2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Classification of Gene Expression Data by Majority Voting Genetic Programming Classifier

Topon Kumar Paul, Yoshihiko Hasegawa, and Hitoshi Iba

Abstract— **Recently, genetic programming (GP) has been applied to the classification of gene expression data. In its typical implementation, using training data, a single rule or a single set of rules is evolved with GP, and then it is applied to test data to get generalized test accuracy. However, in most cases, the generalized test accuracy is not higher. In this paper, we propose a majority voting technique for prediction of the labels of test samples. Instead of a single rule or a single set of rules, we evolve multiple rules with GP and then apply those rules to test samples to determine their labels by using the majority voting technique. We demonstrate the effectiveness of our proposed method by performing different types of experiments on two microarray data sets.**

I. Introduction

Recently, many researchers are investigating the feasibility of different computational methods for gene expression based classification of patient samples. The hypothesis behind this gene expression based classification is that gene expression levels are affected by a large number of environmental factors, including temperature, stress, light, and other signals, that lead to changes in the levels of hormones and other signaling substances, and many or all human diseases may be accompanied by specific changes in the expression levels of some genes [1]. One widely used technique in cancer research to predict the classes of unknown samples is clustering, in which patient samples with similar clinical records are grouped together, and it is used with or without gene selection [2] [3] [4] [5] [6] [7] [8]. In addition to clustering, different machine learning classifiers like support vector machine (SVM) [9], k-nearest neighbor classifier [10], etc have been applied to the classification of gene expression data [7] [8] [11] [12] [13] [14]. For gene selection, both filter and wrapper approaches [15] have been used. Most of the filter approaches use rank based methods like signal-to-noise ratio (SNR) [6] to select some discriminative genes. Wrapper approaches are widely used for selection of genes using evolutionary computation methods [16] [12] [17] [18] [19] [20]. A comparative study on multicategory classification methods for gene expression data can be found in [21]. The main disadvantage of the above methods is that it is very difficult to determine an optimal pairing of gene selection algorithms and classifiers.

Recently, genetic programming [22], an evolutionary computation method, has been applied to the classification of gene expression data [23] [24] [25]. The main advantage of GP is that it can act as a classifier as well as a gene selection algorithm. In its typical implementation, a training set of gene expression data of patient-samples are presented to GP to evolve a boolean or an arithmetic expression of genes describing whether a given sample belongs to a given class or not. Then the evolved best rule (s) is (are) applied to the test samples to get the generalized accuracy on unknown samples. Unlike SVM and other classifiers, the use of GP as a classifier is transparent in the sense that the mechanism used to classify patient samples is available for inspection [25].

However, the potential challenge for genetic programming in classification of microarray data is that it has to search two large spaces of functions and genes simultaneously to find an optimal solution. In most cases, the evolved single rules or sets of rules produce very poor classification accuracy on the test samples. To overcome this limitation of genetic programming, we propose a majority voting technique for prediction of the class of a test sample. We call this method *majority voting genetic programming classifier* (MVGPC). The motivation behind this is that a group of rules can be more accurate than the best member of the group [26]. In its typical implementation, we evolve multiple rules in different GP runs, apply them one by one on a test sample and count their votes in favor of a particular class. Then the sample is assigned to the class that gets the highest number of votes in favor of it. However, the success of majority voting depends on the number of members in a voting group. Here we investigate the number of members in a majority voting group that gives the best results.

We apply MVGPC to two microarray data sets and perform experiments under different conditions. Our results suggest that MVGPC is very effective in classification of gene expression data and the best results are obtained when the number of members in a voting group is equal to the number of samples in the training data.

II. Notation and terms

We use the term *rule* to mean an individual in a population. The term *experiment* in the figures of this paper is used to mean a collection of runs needed to generate all the members of a voting group. Y is used to mean a test sample,

Topon Kumar Paul is with the Graduate School of Frontier Sciences, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa-shi, Chiba 277-8561, Japan (phone: +81-47-136-3873; fax: +81-47-136-3873; email: topon@iba.k.u-tokyo.ac.jp).

Yoshihiko Hasegawa is with the Graduate School of Frontier Sciences, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa-shi, Chiba 277-8561, Japan (phone: +81-47-136-3873; fax: +81-47-136-3873; email: hasegawa@iba.k.u-tokyo.ac.jp).

Hitoshi Iba is with the Graduate School of Frontier Sciences, The University of Tokyo, Kashiwanoha 5-1-5, Kashiwa-shi, Chiba 277-8561, Japan (phone: +81-3-5841-7424; fax: +81-3-5841-6070; email: iba@iba.k.u-tokyo.ac.jp).

which is a real-valued vector of gene expressions. The serial number of a gene in a microarray data set is indicated by 'X' followed by a number. However, during writing of biological significance of a gene, we have used a notation like 'X1234 (ID1) [NM_000123]' where X1234 is the serial number of the gene in the microarray data set, ID1 is the real name of the gene, and [NM_000123] is the GenBank [27] accession number of the mRNA sequences of that gene. We use c to mean the number of categories of samples in a microarray data set and v to denote the number of members in a majority voting group. Other terms and notations are described in places of their uses.

III. Methods

A. Genetic programming

Genetic programming (GP) [22] is an extension of the genetic algorithm in which genetic population consists of computer programs. Genetic programming starts with a population of randomly generated computer programs composed of functions and variables. The population of programs is then bred over many generations in a domain independent way using the Darwinian principle of survival of the fittest and an analog of the naturally-occurring genetic operation of crossover. The crossover operation is designed to create syntactically valid offspring programs from parents that are probabilistically selected based on their fitness at solving the problem at hand [28]. In gene expressions based classification, the programs in a GP population are rules of classifications consisting of functions and variables corresponding to the genes of a microarray data set. A generic example of these rules might look as follows:

IF $(2 * X2474 - X1265/X1223) \geq 0$
THEN 'Cancerous' ELSE 'Normal'

where $X2474$, $X1265$ and $X1223$ correspond to the expression levels of genes 2474, 1265 and 1223, respectively. From now on, we will use the term 'individual' to mean a classification rule. Genetic programming breeds a population of individuals to solve the problem of classification by executing the following steps:

1) Generate initial population of random compositions of functions and terminal sets (genes).
2) Execute each individual in the population and assign it a fitness.
3) While termination criteria is not met do the following sub steps:
    a) Create new offspring by repeatedly applying the following two operations to the parents that are selected from the population with a probability based on fitness:
        i) Copy a selected parent to the new population without any modification.
        ii) Create two offspring for the new population by genetically recombining randomly chosen parts of two selected parents.
    b) Execute each individual in the new population and assign it a fitness.

After termination of the algorithm, the best individual in the population is the solution or sub-solution to the problem. In classification of gene expression data, genetic programming acts as a classifier as well as a gene selection algorithm. There are many parameters like population size, maximum depth of a rule, crossover depth, probability of crossing over, etc. associated with genetic programming. Detailed descriptions on GP can be found in [22].

1) Components of a rule: Each rule in a population consists of randomly chosen functions and genes. Each gene in the rule is represented by an 'X' followed by the gene number. For example, $X1314$ represents gene 1314 of a data set. As functions, arithmetic and/or boolean functions can used for evolution of classification rules. For a binary classification problem, the class of a sample is determined as follows:

Boolean output:

IF (rule) THEN 'Class A' ELSE 'Class B'.

Real-valued output:

IF $(rule \geq 0)$ THEN 'Class A' ELSE 'Class B'.

During writing of a computer program for genetic programming, we have to choose function set depending on our targeted output. If we want Boolean outputs, we can consider a set of functions consisting of either arithmetic and Boolean functions like $\{+, -, *, /, sqr, sqrt, exp, and, or, not, >, >=, <, <=, =\}$ or only Boolean functions like $\{and, or, not, xor, >, >=, <, <=, =\}$. If our targeted output is real, we consider only arithmetic functions like $\{+, -, *, /, sqr, sqrt, ln, exp, power, sin, cos, tan\}$.

2) Evaluation of a rule: The success of an evolutionary computation method is very much dependent on the fitness function used to measure the goodness of an individual. For classification problems, the accuracy or the error rate of a predicting program can be used as a fitness measure; however, these methods may not get the optimum fitness. Matthews [29] proposed correlation between the prediction and the observed reality as the measure of raw fitness of a predicting program. For a binary classification problem, the correlation (C) is calculated as follows:

$$C = \frac{N_{tp}N_{tn} - N_{fp}N_{fn}}{\sqrt{(N_{tn}+N_{fn})(N_{tn}+N_{fp})(N_{tp}+N_{fn})(N_{tp}+N_{fp})}} \quad (1)$$

where N_{tp}, N_{tn}, N_{fp} and N_{fn} are the number of true positives, true negatives, false positives and false negatives, respectively. When the denominator of equation (1) is 0, C is set to 0. The standardized fitness of a rule is calculated as follows:

$$fitness(rule) = \frac{1+C}{2}. \quad (2)$$

Since C ranges between -1.0 and +1.0, the standardized fitness ranges between 0.0 and +1.0, the higher values being better and 1.0 being the best. The ultimate objective of GP is to find a rule that can classify all the samples correctly and thus has fitness=1.0. During execution of a rule on a sample, we take precautions so that the two functions 'sqrt' and '/' do not produce undefined results. In the case of undefined results, we treat them as follows: $\frac{x}{0} = 1$, and $\sqrt{x} = 0$ if $x < 0$. Note that after adjustment, $\sqrt{(x)^2} \neq (\sqrt{x})^2 \neq x$. For example, if $x = -3$, then $\sqrt{(x)^2} = 3$ while $(\sqrt{x})^2 = 0$. Similarly, $z * (x/y) \neq (z * x)/y$; if $y = 0$, then $z * (x/y) = z$ while $(z * x)/y = 1$.

3) Evolution of multiple rules for a multiclass problem: For classification of multicategory samples, we consider one-vs-rest approach. In this approach, if there are c types of samples in a microarray data set, we evolve c classification rules in c GP runs. During evolution of a rule i, the samples of class i are treated as positive; other samples as negative and the fitness is calculated employing equation (2). Suppose the output of a rule i on a test sample Y is $O(Y)$. Then the measures: true positive (TP), true negative (TN), false positive (FP) and false negative (FN) for the fitness of rule i are determined as follows:

IF ($O(Y) \geq 0$) AND (CLASS(Y)=i) THEN TP;
IF ($O(Y) < 0$) AND (CLASS(Y)$\neq i$) THEN TN;
IF ($O(Y) \geq 0$) AND (CLASS(Y)$\neq i$) THEN FP;
IF ($O(Y) < 0$) AND (CLASS(Y)=i) THEN FN.

In general case (without majority voting), the sample Y is correctly classified when only one rule has positive output on it. If more than one rule fit Y, it is treated as a misclassified sample.

B. Class prediction through majority voting

Majority voting technique is widely used in different forms in pattern recognition and classification. For example, in the kNN classifier, the class of a test sample is determined by the majority voting of k nearest neighbors of that sample. In multiclass classification using all-pairs (one-vs-one) approach, the majority voting technique is also used. In another form, when multiple classifiers like SVM, kNN, etc are combined to predict the label of a sample, majority voting is widely used. However, our approach is a little bit different from these methods.

In binary classification, we run genetic programming v times to generate the v best rules for a voting group. Then we execute these rules on the test samples one by one and predict their classes using majority voting. For a multiclass problem, we generate v rules for each category of samples in v runs. By executing a rule on a test sample, if we get a positive output, vote in favor of that class (positive vote) is increased by one; otherwise, the negative vote for that class is increased by one. Then the class is predicted as follows:

$$Class(Y) = \max_i \{r_1, r_2, \dots, r_c\} \tag{3}$$

where c is the number of types of samples in the data set, r_i is the ratio of positive and negative votes for class i. If more than one rule have the same ratio of positive and negative votes, the class corresponding to the lower index will be the predicted class for Y. However, if all the ratios are 0s, the sample is treated as misclassified. Let us give an example. Suppose a microarray data set contains three categories of samples: A, B and C, and we are trying to predict the class of a test sample Y by majority voting with $v = 7$. For each of these three categories, we generate 7 rules in 7 GP runs. If the number of positive and negative votes for the three classes are {0, 4, 5} and {7, 3, 2}, respectively, the predicted class of the test sample will be C since it has the highest ratio (=$\frac{5}{2}$) of the positive and negative votes.

C. Leave-one-out-cross-validation in GP

Since the number of available training samples in a microarray data set is usually very small compared to the number of attributes (genes), leave-one-out-cross-validation (LOOCV) technique [30] is usually used to calculate the training accuracy. In LOOCV, one sample from the training set is excluded, and the rest of the training samples is used to build the classifier. Then, the classifier is used to predict the class of the left out one, and this is repeated for each sample in the training set. The LOOCV estimate of accuracy is the overall number of correct classifications, divided by the number of samples in the training set. Thereafter, the final classifier is built using all the training samples, and the classes of the test samples are predicted one by one using that classifier. Note here that the data corresponding to selected genes remain the same during learning of the classifier using the LOOCV technique. Conversely, in genetic programming, different rules may evolve in different iterations of the LOOCV technique, and therefore we cannot calculate the LOOCV accuracy of a particular rule. Nonetheless, in MVGPC, we have used LOOCV in the following way:

- Generate N(=number of training samples) rules in N GP runs.
- In each run i, leave sample i for validation and use the remaining $(N - 1)$ samples as training data. If the evolved best rule can correctly classify the left out one, add this best rule to the voting group.
- Apply majority voting on the test data using the members of the voting group.

Note here that the number of members in a voting group may be smaller than N.

IV. Experiments

A. Microarray data sets

For our experiments, we chose two benchmark data sets: brain cancer [7], and breast cancer [31]. The summary of these data sets are shown in table I. The preprocessed data of breast cancer are available from the link provided below while the brain cancer data set is preprocessed by us using the technique described in [16].

The brain cancer data set contains expression levels of 12625 genes of 50 gliomas samples: 28 glioblastomas (GBM) and 22 anaplastic oligodendrogliomas (AO) divided into two subsets of classic and non-classic gliomas. The complete sets of data are available at `http://www-genome.`

TABLE I

MICROARRAY DATA SETS USED IN THE EXPERIMENTS. TRS=NUMBER OF TRAINING SAMPLES; TES=NUMBER OF TEST SAMPLES

Data set	#Genes	#Classes	TrS	TeS
Brain cancer	4434	2	28	22
Breast cancer	3226	3	17	5

wi.mit.edu/cancer/pub/glioma. After downloading the classic and non-classic data sets from the website, we merged them to get a single file and then applied preprocessing on it. After preprocessing of the raw data, only 4434 genes were left. Then this data set is divided into training and test subsets containing 28 and 22 samples, respectively. The training subset consists of 14 glioblastomas and 14 anaplastic oligodendrogliomas samples; the test subset consists of 14 glioblastomas and 8 anaplastic oligodendrogliomas samples.

The breast cancer data set contains 22 cDNA microarrays, each representing 5361 genes based on biopsy specimens of primary breast tumors of 7 patients with germ-line mutations of BRCA1, 8 patients with germ-line mutations of BRCA2, and 7 with sporadic cases. After preprocessing, only 3226 genes were left. The preprocessed data set is available at http://research.nhgri.nih.gov/microarray/NEJM_Supplement/. We divide this data set into two mutually exclusive training and test subsets containing 17 and 5 samples, respectively. The numbers of BRAC1, BRAC2 and sporadic samples into training and test subsets are {5, 6, 6} and {2, 2, 1}, respectively. Notice here that one sample, which is labeled as 'Sporadic/Meth.BRCA1' in the original data set, was treated as 'BRAC1' type in our experiments.

B. Values of different parameters

The values of different genetic programming parameters were: population size=4000; maximum depth (size) of a rule=100; maximum number of generations in a run=100; maximum crossover depth=7; maximum initial depth=6; crossover probability=0.9, and reproduction probability=0.1. By some preliminary test runs, we observed that increasing either the population size or the maximum number of generations did not improve classification accuracies. The initial population of each run was generated using the ramped half-and-half method [22]. We used Koza's greedy over selection for choosing mating pairs for crossing over and elitism so that the best found rule of a population survived for the next generation. In each run, the algorithm terminated when either all the training samples were correctly classified or the maximum number of generations had passed. As a set of functions, we used only arithmetic functions: $\{+, -, *, /, sqr, sqrt\}$. However, if more functions (especially complementary functions like $\ln$ and $\exp$) are used or the depth of a rule is increased, the evolved rules may be more complex with little or no improvement in accuracy. By performing some preliminary trial runs on the training data of the three data sets using more functions, we observed these phenomena. For example, we found some rules containing expressions like $\ln \exp(X1234)$ or $\exp \ln(X1234)$, which turns out to be a simple $X1234$.

To compare the classification accuracy of MVGPC, we performed experiments using the kNN classifier with RPMBGA [16]. The kNN classifier is widely used by the bioinformatics community in gene expression based classification and is easy to implement. Moreover, the kNN classifier uses the majority voting technique to predict the class of a sample. We chose RPMBGA because it is an evolutionary computation method like GP and better than traditional genetic algorithm in gene expression based classification of cancer data [16]. Therefore, the comparison of GP and RPMBGA on the data sets is convincing. The values of different parameters of RPMBGA were the same as the values used in [16].

C. Results

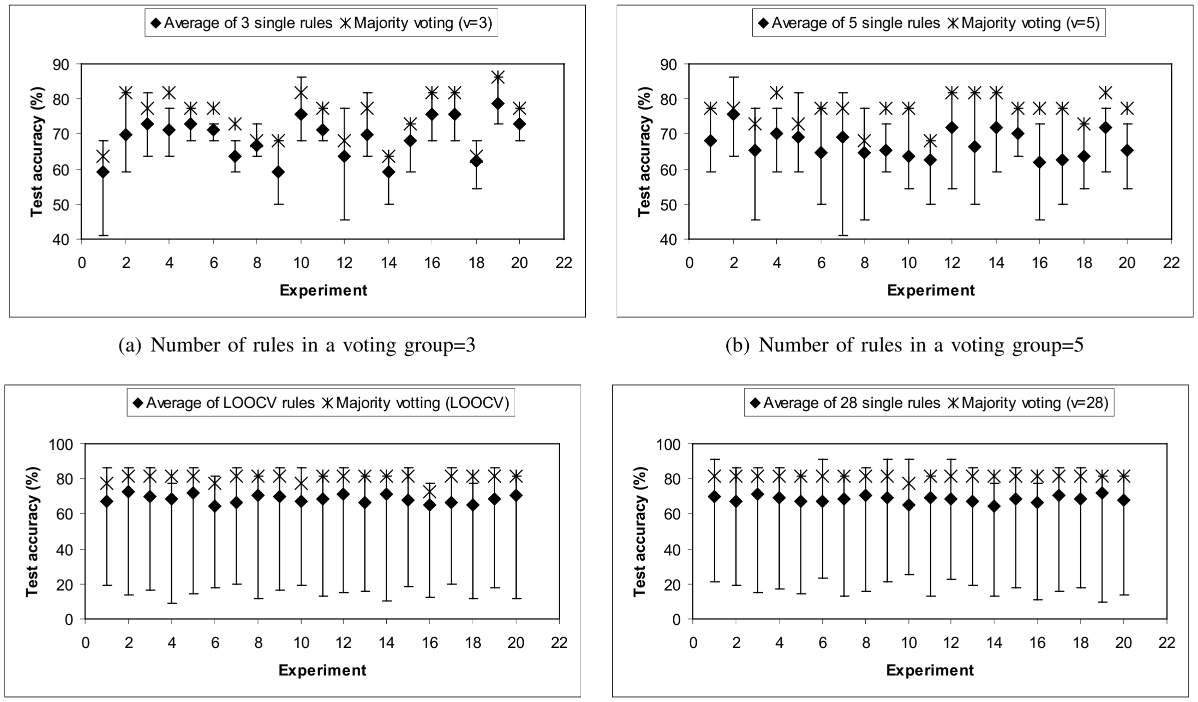
1) Test accuracy: First, we performed different experiments on the brain cancer data with different number of rules in a voting group. Since the brain cancer data set contains two types of samples (GBM vs AO), we started with $v = 3$; the reason behind this value is that the number of voting members in the majority voting technique should be greater than the number of types of samples. Then we did the same with $v = 5$ and $v = 28$. Moreover, we performed different experiments with the leave-one-out-cross-validation technique described above. Our findings are summarized in figure 1. In the figures, the average accuracy stands for the average test accuracy acquired by the v rules of a voting group, and the majority voting accuracy stands for the test accuracy obtained by those v rules using the majority voting technique. In addition to the average accuracy, we have shown the maximum and the minimum test accuracies of the v rules in a group by the corresponding error bars.

From the results of the brain cancer data, we found that the majority voting accuracy was better than the corresponding average accuracy in all cases; the best results were obtained when the number of rules in a voting group was equal to the number of the training samples. In that case, out of 20 experiments, MVGPC obtained 81.82% test accuracy in 19 experiments. Interestingly, in each case, the four test samples that were misclassified by MVGPC were the same: sample# 22, 28, 35 and 41. The five test samples that were misclassified by MVGPC in one experiment also included these four samples; the fifth misclassified sample was sample# 26 (the numbers of positive and negative votes for this sample were 14 each). Out of 560 (=28*20) rules, we found only four single rules that can classify 20 test samples out of 22 correctly. One of them is as follows:

- IF $(SQRT((X3739 - X3947) * (X895 - X3600)) + ((SQRT(X664) - (X1660/X2475) + X1871)/(X664 + X612 - X895 + 2 * X3600)) * SQRT(X1615)) \geq 0$ THEN 'GBM' ELSE 'AO'.

This rule fails to classify the two test samples: 35 and 41.

For the breast cancer data, we performed experiments with v=5, 7 and 17. The results are shown in figure 2. In all cases, the majority voting accuracy was much better than



(a) Number of rules in a voting group=3

(b) Number of rules in a voting group=5

(c) Majority voting with the rules of LOOCV

(d) Number of rules in a voting group=28

Fig. 1. Test accuracies on the brain cancer data under different conditions. For each voting group, in addition to the average accuracy, the maximum and the minimum accuracies are plotted on the graphs through error bars

the average accuracy of the v sets of 3 rules. Like the brain cancer data, the best results were obtained when the number of members in a voting group was equal to the number of training samples. Out of 20 experiments, we got 100% test accuracy in 11 cases and in the remaining 9 cases, the test accuracy was 80%. However, as individual sets of rules, only one set among the 340 (=17*20) sets of rules could classify all the samples 100% accurately. This best set of 3 rules is as follows:

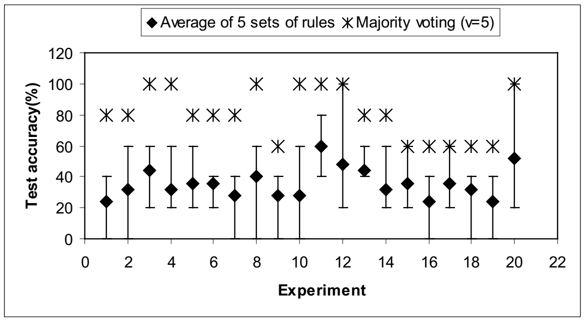
- IF $(SQR(X2942 * X3132 * (X176/X790)) - SQR(X1399 - X337 + (X821/X809))) \geq 0$ THEN 'BRAC1'.
- IF $(SQR(X982) - (((X3221 * X799)/SQR(X2382)) /(SQR(X647) + X1619 - X2454))) \geq 0$ THEN 'BRAC2'.
- IF $(((X1876 - X2126) * X863)/(X2307/X97) + SQR(X1644 + X1096) - SQR(((X1494/X3056) - SQR(X2463)) * (X1876 - X2126))) \geq 0$ THEN 'SPORADIC'.

To get comparative results, we performed experiments using the kNN classifier with RPMBGA (we denote it by kNN+). For the brain cancer data, we performed experiments with different number of nearest neighbors: k=3, 5 and 11. Similarly, for the breast cancer data, we performed experiments with k=5, 7 and 11. The best results for the brain and the breast cancer data were obtained with $k = 3$ and $k = 5$, respectively. The comparative results of MVGPC and the kNN classifier with RPMBGA are provided in table II. In the table, the results of MVGPC are the best ones that were obtained when v was set to the size of the training samples. The results of kNN+ are of 20 independent runs. In all criteria, the test accuracy of MVGPC was better than that of the kNN classifier with RPMBGA.

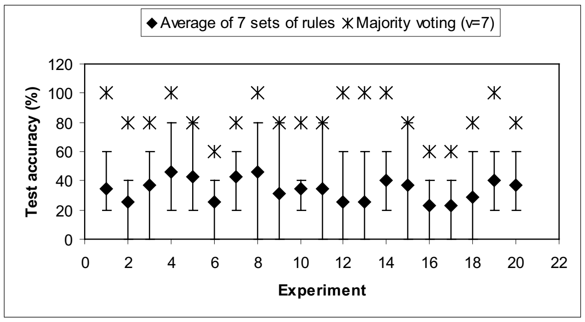
TABLE II
COMPARATIVE TEST ACCURACIES ON THE BRAIN CANCER AND THE BREAST CANCER DATA

Data Set	Method	Average	Max	Min	Median
Brain	MVGPC	81.59	81.82	77.27	81.82
cancer	kNN+	39.55	63.64	36.36	36.36
Breast	MVGPC	91.0	100.0	80.0	100.0
cancer	kNN+	68.0	100.0	0.0	70.0

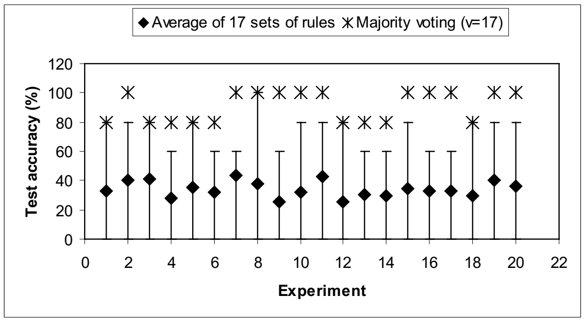
2) Evolution of fitness: When the number of training samples is smaller, the algorithm converges to the optimum very quickly; when the number of training samples is larger, it takes longer to reach the optimum. For the breast cancer data, GP converged to the optimum very quickly. However, for the brain cancer data, GP converged to the optimum fitness slowly and in some runs, it arrived at a local optima. In the 20 experiments on the brain cancer data, the number of rules in each group of 28 rules that could classify all the samples 100% accurately are {21, 19, 25, 27, 22, 21, 22, 23, 20, 24, 23, 21, 23, 24, 26, 19, 21, 21, 22, 24}. However, on the breast cancer data, all the rules (=340) in 20 experiments got 100% accuracy on the training data. In figure 3, we have shown the progression of the fitness of GP in a typical run.



(a) Number of rules in a voting group=5

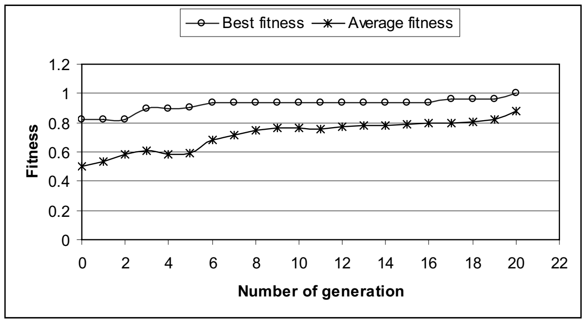


(b) Number of rules in a voting group=7

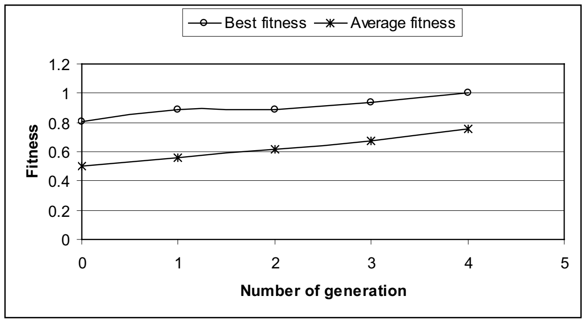


(c) Number of rules in a voting group=17

Fig. 2. Test accuracies on the breast cancer data under different conditions. For each voting group, in addition to the average accuracy, the maximum and the minimum accuracies are plotted on the graphs through error bars



(a) Brain cancer data.



(b) Breast cancer data.

Fig. 3. Progression of fitness in one typical run of GP

Note that only one rule is evolved in each run. For the breast cancer data, we need three separate runs to get three rules for the classification of three-category samples.

3) More frequently selected genes: The more frequently occurred genes in the rules of majority voting that produced the best test accuracies on the brain and the breast cancer data are shown in table III. (For the brain and the breast cancer data, these rules are of the experiments with v=28 and $v = 17$, respectively.) Out of 4434 genes, 3369 genes were included at least once in the 560 (=28*20) rules of the brain cancer data. Similarly, in the 1020 (=17*3*20) rules of the breast cancer data, 3079 genes were included at least once. However, the two genes: X2981 (IGFBP2) [X16302] and X3062 (ECE1) [Z35307] of the brain cancer data are of biological interest. IGFBP2 contributes to glioma progression in part by enhancing MMP-2 gene transcription and in turn tumor cell invasion [32]; ECE1 limits the Abeta accumulation in the mouse brain, which suggests it may have a role in Alzheimer's disease [33]. Among the more frequently occurring genes of the breast cancer data, the two genes: X860 (PLAU) [NM_002658] and X1479 (CCND1) [NM_053056] are known to be involved in breast cancer. PLAU and PAI-1 have roles in progression and recurrence of breast cancer [34]; CCND1 is overexpressed in human breast cancers and is required for oncogene-induced tumorigenesis [35].

V. Discussion

The task of classification of gene expression data faces many challenges due to the smaller number of available training samples compared to huge number of genes. Moreover, many of the genes in the microarray data set are redundant. These redundant genes sometimes affect the acquired classification accuracy by other genes negatively; sometimes they have no effect on the acquired accuracy. It sometimes may happen that the training accuracy of a classifier is 100% but its accuracy on test data is 0%. In our experiments, we observed this phenomenon on breast cancer data. Rather than using a single rule (or a single set of rules) to predict the class of a test sample, multiple rules can be combined for a prediction task. One of the simplest and most intuitive way of combining multiple classifiers is majority voting technique. However, in majority voting, the main point is what should be the size of the voting group. If the size is very smaller or very larger, the prediction accuracy may not improve. Our present experimental results and some preliminary results on other data sets suggest that the size of the voting group should be

TABLE III

MORE FREQUENTLY SELECTED GENES

Data set	Serial#	Accession#	Description	Freq.
Brain cancer	X2981	X16302	IGFBP2: insulin-like growth factor binding protein 2, 36kDa	69
	X3318	S37730	cds of IGFBP2	59
	X1489	X77956	ID1: inhibitor of DNA binding 1	32
	X1930	M80254	PPIF: peptidylprolyl isomerase F	29
	X3062	Z35307	ECE1: endothelin converting enzyme 1	23
Breast cancer	X860	NM_002658	PLAU: plasminogen activator, urokinase	25
	X1479	NM_053056	CCND1: cyclin D1	20
	X2152		ESTs	20
	X2804	NM_002709	PPP1CB: protein phosphatase 1, catalytic subunit, beta isoform	20
	X336	NM_005749	TOB1: transducer of ERBB2, 1	20

equal to the size of the training samples. In that case, the predictive accuracy is almost deterministic.

Nonetheless, some of the genes that are more frequently included in the best voting groups or in the best rules that produce the best classification accuracy have no direct links with the cancer being studied in this paper. It may be the case that these genes regulate the expressions of other genes that are responsible for the cancers. We need further investigation in this regard.

VI. CONCLUSIONS

In this paper, we propose the majority voting technique for the prediction of the class of a test sample by the rules of the genetic programming. By performing experiments on two microarray data sets, we have found that in all cases, the accuracy obtained with majority voting is better than the average accuracy of the rules in a voting group. Individually those rules classify the test samples very poorly but as a group of rules, they classify the samples very accurately. However, the best results were obtained when the number of members in a voting group was equal to the number of training samples. Moreover, some of the more frequently occurred genes in the evolved rules of MVGPC are the potential biomarkers of the types of cancers considered in this paper.

However, our proposed method did not get 100% accuracy in all experiments. In our future work, we want to investigate how we can improve the accuracy and apply our method to other benchmark microarray data sets. Preliminary results on other microarray data sets show that our method will also get better results on other data sets.

REFERENCES

[1] M. Schena, *DNA Microarrays*. New York, USA: Oxford University Press, 2000.

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, "Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of National Academy of Science, Cell Biology*, vol. 96, pp. 6745–6750, 1999.

[3] A. Alizadeh, M. Eisen, R. Davis, C. Ma, I. Lossos, A. Rosenwald, J. Boldrick, H. Sabet, T. Tran, X. Yu, J. Powell, L. Yang, G. Marti, T. Moore, J. J. Hudson, L. Lu, D. Lewis, R. Tibshirani, G. Sherlock, W. Chan, T. Greiner, D. Weisenburger, J. Armitage, R. Warnke, R. Levy, W. Wilson, M. Grever, J. Byrd, D. Botstein, P. Brown, and L. Staudt, "Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, no. 6781, pp. 503–511, 2000.

[4] A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering gene expression patterns," *Journal of Computational Biology*, vol. 6, pp. 281–297, 1999.

[5] M. B. Eisen, P. T. Spellman, P. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proceedings of the National Academy of Sciences*, vol. 95, pp. 14 863–14 868, 1998.

[6] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 15, pp. 531–537, 1999.

[7] C. Nutt, D. Mani, R. Betensky, P. Tamayo, J. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M. McLaughlin, T. T. Batchelor, P. Black, A. von Deimling, S. Pomeroy, T. Golub, and D. Louis, "Gene expression-based classification of malignant gliomas correlates better with survival than histological classification," *Cancer Research*, vol. 63, no. 7, pp. 1602–1607, 2003.

[8] D. Singh, P. Febbo, K. Ross, D. Jackson, J. Manola, C. Ladd, P. Tamayo, A. Renshaw, A. D'Amico, J. Richie, E. Lander, M. Loda, P. Kantoff, T. Golub, and W. Sellers, "Gene expression correlates of clinical prostate cancer behavior," *Cancer Cell*, March 2002. [Online]. Available: http://www.cancercell.org/cgi/content/full/1/2/203

[9] V. Vapnik, *The Nature of Statistical Learning Theory*. New York, USA: Springer-Verlag, 1995.

[10] B. Dasarathy, *Nearest Neighbor(NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, 1991.

[11] F. Pan, B. Wang, X. Hu, and W. Perrizo, "Comprehensive vertical sample-based knn/lsvm classification for gene expression analysis," *Journal of Biomedical Informatics*, vol. 37, no. 4, pp. 240–248, 2004.

[12] L. Li, D. M. Umbach, P. Terry, and J. A. Taylor, "Application of the GA/KNN method to SELDI proteomics data," *Bioinformatics*, vol. 20, no. 10, pp. 1638–1640, 2004.

[13] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machine," *Machine Learning*, vol. 46, no. 1-3, pp. 389–422, 2002.

[14] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. Lander, and T. Golub, "Multiclass cancer diagnosis using tumor gene expression signatures," *Proceedings of the National Academy of Science*, vol. 98, no. 26, pp. 15 149–15 154, 2001.

[15] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997. [Online]. Available: citeseer.nj.nec.com/article/kohavi97wrappers.html

[16] T. K. Paul and H. Iba, "Gene selection for classification of cancers

using probabilistic model building genetic algorithm," *BioSystems*, vol. 82, no. 3, pp. 208–225, 2005.

[17] C. Ooi and P. Tan, "Genetic algorithms applied to multiclass prediction for the analysis of gene expression data," *Bioinformatics*, vol. 19, pp. 37–44, 2003.

[18] T. K. Paul and H. Iba, "Extraction of informative genes from microarray data," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2005*. Washington DC, USA: ACM Press, 2005, pp. 453–460.

[19] ——, "Selection of the most useful subset of genes for gene expression-based classification," in *Proceedings of the 2004 Congress on Evolutionary Computation (CEC2004)*, Portland, Oregon, USA, 2004, pp. 2076–2083.

[20] ——, "Identification of informative genes for molecular classification using probabilistic model building genetic algorithm," in *Lecture Notes in Computer Science (LNCS) 3102*. Springer-Verlag, 2004, pp. 414–425.

[21] A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, "A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis," *Bioinformatics*, vol. 21, pp. 631–643, 2005.

[22] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.

[23] J.-H. Hong and S.-B. Cho, "Lymphoma cancer classification using genetic programming with SNR features," in *Proccedings of 7th European Conference, EuroGP 2004*, Coimbra, Portugal, 2004, pp. 78–88.

[24] W. Langdon and B. Buxton, "Genetic programming for mining dna chip data from cancer patients," *Genetic Programming and Evolvable Machines*, vol. 5, no. 3, 2004.

[25] J. A. Driscoll, B. Worzel, and D. MacLean, "Classification of gene expression data with genetic programming," in *Genetic Programming Theory and Practice*. Kluwer Academic Publishers, 2003, pp. 25–42.

[26] L. Kuncheva and C. Whitaker, "Measures of diversity in classifier ensembles and their relationships with the ensemble accuracy," *Machine Learning*, vol. 51, pp. 181–207, 2003.

[27] Entrez Gene, URL: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi.

[28] J. R. Koza and D. Andre, "Automatic discovery of protein motifs using genetic programming," in *Evolutionary Computation*, X. Yao, Ed. World Scientific, 1999, pp. 171–197.

[29] B. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochemica et Biophysica Acta.*, vol. 405, pp. 442–451, 1975.

[30] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[31] I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, O. Kallioniemi, B. Wilfond, A. Borg, and J. Trent, "Gene-expression profiles in hereditary breast cancer," *The New England Journal of Medicine*, vol. 344, no. 8, pp. 539–548, 2001.

[32] H. Wang, H. Wang, W. Shen, H. Huang, L. Hu, L. Ramdas, Y. Zhou, W. Liao, G. Fuller, and W. Zhang, "Insulin-like growth factor binding protein 2 enhances glioblastoma invasion by activating invasion-enhancing genes," *Cancer Research*, vol. 63, no. 15, pp. 4315–4321, 2003.

[33] E. Eckman, M. Watson, L. Marlow, K. Sambamurti, and C. B. Eckman, "Alzheimer's disease beta-amyloid peptide is increased in mice deficient in endothelin-converting enzyme," *J. Biol. Chem.*, vol. 278, no. 4, pp. 2081–2084, 2003.

[34] N. Harbeck, R. Kates, K. Gauger, A. Willems, M. Kiechle, V. Magdolen, and M. Schmitt, "Urokinase-type plasminogen activator (upa) and its inhibitor pai-i: novel tumor-derived factors with a high prognostic and predictive impact in breast cancer," *Thromb Haemost.*, vol. 91, no. 3, pp. 450–456, 2004.

[35] C. Wang, N. Pattabiraman, J. Zhou, M. Fu, T. Sakamaki, C. Albanese, Z. Li, K. Wu, J. Hulit, P. Neumeister, P. Novikoff, M. Brownlee, P. Scherer, J. Jones, K. Whitney, L. Donehower, E. Harris, T. Rohan, D. Johns, and R. Pestell, "Cyclin d1 repression of peroxisome proliferator-activated receptor gamma expression and transactivation," *Mol Cell Biol.*, vol. 23, no. 17, pp. 6159–6173, 2003.