolution computation. GEP performs global search and discover the classification discriminant with high accuracy. However, it is undesirable to apply GEP on microarray classification directly, since the evolution efficiency of GEP is low when the number of attributes of training data is huge. To solve this problem, the main contributions of this paper include: (1) analyzing the difficulties of applying GEP to classifying microarray data directly, (2) designing a novel method to select GEP terminals from genes of microarray data, (3) proposing a method of constructing GEP classifier committee to improve the classification accuracy, (4) demonstrating the effectiveness of proposed algorithms by extensive experiments on several microarray data. Compared with some typical classification methods, the accuracy is increased as high as 10.46% in average.

1. Introduction

One application of DNA chip-based technology [1-3] is gene expression profiling. The result of gene expression profiling experiment is microarray data, which is widely applied in monitoring the effects of certain treatments, diseases, and developmental stages on gene expression. Applying data mining to microarray analysis is a promising and useful work. Basically, microarray data has following characteristics:

- The number of attributes (genes) in a microarray data is huge.
- The number of samples in a microarray data is smaller compared with the number of genes.
- All values in microarray data (gene expression values) are numeric values.

As microarray data is with huge dimensions, it is a challenging work of applying typical data mining algorithms to microarray data analysis directly. Researchers have proposed some novel methods [4, 5].

Gene Expression Programming (GEP) [6] has been widely used in data mining research fields, such as, symbolic regression [7], classification [8-10], and time series analysis [11]. Since GEP can perform global search and select several attributes of samples to generate classification rules at a time, it is desirable to use GEP for finding the classification discriminant.

As gene expression values of microarray data are numeric and GEP is efficient for numeric data calculation, it is promising to apply GEP to building classifier for microarray data. To the best of our knowledge, applying GEP on microarray data classification has not been explored systematically before, although some methods of using GEP to classification have been proposed. However, the evolution efficiency is low if we apply GEP to microarray data classification directly. Due to the individual length limit, not all terminals are selected in most cases. Moreover, the individual will be too complex to be understood if its length is large. Thus, it is necessary to design a method to select attributes that can be used to build effective GEP classifier.

The main contributions of this study include: (1) analyzing the difficulties of applying GEP to classifying microarray data, (2) designing a novel method to select GEP terminals from microarray genes, (3) proposing a method of constructing GEP classifier committee to improve the accuracy of classification

* This work was supported by the National Natural Science Foundation of China under grant No. 60773169, and the 11th Five Years Key Programs for Sci. &Tech. Development of China under grant No. 2006BAI05A01.

978-0-7695-3736-8/09 $25.00 © 2009 IEEE
DOI 10.1109/ICNC.2009.267

result, (4) demonstrating the effectiveness of our algorithms by extensive experiments. Compared with some typical classification methods, the accuracy is increased as high as 10.46% in average by our method.

2. Related work

The main process of GEP and its predecessors, GA and GP, are similar [6]. In GEP, the phenotype of each individual is a non-linear entity, while its genotype is encoded as simple strings of fixed length. The GEP algorithm begins with the random generation of a set of chromosomes (initial population). Then the chromosomes are expressed as its phenotype and the fitness of each individual is evaluated according to fitness function. The individuals are then selected according to fitness to reproduce with modification on genotype, leaving progeny with new traits. The new generated individuals are subjected to the evolution process: expression of the genomes, confrontation of the selection environment, and reproduction with modification. This procedure is repeated until a satisfactory solution is found, or a predetermined number of generations is reached. Then evolution stops and the best-so-far solution is returned. The details of GEP implementation can be referred in [6]. Compared with other evolution computing methods, GEP is more versatile and efficient. GEP has been widely used in data mining research fields [7-13].

The basic idea of applying GEP to classification is proposed in [6]. Specifically, given a two-class problem, suppose $dst(\ )$ is the discriminant discovered by GEP and t is the threshold value predetermined, then for each sample s_i GEP classifies s_i into class k or not according to the following rule.

Rule 1. IF $dst(s_i) > t$ THEN $s_i \in$ Class k ELSE $s_i \notin$ Class k.

One-against-all learning method is used to transform one n-class problem into n two-class problems [6, 8]. In [8], C. Zhou et al. proposed a way to incorporate both the rule consistency gain and completeness in the fitness function and applied GEP to classification. The performance study demonstrated that the results of GEP are more accurate than those of GP. W.R. Weinert and H.S. Lopes designed a classification rule discovery tool based on GEP in [9]. In [10], based on the basic GEP classification method, a dynamic way to calculate the classification threshold was proposed. In [12] and [13], M. H. Marghny and I. E. El-Semman gave methods to extract logical and fuzzy classification rules.

3. Classifying microarray data with GEP

The efficiency of GEP is low when the number of attributes is huge. As a result, special attribute (gene) selection way should be designed. Generally, we design the selection method from two aspects as follows.

- Single discriminative power: each selected attribute should aid GEP to evolve discriminant with high accuracy.
- Group discriminative power: the set of selected attributes should aid GEP to evolve discriminant with high accuracy.

3.1. Entropy-based attributes sorting

We evaluate the single discriminative power of each gene by the entropy-based method [14, 15]. All gene expression values in microarray data are numeric. We calculate the class information entropy of each gene in microarray data.

In this study, given a microarray data, all genes are evaluated by the entropy-based method and sorted according to information gains in descending order. It is worth to note that this entropy-based method evaluates genes separately, so genes with high information gain have high single discriminative powers.

3.2. GEP classifier terminal set selection

In GEP, terminals are selected from attributes of training data. Given an individual in GEP, the evolution efficiency will be low if the number of terminals is large. As GEP selects terminals randomly in evolution, it is hard for GEP to evolve good solutions in small number of generations if the number of terminals is large. Since the length of GEP individual is predetermined by user, we can avoid this situation by set a relative short individual length. Furthermore, we select a small subset of genes from microarray data as terminals in our study. The reasons are: (a) the efficiency of GEP will be decreased if all genes are put in the terminal set, (b) it is hard for user to understand long and complex discriminant, and (c) not all genes have good single discriminative powers.

One simple way to select genes is selecting genes with high information gains. However, we want to find the subset of genes that not only each gene in it has high discriminative power, but also the group of genes has high discriminative power. The method of evaluating group discriminative power adopted in our study is based on following observation.

Observation 1. Given a training data, if the differences among samples of different classes are obvious, it is easier for GEP to discover an accurate discriminant.

For example, suppose *dst*( ) is a discriminant which can classify all samples correctly. Then samples belong to different classes must not be the same. Intuitively, it is difficult for GEP to evolve a good discriminant if samples belong to different classes are similar. Thus, we want to select genes whose values are different in different classes. The gene expression values in microarray are numeric. It is meaningless to compare gene expression values directly, since few genes have the same expression value on different samples. To deal with this problem, we use the equal width binning method to convert gene expression values of each gene in microarray data into several intervals.

Given two genes, g_1 and g_2, we evaluate their group discriminative power by comparing their discretized values between positive class and negative class. Let p be the discretized values of g_1 and g_2 as a pair. Then, the pair share count of p is defined as follows.

Definition 1 (pair share count). Suppose p is a pair consists of two discretized values. Let n_p be the number of p in positive class and n_n be number of p in negative class. Then, the pair share count of p, denoted as $psc(p)$, is the minimal value of n_p and n_n.

Definition 2 (pair discriminative score). Suppose N is the number of samples. Given any two genes g_i and g_j, Let P be the set of pairs in g_i and g_j. Let $psc(P)$ be the sum of pair share counts of each pair in P. Then, the pair discriminative score of (g_i, g_j), denoted as dis-score(g_i, g_j), is dis-score$(g_i, g_j) = N - psc(P)$.

Example 1. Table 1 lists discretized values of a microarray data. We get all share pairs and calculate the pair discriminative scores of every two genes.

Table 1. Discretized gene expression values

g_1	g_2	g_3	g_4	class
1	3	5	3	positive
0	1	1	0	positive
0	5	2	2	positive
3	0	0	0	positive
0	2	1	1	negative
2	4	2	5	negative
0	0	2	2	negative
1	1	1	0	negative
5	0	0	3	negative

Based on Table 1, we list the share pairs and discriminative scores of each gene pair.

gene pair	share pair: counts	dis-score
(g_1, g_2)	\	9 – 0 = 9
(g_1, g_3)	(0, 1):1	9 – 1 = 8
(g_1, g_4)	(0, 2):1	9 – 1 = 8
(g_2, g_3)	(1, 1):1, (0, 0):1	9 – 2 = 7
(g_2, g_4)	(1, 0):1	9 – 1 = 8
(g_3, g_4)	(1, 0):1, (2, 2):1	9 – 2 = 7

We sort all genes in microarray data in information gain descending order. Then, we select GEP terminals one by one. The first terminal is the gene with the largest information gain. Then, we calculate the pair discriminative scores of each unselected gene and each selected gene. The gene has the largest average pair discriminative score will be chosen as the next terminal. This process is repeated until the number of selected genes equals to the predetermined value.

Algorithm 1 gives the pseudo codes to select GEP terminals from microarray data.

Algorithm 1 TermSelection(G, k)
Input: G: discretized data sorted by information gain
k: size of terminal set
Output: T: the index set of terminals
Begin
1. record the index of the first gene in G in T
2. while SIZE(T) < k do
3. for each gene in G, calculate its dis-score with selected genes in T
4. record the index of the gene that has the largest average dis-score in T
5. return T

End.

In Step 2, function SIZE(T) returns the number of indexes in T. It is easy to see that our method considers both single discriminative and group discriminative powers. In our study, the number of binning is 10.

3.3. Building GEP classifier committee

Building classifier committee is an effective way to improve the classification accuracy. Bagging [16] and Boosting [17] are effective methods to generate classifier committee by manipulating the training data. As the number of samples in microarray data is small, our method keeps the original training data unchanged but changes the learning phase of GEP terminal selection. In addition, the disadvantages of applying Bagging and Boosting methods to microarray data are analyzed in [5].

To build each discriminant in the committee, we use Algorithm 1 to select genes and evolve the discriminant by GEP. The discriminants are discovered one by one, and the selected genes are removed from the original training data each time. Algorithm 2 describes the process of discovering GEP classifier committee.

Algorithm 2 GEPClassifierComm(M, t, k)
Input: M: microarray data
t: the number of discriminants

k: size of terminal set

Output: C: classifier committee

Begin

1. sort genes in M according to information gain descending order
2. apply equal width binning method to each gene in M, let the discretized data be G
3. while SIZE(C) < t
4. T ← TermSelection(G, k)
5. evolve a GEP discriminant with ATTS(M, T)
6. add the generated discriminant into C
7. G ← G – ATTS(G, T)
8. return C

End.

Note that, function ATTS(M, T) in Step 5 returns attribute values of M whose indexes are in T, and ATTS(G, T) in Step 7 returns attribute values of G whose indexes are in T.

4. Experimental study

We test our algorithms on four microarray data[1]. Table 2 lists the characteristics of these data. All our proposed algorithms are implemented in Java. The experiments are performed on an Intel Pentium Dual 1.80 GHz (2 Cores) PC with 2G memory running Windows XP operating system.

Table 2. Data characteristics of 4 microarray data

Dataset	# objects in class 1	# objects in class 2	# attr.
Breast Cancer	44 (non-relapse)	34 (relapse)	24481
Central Nervous	21 (survivors)	39 (failures)	7129
Colon Cancer	22 (normal)	40 (cancer)	2000
Leukemia	11 (AML)	27 (ALL)	7129

In experiments, we set the binning number is 10, the size of terminal set is 3, the head length of GEP individual is 7, the population size is 200, the number of evolution generations is 500. The function set includes "+, –, *, /, sin, cos, log". We name our algorithm as HD-GEP. To validate the effectiveness of our gene selection method (Algorithm 1), we implement our algorithm with selecting top-k information gain genes as terminals each time. We name this algorithm as HD-GEP(IG). As discussed before, the efficiency of applying original GEP to microarray data classification is very low. We compare our method with Naïve Bayes, KNN(K=3), J48 and SMO which are available in WEKA [18]. Moreover, we compare our method with CS4 [5] which constructs a decision tree based classifier committee. The number of decision trees in the classifier committee is 5. All classification accuracies are evaluated in 5-fold-cross validation. For HD-GEP and HD-GEP(IG), the classification accuracy on each microarray data is the average value of 20 independent runs. We set the number of discriminants is 5, and each discriminant in classifier committee has equal vote weight.

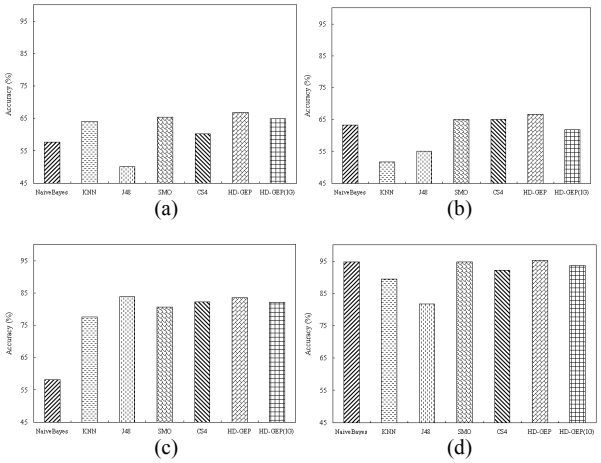

Figure 1. Accuracies on Breast Cancer (a), Central Nervous (b), Colon Cancer (c), Leukemia (d)

Figure 1 illustrates the classification accuracies of all methods. We can see that HD-GEP can always get desirable accuracy compared with other methods. The average accuracy got by HD-GEP is the highest on 3 microarray data: Breast Cancer, Central Nervous and Leukemia. J48 plays a little better than HD-GEP on Colon Tumor, but HD-GEP is better than other methods. We can see that the accuracy of HD-GEP is more stable than other methods, that is, HD-GEP can always get high accuracies on microarray data. Compared with Naïve Bayes, KNN, J48, SMO and CS4, the average accuracy on the four microarray data is increased by 9.61%, 7.40%, 10.46%, 1.63% and 3.16%, respectively. Moreover, the accuracy of HD-GEP on each microarray data is higher than that of HD-GEP(IG), so our terminal selection method is more effective compared with selecting top-k information gain genes as terminals directly.

As a classifier committee is composed of several discriminants, the number of discriminants is important. We record the accuracies under different numbers of discriminants. We find that more discriminants can increase the classification accuracy. However, the accuracy will be decreased if the number of discriminants is too large. Figure 2 illustrates the accuracies when the number of discriminants is 1, 3, 5, 7 and 9 on these microarray data. It is interesting to see that the highest accuracies are got when the number of discriminants is 5. In our method, discriminants in a classifier committee are generated one by one. Among

[1] http://datam.i2r.a-star.edu.sg/datasets/krbd/

the discriminants in a classifier committee, we find that the accuracies of discriminants generated later are lower than those generated earlier. As a result, the highest accuracy can be got when the size of classifier committee is 5 in our experiment. The size of terminal set is 3 in our experiment. More complicated discriminants with high accuracies may be discovered if more attributes are selected in the terminal set.

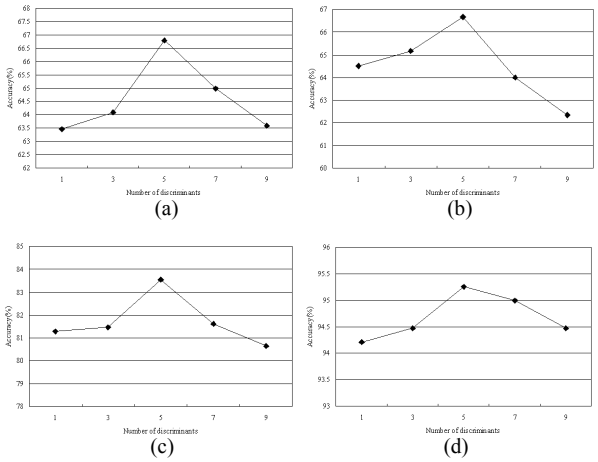

Figure 2. The classification accuracies under different number of discriminants on Breast Cancer (a), Central Nervous (b), Colon Cancer (c), Leukemia (d)

5. Conclusions and future work

The efficiency of evolution is low if apply GEP to microarray data classification directly, since the number of genes in microarray data is very large. To break this limit, we design an effective method to construct a classifier committee for microarray data based on GEP. The experiments show the effectiveness of our method. The future work includes analyzing the effect of assigning different weight values on different discriminants, and implementing the parallel version of our current method to improve the efficiency.

6. References

[1] M. Schena, D. Shalon, R. Davis and P. Brown. Quantitative Monitoring of Gene Expression Patterns with a Complementary DNA Microarray. *Science*, 1995, 270 (5235), P. 467-470.

[2] D. Lockhart, and *et al*. Expression Monitoring by Hybridization to High-density Oligonucleotide Arrays. *Nature Biotech.*, 1996, 14 (13), P. 1675-1680.

[3] V. Velculescu, L. Zhang, B. Bogelstein, and K. Kinzler. Serial Analysis of Gene Expression. *Science*, 1995, 270 (5235), P. 484-487.

[4] F. Pan, G. Cong, A.K.H. Tung, J. Yang, M.J. Zaki. CARPENTER: Finding Closed Patterns in Long Biological Datasets. In: *Proc. of SIGKDD 2003*, P. 637-642.

[5] J. Li, H. Liu. Ensembles of Cascading Trees. In: *Proc. of the 3rd IEEE Int' l Conf. on Data Mining (2003)*, P. 585-588.

[6] C. Ferreira. Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence. Angra do Heroismo, Portugal, 2002.

[7] H.S. Lopes and W. R. Weinert. EGIPSYS: An Enhanced Gene Expression Programming Approach for Symbolic Regression Problems. *Int'l Journal of Applied Mathematics and Computer Science*, 14 (3), P. 375-384.

[8] C. Zhou, W. Xiao, T.M. Tirpak and P.C. Nelson. Evolution Accurate and Compact Classification Rules with Gene Expression Programming. *IEEE Trans. on Evolutionary Computation*. Vol. 7, NO.6, 2003, P. 519-531.

[9] W.R. Weinert, and H.S. Lopes. GEPCLASS: A Classification Rule Discovery Tool Using Gene Expression Programming. In *Proc. of ADMA (2006)*. P. 871-880.

[10] L. Duan, C. Tang, T. Zhang, *et al*. Distance Guided Classification with Gene Expression Programming. In: *Proc. of ADMA (2006)*. P. 239-246.

[11] J. Zuo, C. Tang, C. Li, *et al*. Time Series Prediction based on Gene Expression Programming. In: *Proc. of WAIM (2004)*. P. 55-64.

[12] M. H. Marghny and I. E. El-Semman. Extracting Logical Classification Rules with Gene Expression Programming: Microarray Case Study. In: *Proc. of AIML (2005)*, Cairo, Egypt, 2005.

[13] M. H. Marghny and I. E. El-Semman. Extracting Fuzzy Classification Rules with Gene Expression Programming. In: *Proc. of AIML (2005)*, Cairo, Egypt, 2005.

[14] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In: *Proc. of ICML (1995)*. P. 194-202.

[15] U. Fayyad. and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In: *Proc. of IJCAI (1993)*. P. 1022-1029.

[16] L. Breiman. Bagging Predictors. *Machine Learning*, 24, P. 123-140, 1996.

[17] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In L. Saitta, editor, *Machine Learning: Proceedings of the Thirteenth International Conference*, P. 148-156, Bari, Italy, July 1996. Morgan Kaufmann.

[18] I. H. Witten, E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.