A Genetic Approach for Gene Selection on Microarray Expression Data

Yong-Hyuk Kim[1], Su-Yeon Lee[2], and Byung-Ro Moon[1]

[1] School of Computer Science & Engineering, Seoul National University
Shillim-dong, Kwanak-gu, Seoul, 151-742 Korea
{yhdfly, moon}@soar.snu.ac.kr
[2] Program in Bioinformatics, Seoul National University
Shillim-dong, Kwanak-gu, Seoul, 151-742 Korea
suylee@soar.snu.ac.kr

Abstract. Microarrays allow simultaneous measurement of the expression levels of thousands of genes in cells under different physiological or disease states. Because the number of genes exceeds the number of samples, class prediction on microarray expression data leads to an extreme "curse of dimensionality" problem. A principal goal of these studies is to identify a subset of informative genes for class prediction to reduce the curse of dimensionality. We propose a novel genetic approach that selects a subset of predictive genes for classification on the basis of gene expression data. Our genetic algorithm maximizes correlation between genes and classes and minimizes intercorrelation among genes. We tested the genetic algorithm on leukemia data sets and obtained improved results over previous results.

1 Introduction

With the development of microarray technology, scientists can now examine multiple genome-wide gene expression patterns at the same time. Microarrays have been powerful experimental tools for extracting functional information from genome [5] [15]. As well as the diagnosis of disease, the classification of disease types is one of the most useful applications of microarrays. Recently, microarrays were used to profile the global gene expression patterns of normal and transformed human cells in several tumors, such as leukemia [11]. These researches may shed light on identifying biomarkers for cancer classification (molecular diagnosis). A wide-spread technique for microarray data analysis is clustering analysis [1] [3] [4] [10] [9] [13]. Clustering analysis groups genes that have correlated patterns of expression which can provide insight into gene-to-gene interactions and gene functions.

While microarrays have been extensively used in the gene expression profiling of tumor cells or tissues, successful applications of the microarray technology in cancer classification rely on data mining tools. This is because, among a lot of genes examined, only a fraction present distinct profiles for different classes of samples. Thus, it is critical to have computational tools that are capable of

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 346–355, 2004.

identifying a subset of informative genes embedded in a large dataset that is contaminated with high-dimensional noise.

Microarray data consist of a large number of genes (parameters) and relatively a small number of samples. It makes a "curse of dimensionality" problem; i.e., too many parameters for the data points. To reduce this problem, we try to identify a small subset of relevant genes. The major topic of this paper is to introduce an approach for gene selection with the help of a genetic algorithm.

Since typical microarray data consist of a large number of genes, many subsets of genes that distinguish between different classes of samples may exist. Our strategy is to find many such subsets and then evaluate the relative importance of genes for sample classification by examining inter-correlations of gene pairs in the subset. When selected genes were used for sample classification using a test set, samples were classified with accuracy. Other computational methods that select a subset of genes for sample classification were also developed [11] [2] [14] [12] [16]. The patterns of gene selection and the classification reliability of the selected genes using an independent test set are analyzed. We examine the sensitivity of gene selection results to the assignment of samples to the training set. We do this by dividing the dataset into a training set and test set in different ways, resulting in different training and test sets for the same data. Each training set is used to select a subset of genes.

In this paper, leukemia dataset is used as a benchmark dataset. We report the detailed analysis of the leukemia data using a genetic approach to find a subset of genes that can discriminate between acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). The results are compared with previous works.

The remainder of this paper is organized as follows. In Section 2, we summarize dataset and class predictor used in this paper. We propose a genetic approach for gene selection in Section 3. In Section 4, we present experimental results. Finally, we make our conclusions in Section 5.

2 Preliminaries

Recently, Golub *et al.* [11] proposed a method for selecting a subset of discriminative genes for sample classification. They successfully applied neighborhood analysis to identify a subset of genes that discriminates between AML and ALL, using a separation measure. The 50 genes that best distinguish AML from ALL in 38 training set samples were chosen as a class predictor that correctly classified 36 of the 38 training set samples. When these genes were subsequently used to predict the class of the test samples, 29 of the 34 samples were correctly classified with high confidence. In our implementation, four of the five samples were not classified (undecided) and one of the five samples was misclassified.

2.1 Dataset

The original leukemia dataset was downloaded from the web site[1]. The data contain the expression levels of 6,817 genes across 72 samples, of which 47 was classified as ALL and 25 as AML [11]. We divided the dataset into a training set (first 38 samples) and a test set (34 samples) following Golub *et al.* [11]. The training set was used to obtain a subset of genes that can discriminate between AML and ALL. The 50 most informative genes obtained using the training set were subsequently used in validation, to predict the classification of the test samples.

2.2 Class Predictor

Golub *et al.* [11] developed a procedure that uses a fixed subset of informative genes and makes a prediction based on the expression level of these genes in a new sample. Figure 1 shows the structure of their class predictor. Each informative gene casts a weighted vote for one of the classes, with the magnitude of each vote dependent on the expression level in the new sample and the degree of that gene's correlation with the class distinction. The votes are summed to determine the winning class as well as a prediction strength (PS), which is a measure that ranges from -1 to 1. The sample is assigned to the winning class if PS exceeds a predetermined threshold, and is considered undecided otherwise. We used the threshold of 0.3 following [11].

3 A Genetic Algorithm

We propose a genetic algorithm (GA) for gene selection to choose a good subset of genes. It selects genes based on the training set. It conducts a search for a good subset of genes using a correlation-based evaluation function. The search space with n genes has $2^n - 1$ elements if all nonempty subsets are considered. If the number of genes to be selected is predetermined, the optimal subset of size k can be found by enumerating and testing all possibilities, which requires $\binom{n}{k}$ tests. Then, this makes the problem intractable. Our GA provides an alternative search method to find a good subset with a predetermined size.

The dataset consists of 6,817 genes. If all the genes are considered as a candidate of informative genes, the problem size becomes intractable. So, we used the gene set filtered by the correlation ρ'. We considered three cases: $|\rho'| > 0.8$ (136 genes), $|\rho'| > 0.7$ (299 genes), and $|\rho'| > 0.5$ (980 genes).

The dataset is divided into two independent sets: the training set and the test set. Our GA runs on the training set until a termination criterion is satisfied and selects a predefined number of genes (50 genes in our experiments[2]). After our GA selects a subset of genes, the predictive model is tested on the test set.

[1] `http://www.genome.wi.mit.edu/MPR`

[2] To compare with the previous work [11] under the same condition, we fixed the number of genes to 50.

```
ClassPredictor(sample x = (x_1, x_2, ..., x_#genes))
{
    // x_i: expression level of gene i
    V_AML ← 0, V_ALL ← 0;
    for each informative gene g,
        μ_AML(g) ← mean expression levels of g for the samples in AML;
        μ_ALL(g) ← mean expression levels of g for the samples in ALL;
        σ_AML(g) ← SD expression levels of g for the samples in AML;
        σ_ALL(g) ← SD expression levels of g for the samples in ALL;
        ρ'(g, C) ← (μ_AML(g) − μ_ALL(g))/(σ_AML(g) + σ_ALL(g));
        v_g ← ρ'(g, C) · (x_i − (μ_AML(g) + μ_ALL(g))/2);
        if (v_g > 0) V_AML ← V_AML + v_g;
        else V_ALL ← V_ALL − v_g;
    PS ← (V_AML − V_ALL)/(V_AML + V_ALL);
    if (|PS| < threshold) return undecided;
    else if (PS > 0) return AML;
    else return ALL;
}
```

Fig. 1. The structure of class predictor [11]

3.1 Genetic Operators

The general structure of steady-state genetic algorithms is used in our GA.

- *Encoding*: In this problem, a chromosome is represented by binary encoding. A gene has value one if the gene belongs to the informative gene subset; otherwise, it has value zero.
- *Initialization*: We first create p subsets at random. The only constraint on a chromosome is that the number of 1's should be 50. We set the population size p to be 100.
- *Selection*: We assign to each chromosome in the population a fitness value calculated from its object value. We use the roulette-wheel-based *proportional selection* scheme.
- *Crossover and Mutation Operators*: A crossover operator creates a new offspring by combining parts of the two parents. In our experiments, we use one-point crossover and use element-swap mutation that swaps the values of a random pair of genes. After the crossover, an offspring may not satisfy the constraint. It then selects random points on the chromosome and changes the required number of 1's to 0's (or 0's to 1's). This adjustment also produces some mutation effect.
- *Replacement*: After generating an offspring and applying a local optimization on it, we replace a member of the population with the offspring. We use the replacement scheme of [6]. The offspring tries to first replace the more similar parent, measured in bitwise difference; if it fails, then it tries to replace the other parent (replacement is done only when the offspring is better than one

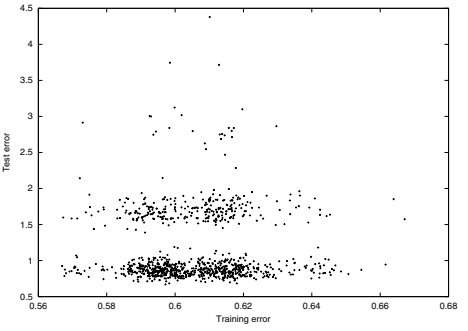


Fig. 2. Training error vs. test error: $|\rho'| > 0.8$

of the parents); if the offspring is worse than both parents, we replace the worst member of the population (Genitor-style replacement [18]).

– *Stopping Condition*: For stopping, we use the number of consecutive fails to replace one of the parents. We set the number to be 20 in our GA.

3.2 Evaluation Function

The error rate[3] in training samples can be considered as an evaluation function value for the GA. The predictivity for training samples is calculated by cross-validation (for details, see Section 4). Figure 2 shows the plotting of training error versus test error for 50-gene subsets randomly chosen from 136 genes with $|\rho'| > 0.8$. It is clear that a low training error does not mean a low test error. The training error may not be adequate as the evaluation function. If the GA minimizes the training error, overfitting in training samples may occur.

Our evaluation function is to find the genes that highly correlated with the class and less correlated with other genes in the subset of genes. Our GA minimizes the following object function.

$$objectfunction = \frac{1}{\overline{|\rho'(X,C)|}} + p \cdot \overline{|\rho(X,Y)|},$$

where $\rho'(x,C)$ means the correlation between the expression levels of gene x and the class distinction described in [11] (see Figure 1), p is the intercorrelation factor, and $\rho(x,y)$ is the Pearson correlation coefficient between gene x and gene y. In the fitness function, we use intercorrelation factor p. When p is zero, the

[3] In this paper, we calculated the error rate by the following formula.

$$\frac{1+e}{1-(u+e)/n} \cdot \overline{\frac{1}{2-|C-PS|}}$$

where e is the number of misclassified samples, u is the number of undecided samples, C is the class value ($C = 1$ for AML and $C = -1$ for ALL), and PS is the prediction strength given in Figure 1.

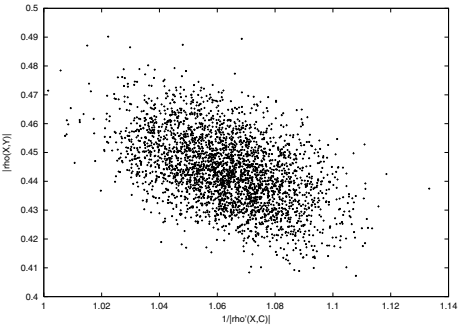


Fig. 3. Inverse correlation ($1/\overline{|\rho'(X,C)|}$) versus intercorrelation ($\overline{|\rho(X,Y)|}$): $|\rho'| > 0.8$

objective function is simply to find the genes that are highly correlated with the class. If p is positive, a gene subset less correlated with other genes in the set is preferred. Otherwise, a gene subset highly correlated with other genes in the set has a low object value. Figure 3 shows the relation between inverse correlation ($1/\overline{|\rho'(X,C)|}$) and intercorrelation ($\overline{|\rho(X,Y)|}$) for 50-gene subsets randomly chosen from 136 genes with $|\rho'| > 0.8$. It was observed that there is strongly negative correlation between the two values. In this paper, to make balance between the two values, we set the intercorrelation factor p to 2.

4 Experimental Results

The set of informative genes selected by Golub *et al.* [11] consists of the 25 genes closest to the class AML and the 25 genes closest to the class ALL. That is, in the view of the correlation ρ', the topmost 25 genes and bottommost 25 genes are chosen. In "Greedy," we choose the topmost 50 genes with respect to the value $|\rho'|$. We also compared the prediction results using 50 genes given by Li *et al.* [14], though they did not adopt the class predictor of [11]. "Random" means 50 genes randomly chosen in the given candidate gene set.

We denote our GA with intercorrelation factor $p = 2$ by dispersed-gene-based GA (DGA). To test the validity of our object function, we made additional experiments of GAs with other object functions. The GA denoted by biased-gene-based GA (BGA) has the object function with $p = -2$. Training-set-fitted GA (TGA) was designed to maximize the accuracy on training samples.

It is crucial to maximize the number of correctly classified samples. Especially, rather than to minimize the number of undecided samples, it is more important to minimize misclassified samples [11]. To test the validity of selected informative genes, we made experiments with two types of test procedure. First, we follow the test procedure of [11]. It consists of two steps. The accuracy is first tested by Leave-One-Out-Cross-Validation (LOOCV) on the training dataset. Briefly, one withholds a sample, builds a predictor based on the remaining samples, and predicts the class of withheld sample. The process is repeated for each

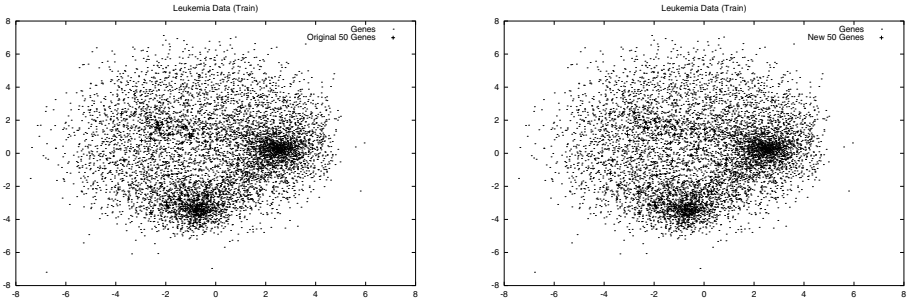
Table 1. Data Comparison in Given Data Samples

Method	Training data		Independent data	
	Undecided	Error	Undecided	Error
Golub *et al.* [11]	2	0	4	1
Greedy	1	0	6	1
Li *et al.* [14]	2	0	6	0
Random1	1.58	0.00	6.54	0.53
Random2	2.51	0.00	8.63	0.72
Random3	5.18	0.01	11.90	0.95
DGA1	0.1	0.0	4.1	**0.0**
DGA2	0.3	0.0	6.3	**0.0**
DGA3	0.0	0.0	5.5	**0.0**
BGA1	1.2	0.5	4.5	1.7
BGA2	2.6	0.6	5.4	1.4
BGA3	2.8	0.2	6.3	1.0
TGA1	0.3	0.0	6.1	1.0
TGA2	0.1	0.0	7.1	1.1
TGA3	0.2	0.0	6.7	1.0

Sampling 1. $|\rho'| > 0.8$ (136 genes).
Sampling 2. $|\rho'| > 0.7$ (299 genes).
Sampling 3. $|\rho'| > 0.5$ (980 genes).
Object function of DGA is $1/\overline{|\rho'(X,C)|} + 2 \cdot \overline{|\rho(X,Y)|}$.
Object function of BGA is $1/\overline{|\rho'(X,C)|} - 2 \cdot \overline{|\rho(X,Y)|}$.
Object function of TGA is $1/(2 - |C - PS|)$.
The results of GAs are averages over 10 runs.
The results of Random's are averages over 3,000 runs.

sample, and the cumulative error is calculated. Then, one builds a final predictor based on the training data set and assesses its accuracy on an independent test dataset. Table 1 shows its experimental results. The results of randomly sampled gene subsets (Random's) are average values from 3,000 runs. The more candidate genes were considered, the more errors were detected. This shows that the correlation ρ' is a good measure for evaluating genes' predictivity. However, Greedy, which chooses only genes with the highest ρ' values, performed worse than others. This suggests that an additive measure is needed to find a more informative gene subset. In case of DGA where the intercorrelation measure is also considered as a minimizing factor, it showed the best performance among all the tested methods. It had nearly zero undecided and error sample in LOOCV of training samples (almost 100% accuracy). Also, it showed no misclassification in test samples. In case of BGA which finds the gene subset highly correlated with other genes, since selected genes are biased in gene space, its results were much worse than those of DGA.

In training samples, each gene is considered to be a point in the 38-dimensional gene space. Figure 4 shows the two dimensional mapping of infor-



(A) Original 50 genes [11] (B) New 50 genes

Fig. 4. Plotting of informative gene subsets

mative gene subsets using Sammon's mapping[4] [17]. We used the inverse value of correlation, $\frac{1}{|\rho'|} - 1$, as the distance between each gene pair for mapping. We can see that the 50 genes of [11] (figure (a)) are clustered but the 50 genes selected by DGA (figure (b)) are well distributed in the gene space composed of 6,817 genes.

Next, the prediction test is repeated for a number of different bootstrap samples. Bootstrapping, which is introduced by Efron [7], is a well-known technique for estimating generalization of a predictive model based on resampling. In bootstrapping, a data sample of size n is uniformly taken from the original data of size n with replacement [8]. In our implementation, the (n-v) samples for dataset are drawn without replacement, where n is the number of samples in the dataset and v is the number of samples in the test set. There is no overlapping between the training and test sets for each bootstrap sample. Our gene selection model is trained by using the training set and its sensitivity are measured on the test set. According to the given factor of divided samples, we drew (72-34) samples. Table 2 shows the experimental results. The results are calculated based on 100 bootstrap samples. In bootstrapping, we only considered genes with $|\rho'| > 0.7$ as a candidate gene set. The average size of candidate gene sets was about 175. In training samples, DGA showed the best accuracy. Undecided samples in test samples were more in DGA than in Golub *et al.* [11] and Greedy. However, DGA showed the smallest misclassified rate in test samples.

[4] Sammon's mapping is a mapping technique for transforming a dataset from a high-dimensional (say, m-dimensional) input space onto a low-dimensional (say, d-dimensional) output space (with $d < m$). The basic idea is to arrange all the data points on a d-dimensional output space in such a way that minimizes the distortion of the relationship among data points. Sammon's mapping tries to preserve distances. This is achieved by minimizing an error criterion which penalizes the differences of distances between points in the input space and the output space.

Table 2. Data Comparison in Bootstrap Samples

	Training data		Independent data	
Method	Undecided Ave($\sigma/\sqrt{n}$)	Error Ave($\sigma/\sqrt{n}$)	Undecided Ave($\sigma/\sqrt{n}$)	Error Ave($\sigma/\sqrt{n}$)
Random	2.10(0.12)	0.12(0.03)	4.69(0.22)	0.61(0.09)
Golub *et al.* [11]	1.70(0.10)	0.04(0.02)	3.08(0.16)	0.48(0.06)
Greedy	1.77(0.11)	0.12(0.03)	3.08(0.14)	0.67(0.07)
DGA	0.93(0.09)	0.00(0.00)	3.60(0.19)	**0.29**(0.05)

Sampling in Random and DGA: $|\rho'| > 0.7$.
Average # of candidate genes = 174.69.
# of training samples = 38.
# of independent samples = 34.
Average over 100 datasets.

5 Discussion

As more genes were included, leading to the curse-of-dimensionality problem, the number of misclassified samples increased. This emphasizes that not all expression data are relevant to the distinction between ALL and AML. It is evident that not all genes are relevant to sample classification. Thus, the identification of informative genes is essential. The important issue is that microarray data consist of a large number of genes and a small number of samples, and, as a result, a great number of distinct and effective classifiers may exist for the same training set. Most of current literature methods seek a single subset of discriminative genes. Often, the informative genes identified for a given dataset vary from method to method. In conclusion, a number of methods have been developed for sample classification based on gene expression data. Our algorithm selected a good subset of genes and improved the predictive quality of the existing prediction model. As the quantitative aspect of the microarray technology is improved and computational methods that mine the resulting large dataset are further developed, this study will have a notable impact on biology and related areas.

Acknowledgments. This study was supported by a grant of the International Mobile Telecommunications 2000 R&D Project, Ministry of Information & Communication, Republic of Korea. This was also partly supported by grant No. (R01-2003-000-10879-0) from the Basic Research Program of the Korea Science and Engineering Foundation, and by Brain Korea 21 Project. The ICT at Seoul National University provided research facilities for this study.

References

1. 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 analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl Acad. Sci. USA*, 96:6745–6750, 1999.
2. A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. In *The Fourth International Conference on Computational Molecular Biology (RECOMB2000)*. ACM Press, New York, 2000.
3. A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *J. Comput. Biol.*, 6:281–297, 1999.
4. M. Bittner, P. Meltzer, and J. Trent. Data analysis and integration: of steps and arrows. *Nature Genetics*, 22:213–215, 1999.
5. P. O. Brown and Botstein D. Exploring the new world of the genome with DNA microarrays. *Nature Genetics*, 21:33–37, 1999.
6. T. N. Bui and B. R. Moon. Genetic algorithm and graph partitioning. *IEEE Trans. on Computers*, 45(7):841–855, 1996.
7. B. Efron. *The jacknife, the bootstrap, and other resampling plans*. Society for Industrial and Applied Methematics, 1982.
8. B. Efron and R. Tibshirani. *Cross-validation and the bootstrap: Estimating the error rate of a prediction rule*. Dept. of Statistics, Stanford University, 1995.
9. G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. *Proc. Natl Acad. Sci. USA*, 97:12079–12084, 2000.
10. G. Getz, E. Levine, E. Domany, and M. Q. Zhang. Superparamagnatic clustering of yeast gene expression profiles. *Physica A*, 279:457–464, 2000.
11. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
12. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.
13. E. Hartuv, A. O. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrach, and R. Shamir. An algorithm for clustering cDNA fingerprints. *Genomics*, 66:249–256, 2000.
14. L. Li, T. A. Darden, C. R. Weinberg, and L. G. Pedersen. Gene assessment and sample classification for gene expression data using a genetic algorithm/k-nearest neighbor method. *Combinatorial Chemistry & High Throughput Screening*, 4:727–739, 2001.
15. D. J. Lockhart and E. A. Winzeler. Genomics, gene expression and DNA arrays. *Nature*, 405:827–836, 2000.
16. H. Iba S. Ando. Artificial immune system for classification of gene expression data. In *Genetic and Evolutionary Compatation Conference*, pages 1926–1937, 2003.
17. J. W. Sammon, Jr. A non-linear mapping for data structure analysis. *IEEE Transactions on Computers*, 18:401–409, 1969.
18. D. Whitley and J. Kauth. Genitor: A different genetic algorithm. In *Rocky Mountain Conference on Artificial Intelligence*, pages 118–130, 1988.