Classifying Gene Expression Data using an Evolutionary Algorithm

Thanyaluk Jirapech-umpai

Master of Science

School of Informatics

University of Edinburgh

2004

Abstract

Since the advent of microarray technology, the large amount of gene expression data leads to statistical and analytical challenges. One challenge area in the studies of gene expression data is the classification of the expression dataset into correct classes. The dissertation is addressing multiclass classification which has been shown to be more difficult than the binary classification. The main difficulties in solving microarray classification are the availability of very small amount of samples compared to the number of genes in the sample and the extremely large search space of solutions. Moreover, the variation in microarray experiment also causes noise in the gene expression data. This makes classification task more difficult in order to be able discover the underlying pattern in noisy gene expression data.

The dissertation aims to implement and evaluate an evolutionary algorithm described in Deutsch (2003) for microarray multiclass classification. Starting with normalization on gene expression data to reduce the variance and noise in microarray data. Next step, some genes are used to build the initial gene pool. The evolutionary algorithm is implemented to explore the large search spaces to discover the best solution with optimal number of predictive genes in the initial gene pool. Performance of the solutions is evaluated using the k-nearest neighbour classifier to identify the class on training samples with the leave-one-out cross validation technique. Furthermore, the dissertation aims to evaluate the parameters that may affect the performance of the evolutionary algorithm: population size, feature size, and initial gene pools built by various ranking methods. Finally, the best parameters will be tested again using the 0.632 bootstrap estimation method to give effective performance.

Acknowledgements

Firstly, I would like to thank my supervisor, Dr. Stuart Aitken, for his guidance and advice in the research methodology, software development, and documentation. I also would like to thank my family for their support and an inspiration, and James Wilkie for his patience as a good listener for my ideas and the encouragement through the years.

Finally, I would like to thank the Royal Thai Government who provided financial support during my year of study.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Thanyaluk Jirapech-umpai*)

Table of Contents

Chapter 1

Introduction

1.1 Multiclass Classification for microarray experiments

Microarray technology has provided biologists with the ability to measure the expression levels of thousands of genes in a single experiment. The vast amount of raw gene expression data leads to statistical and analytical challenges. One challenge area in the studies of gene expression data is the classification of the expression dataset into correct classes. The goals of classification are to identify the differentially expressed genes that may be used to predict class membership for new samples. First, supervised classification identifies a set of genes that can differentiate different classes of samples by using the training dataset with known classes. Then, the selected set of discriminative genes, or predictive genes, is used to identify the class of unknown samples. This project is addressing multiclass classification which has been shown to be more difficult than the binary classification. The main difficulties in solving microarray classification are the availability of very small amount of samples compared to the number of genes in the sample and the extremely large search space of solutions. Moreover, the variation in microarray experiments can affect the gene expression levels measurement. It also causes a problem for the identification of predictive genes from noisy gene expression data.

There are two main procedures in gene expression data classification task: feature selection and classification. The feature selection method, or gene selector, identifies and selects the set of predictive genes (that are relevant for distinguishing classes) in order to reduce the number of genes to be considered as a feature of the sample for use in the classification task. Several gene selection methods have been developed to select these predictive genes, such as t-statistics, information gain, twoing rule, the ratio

of between-groups to within-groups sum of squares(BSS/WSS), Principle Component Analysis, and Genetic Algorithm(GA). Currently, the interesting issue in gene selector is the how to determine the optimum number of predictive genes for each dataset in order to gain better performance in multiclass classification. In the classification task, both supervised and unsupervised classifiers have been used in order to build classification model from the selected set of predictive genes. The dissertation will focus on only supervised classifiers which classify samples using training samples of known class. Many classifiers have been used for this task such as Fisher Linear Discrimination Analysis, Maximum Likelihood Discriminant Rules, Classification Tree, Support Vector Machine(SVM), K Nearest Neighbour(KNN), and the aggregating classifiers.

1.2 Evolutionary algorithms

In general, problem solving can be perceived as a search through a space of potential solutions. The task to find the best solution can be called optimisation process (Dasgupta and Michalewicz, 1997). Many search and optimisation techniques have been developed and used in the search problems such as random search, simulation-based optimization, simulated annealing, and Markov chain Monte Carlo.

Evolutionary algorithms(EAs) are also stochastic search and optimisation techniques which have been developed during the last 30 years. The evolutionary algorithms are based on the same principles of evolution in the biological world involving natural selection, and survival of the fittest. EAs provide the optimization techniques that differ from other traditional optimizations in that they involve a search through a population of solutions, not from a single point. Evolutionary algorithms can be divided into three main areas of research: Genetic Algorithms (GA) mainly developed by J.H. Holland in 1975, Evolution Strategies (ES) developed by I. Rechenberg and H.-P. Schewefel in 1981 and Evolutionary Programming. There are several variants of the different types of EAs but the basic structure of any evolutionary method is similar to each other. A common structure is shown in Figure 1.1.

```
Initialize the population of solution
Evaluate initial population
While (not termination-condition) do
begin
    Perform competitive selection
    Apply genetic operators to generate new solutions
    Evaluate solutions
end
```

Figure 1.1: A structure of Evolutionary Algorithms

1.3 Evoluationary algorithms in microarray multiclass classification

In microarray classification, the gene selection problem is an optimization problem, with a performance measure for each subset of genes to measure its ability to classify the samples into a correct category. The problem is to search through the space of gene subsets to identify the optimal or near optimal one with respect to the performance measure. Focusing on datasets gathered from microarray experiments, the number of genes is relatively large compare to the number of samples. Searching for the optimal gene subset in a large space is a difficult task. The evolutionary approach has been brought into this search area with the hope that it can improve the searching performance which will lead to the better performance in other machine learning techniques such as classification and clustering. The evolutionary algorithm uses a wrapper technique to integrate gene selection into a classification algorithm. Each subset of genes is evaluated using a learning algorithm in order to progressively generate new and better subsets.

Evolutionary algorithms(EAs) have been implemented in microarray classification in order to search the optimal or near-optimal solutions on complex and large spaces of possible solutions. EA is mostly used as a gene selector which is embedded into the classification task to search for the optimal genes set that gives high accuracy prediction. Several studies have implemented different types of EAs with different classifiers: KNN, Neural Network , and Maximum Likelihood(MLHD). Furthermore, EAs have been applied in the aggregated classification method to search for the optimal ensemble of feature selection-classifier pairs (Park and Cho, 2003). The report showed that

the EAs improved the classification result by giving the good combination between gene selectors and classifiers.

1.4 Aims

The aim of this dissertation is to implement and evaluate an evolutionary algorithm described in Deutsch (2003) for microarray multiclass classification. Two real datasets are used to test the performance of the algorithm: the leukemia and NCI60 dataset. Before any data analysis the data, noisy gene expression data is transformed and normalized. Next step is to build initial gene pool using the ranking methods in the RankGene software. The evolutionary algorithm is implemented to explore the large search spaces of initial gene pool to discover the best solution with optimal number of predictive genes in the initial gene pool. Performance of the solutions is evaluated using the k-nearest neighbour classifier to identify the class on training samples based on the leave-one-out cross validation. The best solution within each generation is evaluate with the test dataset to give the true error rate. Furthermore, the dissertation aims to investigate the parameters that may affect the performance of the evolutionary algorithm: population size, feature size, and initial gene pools built by various ranking methods. After the exploring on various parameters, the appropriate parameters are chosen to evaluate the performance again using 0.632 bootstrap estimation method to give effective performance.

Chapter 2

Literature Review

Since the advent of microarray technology, one important use for the large amount of gene expression data is to classify the samples in microarray dataset into the correct class. The samples may be types of disease, tumor tissue, or cell line. Many machine learning methods have been introduced into microarray classification to attempt to learn the gene expression data pattern that can distinguish between different classes of the sample. Before the data given from microarray can be analysis with any statistic methods, the gene expression data has to be normalized to reduce noise that may occur during the experiment in the laboratory. In classification task, the studies of multiclass classification have investigated many combinations between gene selectors and classifiers. There are also many datasets that are used to evaluate the performance of classification. Here, the 2 popular datasets are used with in this dissertation.

2.1 The Datasets

The two common datasets which have been frequently used in the literature for evaluating the multiclass classification are: the 3-class leukemia dataset (Golub et al., 1999) and 8-class cell lines tumor NCI60 dataset (Ross et al., 2000). Both will be used in the present study. The leukemia dataset is available at http://www.genome.wi.mit.edu/MPR. In the dataset, gene expression levels were measured using Affymetrix high-density oligonucleotide arrays containing 6,817 genes. This dataset comes from a study of gene expression into two type of acute leukemia: acute lymphoblastic leukemia (ALL) and acute myeloblastic leukemia (AML). The ALL part of the dataset comes from two types, B-cell and T-cell. Golub et al. (1999) studied the data for binary classification between AML and ALL. However, many researchers treated this dataset as a

three-class dataset (B-cell, T-cell, and AML) for multiclass classification. The dataset consists of 47 samples of ALL (38 B-cell and 9 T-cell) and 25 samples of AML. Golub et al. (1999) already divided into a training set of 38 samples and a test set of 34 samples.

The second dataset, NCI60 dataset, is available at http://genome-www.stanford.edu/sutech/download/nci60. The NCI60 dataset contains the gene expression profiles of 64 cancer cell lines measured by cDNA microarrays. The dataset contains 9,703 genes. The single unknown cell line and two prostate cell lines were excluded from analysis due to their small number. The number of cell lines was reduced to 61 with 9 classes of samples: breast (7), central nervous system(5), colon(7), leukemia(6), melanoma(8), non-small-cell-lung-carcinoma or NSCLC(9), ovarian(6), renal(9) and reproductive(4). This dataset is quite difficult to evaluate the performance of the multiclass classifier because there is no test set available.

2.2 The Normalization method for cDNA microarray

The concept underlying cDNA microarray analysis is that the measured intensities for each arrayed gene represent its relative expression level. The relevant patterns of expression are identified by comparing measured expression levels of mRNA between different samples on a gene-by-gene basis. The relative expression from each array is usually measured as the ratio of the red and green intensities for each spot. Although the ratio, or fold-changes, provides an intuitive measure of expression level changes, they have the disadvantage of treating up- and down- regulated genes differently. Genes up-regulated by factor of 2 have an expression ratio of 2, whereas those down-regulated by the same factor have an expression ratio of -0.5. The most widely used of the ratio is the logarithm base 2, which has the advantage of producing a continuous spectrum of values and treating up- and down-regulated genes in a similar fashion. Furthermore, using log base 2 scale is preferred for a number of reasons such as variation of log-ratios is less dependent on absolute magnitude, and taking the log of the ratio evens out the highly skewed distribution, providing a more realistic sense of variation. The log-ratio for each spot can be written M where $M = log_2(R/G)$ or $M = log_2(R) - log_2(G)$.

Many sources in microarray experiment may cause the systematic variation, and affect gene expression level measurement. Many variations in the experiment that can be found are the biases associated with unequal quantities of starting RNA, differences

in labelling or detecting fluorescent dyes, and systematic biases in the expression level measurements (Quckenbush, 2002). Many of these factors make distinctions between differentially and constantly expressed genes difficult. The normalization process must be done on with the log-ratio values before performing any statistical analysis. The normalization step plays an important role in order to minimize these systematic variations, eliminate low-quality measurements, and adjust the measured intensities to facilitate comparisons. Biological differences can be more easily distinguished as well as to allow the comparison of expression level across slides.

The simplest and most widely used within-array normalization method is called global normalization which assumes that the red-green bias is constant on the log-scale across the array. The log-ratios are corrected by subtracting a constant c to get normalized values $M - c$ where the global constant c is usually estimated from the mean or median of the log-ratios(M-value) for a specified set of the genes assumed to be not differential expressed. There are also other estimation methods for the constant c that have been proposed.

At the next level of complication, the location normalization method is often necessary to allow the correction c to vary between spots in an intensity-dependent manner. The log-ratios can be normalized by $M - c(A)$, where $c(A)$ is a function of average spot intensity A. Several intensity-dependent methods have been proposed for location normalization cited in Yang and Thorne (2003) such as local weighted regression (loess) and iterative linear normalization.

2.3 Classification of gene expression data

2.3.1 Gene Selectors

In the classification of tumors using gene expression data, there are two main steps of this problem: gene(feature) selection and classification. The first step is identification of predictive genes that characterize the different tumor classes, this step called feature or gene selection. Data from microarray experiments present a *"large p, small n"* problem; that is, a very large number of features (genes) relative to the number of samples. Feature reduction in microarray data is necessary, since not all genes are relevant to tumor sample distinction. Usually, statisticians determine whether or not a gene is differentially expressed via methodologies known as hypothesis tests. A hypothesis test builds a probabilistic model for the observed data known as a null

hypothesis. The null hypothesis is that there is no biological effect which means the gene is not differentially expressed between different classes. If the null hypothesis were true, then variability in the data does not represent the biological effect under study, the result is from the differences between individuals or measurement error.

The two-sample t-test This is a commonly used method, based on the hypothesis test, for selecting discriminative genes. For each gene, a t-value is computed and genes are ranked by t-value. The t-statistics measure was first used in Golub et al. (1999) to measure class predictability of genes for two-class problems. The probabilistic model known as the p-value can be calculated using t-statistics. This p-value can determine the significance of differentially expressed genes. The gene with lowest p-value is a most predictive gene. But the problems with t-test are it requires that the distribution of data being tested is normal, and since typical microarray data consist of thousands of genes, a large number of t-tests are involved. Moreover, a t-test depends on strong parametric assumptions that may be violated and are difficult to verify with small sample size.

The Correlation between a gene and a class distinction This method identifies informative genes based on their correlation with the class distinction. The correlation between a gene and a class distinction can be measured in a variety of ways such as Pearson correlation or Euclidean distance. Golub et al. (1999) measured correlation, $P(g,c)$, that emphasizes the signal-to-noise ratio in using the gene as a predictor.

$$P(g,c) = \frac{\mu_1(g) - \mu_2(g)}{\sigma_1(g) + \sigma_2(g)}$$

where $\mu_1(g)$,$\sigma_1(g)$ and $\mu_2(g)$,$\sigma_2(g)$ denote the mean and standard deviation of the log-ratio of gene g for the samples in class 1 and class 2. P(g,c) reflects the difference between the classes relative to the standard deviation within the classes. Large values of $\|P(g,c)\|$ indicate a strong correlation between the gene expression and the class distinction.

The ratio of their between-group to within-group sums of squares (BSS/WSS)

The BSS/WSS ratio was introduced by Dudoit et al. (2000) For a gene j, the ratio is

$$\frac{BSS(j)}{WSS(j)} = \frac{\sum_i \sum_j I(y_i = k)(\bar{x}_{kj} - \bar{x}_j)^2}{\sum_i \sum_j I(y_i = k)(\bar{x}_{ij} - \bar{x}_{kj})^2}$$

where I(.) denotes the indicator function, equaling 1 if the condition in parentheses is true, and 0 otherwise. The x_j and x_{kj} denote the average expression level of gene j

across all tumor samples and across samples belonging to class k only. The predictors were built using the p genes with the largest BSS/WSS ratios.

Principle Components Analysis (PCA) This is a well known dimension reduction method and has been used to reduce the high dimension data to only a few gene components which explain as much of the observed total gene expression variation as possible. In PCA, orthogonal linear combinations are constructed to maximize the variance of the linear combination of the gene variables. Generally, the genes are standardized to have mean zero and standard deviation of one. Although the PCA method can handle a large number of genes, only a subset of genes is of interest in practice. Nguyen and Rocke (2002) applied a simple t-statistics of all genes spaces to construct the top p rank genes and then performed PCA to reduce genes from p to K gene components.

The Rankgene software This software has been proposed by Su et al. (2003) for use in gene selection. Eight feature selection methods are supported in the program: information gain, twoing rule, sum minority, max minority, Gini index, sum of variances, one-dimensional SVM, and t-statistics. The first six of these have been widely used either in machine learning (information gain and Gini index) or in statistical learning theory (twoing rule, max minority, sum of variances). They quantify the effectiveness of a feature by evaluating the strength of class predictability when the prediction is made by splitting the full range of expression of a given gene into two regions, the high region and the low region. The split point is chosen to optimize the corresponding measure.

Genetic algorithm(GA) The GA, first described by John Holland in the 70's (Holland, 1975), is a stochastic search and optimization technique that is derived from the principles of evolution and natural genetics. A GA maintains a population of encoded solution candidates that are competitively manipulated by applying some variation operators to find a global optimum. It starts with a random population of chromosomes which are usually represented by a set of strings, either binary or non-binary, constituting the building blocks of the encoded candidate solutions. The better the fitness of a chromosome, the larger its chance of being passed to the next generation. The genetic operators such as selection, mutation and crossover are carried out to introduce new chromosomes into the population. Through evolution, a near optimal solution evolves in the run. Many studies have attempted to use different GA approaches to solve the problem of large search spaces and noisy gene expression data. Several approaches have successfully discovered the optimal gene sets that can correctly classify

100% on some datasets.

Evolutionary algorithm The evolutionary algorithm is based on the same principle of the genetic algorithm but it focuses on a single parent-single offspring search (Fogel, 1994). The crossover operator between two parents is not performed on the algorithm. Initially, a population of chromosome is randomly constructed. Then offsprings are created by randomly mutating each parent with some probability. The offsprings are evaluated and selected into the next generation proportional to its fitness value. The best optimal solution come from the solution with the highest fitness value in the run. Many researches also have been shown the success in applying an evolutionary algorithm to select the small set of genes that can distinguish the classes of samples in microarray dataset correctly (Deb and Reddy, 2003).

2.3.2 The Classification Methods

Once the predictive genes are constructed by gene selection, the second step is to identify samples into known classes using predictive genes as properties of those samples. In general, classification methods can be divided into two categories: supervised and unsupervised. An unsupervised, or clustering, approach has a goal to group together the samples with similar properties. A supervised method is a technique using set of samples with known classification to develop to classifier. Here, the focus is on multiclass classification of gene expression data into known classes. The supervised classifiers that have been used in this field of study include neighbourhood analysis (Golub et al., 1999), support vector machine (SVM) (Ramaswamy et al., 2001), k-nearest neighbours (KNN) (Li et al., 2001), and linear discriminant analysis (LDA) (Dudoit et al., 2000).

The weighted voting This scheme for binary classification was one of the first applications of a gene expression data classification proposed by Golub et al. (1999). It uses a fixed subset of the predictive genes and makes a prediction on the basis of the expression level of these genes in a new sample. Each predictive 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 were summed and called *prediction strength* (PS) in order to determine the winning class. The prediction strength is a measure of the margin of victory ranges from 0 to 1. A sample was assigned to the winning class if PS exceeded a predetermined threshold, and was otherwise considered uncertain. The

threshold of 0.3 was chosen for their prediction.

Support Vector Machines (SVM) The SVM has been shown to give superb performance in binary classification tasks. Intuitively, SVM aims at searching for a hyperplane that separates the two classes of data with largest margin (the margin is the distance between the hyperplane and the point closest to it). For multiclass SVM, there are many decomposition techniques that can adapt SVM to identify non-binary class divisions such as one-versus-the rest, pairwise comparison, and error-correcting output coding. SVM was used in the Park and Cho (2003) and Li et al. (2004) studies.

Maximum Likelihood Discriminant Rules This method is used in Dudoit et al. (2000) research. In a situation where the tumor class conditional densities are known, the maximum likelihood (ML) Discriminant rule predicts the class of a sample with set of predictive genes by assigning class with the largest likelihood to that sample. In practice, however, even if the forms of the class conditional densities are known, their parameters must be estimated from a training set. The computation of the discriminant function for any class is based upon two parameters: the class mean and the common covariance matrix of data for all training samples belonging to that class.

Decision Tree The decision tree takes as input an object described by a set of attributes and returns a decision as a predicted output value. A decision tree reaches its decision by performing a sequence of tests in each node which corresponds to a binary predicate on one attribute. A branch corresponds to the possible values of the test. Each leaf is labelled by a class. To predict the class label of an input, a path to a leaf from the root is found depending on the values of the predicates at each node that are visited. The predicates are chosen by calculating the information gain of each attribute from root to leaf. The decision tree prevent overfitting by using a post-pruning technique. The decision tree has been used in two papers: Dudoit et al. (2000) in aggregating classification and Li et al. (2004) to compare the performance with other classification methods.

K Nearest Neighbour(KNN) The method is based on a distance function for pairs of tumor samples, such as the Euclidean distance, Pearson's correlation, or one minus the correlation of their gene expression profiles. Each sample is classified according to the class memberships of its k nearest neighbours, as determined by one of the distance functions mentioned above. KNN has been involved in much research because of its simple calculation and it also has been shown to perform better than complex methods in many applications (e.g. Dudoit et al. 2000). The KNN classifier defines nonlinear decision boundaries. That is, the KNN can improve the performance

in the case that the sample has small nonlinear fluctuations around the decision boundary result.

Aggregating approach The aggregating classifiers technique includes the bagging approach (Dudoit et al., 2000) and ensemble using voting scheme (Park and Cho, 2003). One way to gain accuracy in classification is to aggregate several classifiers built from perturbed versions of the training dataset. The aggregating technique tends to give benefit to an unstable classifier (e.g. the decision tree) more than a stable one (e.g. the KNN). With the aggregating method, the predicted class for a sample is obtained by a weighted voting schema. There are two types of method for generating perturbed versions of the training set: bagging and boosting. For bagging (non parametric bootstrap) method, the perturbed training sets of the same size as the original training set are formed by drawing at random with replacement from the training set. Classifiers are built for each perturbed dataset and aggregated by plurality voting. A general problem of the bagging for small datasets is the discreteness of the sampling space. Dudoit et al. (2000) performed two methods for solving this problem: the parametric bootstrap and the convex pseudo-data. For boosting method, the training set are re-sampled adaptively so that the weights in the re-sampling are increased for those cases most often misclassified. The aggregation of predictors is done by weighted voting.

2.4 Multiclass classification

In the multiclass classification problem, many combinations between gene selector and classifier have been proposed to improve classification performance.

The research by Dudoit et al. (2000) compared different multiclass classifiers using the BSS/WSS ratio as a gene selection method. The number of the predictive genes was chosen differently in every dataset. There is no specific rule for selecting the size of these predictive gene sets. The comparison between different classifiers is based on random divisions of each dataset into learning set (LS) and test set (TS). The test size was one-third of the dataset. The result from the study in gene selection for leukemia dataset showed that the performance of classifier did not alter greatly when the number of genes increased. For the NCI60 dataset, the error rates were generally lower when the number of predictive genes was equal to 200. The main conclusion of these experiment was that simple classifiers such as DLDA and KNN performed remarkably well compared with more sophisticated ones, such as aggregated classification trees.

The performances of some classifiers were not very sensitive to the number of predictive genes, even though they improve slightly with an increasing number of variables. The reason might be because the predictive genes ranked from this ratio are not really effective.

PCA, which is a well known dimensional reduction method, has been performed on the pre-ranked genes based on a simple t-statistics. For each dataset, Nguyen and Rocke (2002) defined the number of pre-ranked genes to be 50. Following the pre-gene selection using t-statistics and dimension reduction by PCA, the two classifiers were chosen to identify the class of samples: Logistic Discimination (LD) and Quadratic Discriminant Analysis (QDA). The error rate of leukemia dataset was evaluated by the re-randomization method. The result for NCI60 dataset was assessed using Leave-out-one Cross Validation method due to a few numbers of samples in the dataset.

Genetic algorithms were introduced into multiclass classification in many researches as a gene selector. The GA/KNN, proposed by Li et al. (2001), used a GA to find many such gene subsets, and then assessed the relative importance of genes in predictor by submitting the predictor to the KNN classifier in order to identify training samples. The importance give better performance in classification. The subset of genes that give good performance were analysed for the frequency of membership of the genes in these near-optimal predictors. The most frequently selected genes are presumed to be most relevant to sample distinction. The sensitivity of gene selection results was examined by dividing each data set into a training set and a test set in three different ways randomly. In the study, 10000 set of near-optimal chromosomes corresponds to a set of d genes that can jointly discriminate between different classes of samples in training set. The genes were ranked according to the frequency of selection with the top-most gene assigned a rank of 1. In order to validate the result of this method, the dataset was divided into three different ways: the original, random, and discrepant which resulted from multiple splitting of the same training dataset. For the lymphoma dataset provided by Alizadeh et al.(2000), there are only 2 samples misclassified out of 13 samples. For the colon dataset from the study of Alon et al.(1999), there is only one misclassified sample out of 17 samples. Unfortunately, this method was not performed on the datasets that we focus on: leukemia and NCI60 dataset.

The Genetic algorithms and maximum likelihood (GA/MLHD) approach has achieved very high classification accuracies on the multi-class test dataset. The method gave only 5% percent of error rate of the NCI60 dataset while others gave a minimum test error rate of 19%. Although the authors claimed a successful result from this study, it

obviously was an expensive computational technique. The algorithm consists of many parameters to assess for each dataset such as selection methods: stochastic universal sampling or roulette wheel selection and crossover operations: one-point or uniform. The overall strategy consists of two main components: a GA-based gene selector and a maximum likelihood (MLHD) classifier. The GA method finds a set of R genes that is used to classify the samples, where R lies in a pre-specified range. Each chromosome in population represents a subset of these predictive genes. The fitness function was used to determine the classification accuracy of a subset of genes in chromosome by building the function of cross validation error rate and independent test error rate. Despite the high performance result on NCI60 dataset, the error rate estimation method that was used to determine the performance is different from the common error rate estimators. There appeared to be a consistent trade-off between cross validation and test error rate, so they had to apply the sorting technique to look at the most optimal predictor set that represents the best compromise between two types of the error rate. Therefore, the error rate of 5% was got from using only one training/test set compared to 33.4% error given in Dudiot et al.(2000) experiment evaluated with more than one different training/test set. The problem of combining gene selector and classifier is still interesting and challenge in microarray multiclass classification.

Recently, the evolutionary algorithm has been developed by Deutsch (2003) to find the optimal set of predictive genes. The approach embedded the gene selection method into the classifier. The concept of this evolutionary algorithm is that chromosomes, or predictors, in the population was constructed using the subset of genes it utilized in making a prediction. The fitness function, or scoring function, was built by calculating the LOOCV plus an additional mark for the predictor that does a good job of grouping the data into well separated clusters, each cluster corresponding to the same type of cancer. Deutsch also provided the method for searching through a large number of different subset of genes to come up with a population of the highest scoring predictors. The random mutations of genes and the replications of a particular chromosome depending on how the mutation effect the scoring function were the methods for evolving chromosomes in the population. The most successful predictor was the one giving the fewest mistakes on test data. The algorithm was applied on the leukemia dataset to solve three-class problem. The chromosomes with higher fitness can survive in the next generation. Deutsch reported that the algorithm was lack of convergence to near perfect predictors is a problem of this dataset. However, the average number of predictive genes that was found by using this approach is nine with none misclassified

sample of the test dataset.

Table 2.1 summarises the results of the performance in different microarray classification studies reported above on three data sets: 2-class Leukemia dataset, 2-class Leukemia dataset, Lymphoma dataset, and 9-class NCI60 dataset. Most of these approaches have been successful in classification on the leukemia datasets. The methods report on accuracy on the test sample of more than 85%. But for NCI60 dataset, the accuracy is still not impressive due to the small set of samples that makes the performance evaluation difficult. The performance of NCI60 dataset from different studies was evaluated using different resampling techniques. For example, the study by (Ooi and Tan, 2003) used one training set and one test set randomly divided to evaluate the performance whereas Dudoit et al. (2000) used 150 training/test set randomly divided to evaluate the performance of classification.

author	dataset	method		no. genes	accuracy[%]
		gene selector	classifier		
Golub et al. (1999)	Leukemia(2)	t-statistics	weighted voting	50	85.29
Dudoit et al. (2000)	Leukemia(3)	BSS/WSS	KNN	40	97.06
	NCI60			30	86.67
Dudoit et al. (2000)	Leukemia(3)	BSS/WSS	DLDA	40	97.06
	NCI60			30	88.33
Nguyen and Rocke (2002)	Leukemia(2)	PCA	LD	50	97.06
			QDA	50	95.40
Park and Cho (2003)	Lymphoma	Majority voting: IG-KNN(P), MI-KNN(C), PC-KNN(C), SN-KNN(P), and SN-SASOM		50	100.00
Li et al. (2004)	NCI60	sum-minority	SVM	50	66.70
Ooi and Tan (2003)	NCI60	GA	MLHD	13	95.00
Keedwell et al.(2002)	Leukemia(3)	GA	NN	50	88.00
Li et al. (2001)	Leukemia(3)	GA	KNN	50	97.06
Deutsch (2003)	Leukemia(3)	EA	KNN	9	100.00

Leukemia(2) = 2-class Leukemia dataset (ALL,AML)
Leukemia(3) = 3-class Leukemia dataset (ALL B-cell, ALL T-cell, AML)
Lymphoma = Lymphoma cancer dataset available at: http://genome-www.standford.edu/lymphoma
IG-KNN(P) = Information Gain - K nearest neighbour (Pearson's correlation)
MI-KNN(C) = Mutual Information - K nearest neighbour (Cosine coefficient)
PC-KNN(C) = Pearson's correlation - K nearest neighbour (Cosine coefficient)
SN-KNN(P) = Signal to noise ratio - K nearest neighbour (Pearson's correlation)
SN-SASOM = Signal to noise ratio - Structure Adaptive Self Organizing Map

Table 2.1: The result of classification tasks from many researches

Research by Li et al. (2004) has studied and compared the result of multiclass classification using many feature selections and classification methods. The RankGene software (Su et al., 2003) was used to select the predictive genes on the training set with eight methods supported in this software: information gain, twoing rule, sum minority, max minority, Gini index, sum of variances, one-dimensional SVM, and t-statistics. For the number of predictive genes used in classifier, they decided to use the 150 top rank genes of each sample in every dataset. The multiclass classifiers that have been tested were: SVM, KNN, and Decision Tree. They discussed that the SVM was the best classifiers for tissue classification based on gene expression. However, the best decomposition method for SVM appears to be problem-dependent. The KNN classifier gave good performance on most of the datasets which means it is not problem-dependent. Other interesting discussions of their report were that it was difficult to choose the best feature selection method, and the way that feature selection and classification methods interact seems very complicated. Due to the separating of the gene selector part from classifier, there is no learning mechanism to learn how those two component interact with each other and no way to select only the predictive genes from the original set that the RankGene software provided.

Many researches have concluded that feature selection should not be treated separately from classification because gene expression levels in the different samples gives different values, and therefore it would be difficult for the feature selection method to specify genes that are effective to the classifier for each dataset. Using only the RankGene software to filter for the predictive genes without combining it with the classifier may decrease the performance of the classification task. Some research applied both filter method and wrapper method to build the hybrid feature selection method which improve the performance of classification (Xing et al., 2001).

This dissertation will apply the wrapper technique in order to include the gene selection method into an evolutionary algorithm to determine classification performance. The different sets of genes constructed by different gene selection methods in the RankGene software will also be explored.

2.4.1 Evaluation Method

Performance on the microarray classification task is evaluated based on the prediction error rate. The most commonly reported method to evaluate the performance of classifier is the test set error rate estimation. The data set is divided into two sets:training

set and test set. The training set is used to build the classifier. The test set is used to evaluate the performance of the classifier. The error rate is computed by the misclassification on the test set. This training/test set error rate estimation is widely used in the case that the size of dataset is big enough, but in the case of microarray classification, the number of samples collected in the experiments is remarkably small compare to other studies such as digital image classification.

Many evaluation methods has been studied to use for the small-sample error estimation. Typically, the microarray experiment provides a dataset of small size, the most commonly used method for error estimation for a small dataset is leave-one-out cross validation (LOOCV). That is, one of the samples is left out to be a pseudo test data and the classifier is built based on all but the left out sample. The LOOCV is used for each of sample in training dataset. The LOOCV error rate estimator is a straightforward technique for estimating error rates and it is also an almost unbiased estimator. But it is still possible that classifier using LOOCV estimator may accidentally select a model that fits the training data especially due to capitalizing on chance and can give large variance. Another drawback is that it requires expensive computation.

The recent paper by Braga-Neto and Dougherty (2004) compared and discussed various error estimation methods:the resubstitution estimator, k-fold cross-validation, leave-one-out estimation, and bootstrap methodology. These methods were tested with many classifiers: linear discriminant analysis(LDA), 3-nearest-neighbour (3NN), and decision trees(CART)). The leave-one-out (LOOCV) method which has been popular in estimating small-sample error rate is almost unbiased but it has high variance which leads to unreliable estimates. For the over all performance, Braga-Neto and Dougherty (2004) has shown that the bias-corrected bootstrapping is slightly better than cross-validation. The .632 bootstrap proved to be the best overall estimator in their simulations, but the draw back of this method is their computational cost is relatively high compare to LOOCV method.

Chapter 3

The methodology

3.1 The overview

As outlined in chapters 1 and 2, there are some parameters to be determined on the evolutionary algorithm developed by Deutsch (2003). This dissertation focuses on the investigation some parameters that may affect to the performance and effectiveness of the evolutionary algorithm. Then, uses the best parameters to find the subset of genes that gives a high performance in classification and clustering. The methodology is implemented mainly following the methodology in the research by Deutsch (2003) using java programming language. The data preprocessing method was done before further analysis by cutting off the missing values, computing the log-ratios for cDNA dataset, and applying global normalization to the dataset. The algorithm begins with building the initial genes pool containing the more informative genes. Then the evolutionary algorithm builds the initial population of the subsets of genes called the predictor. Compared to the common evolutionary algorithm, a predictor represents the solution that the algorithm try to optimise. The evaluation method for each predictor is implemented using the K nearest neighbour classifier. The performance is determined by counting the number of samples that are correctly classified. The mutation operators that were applied into the algorithm consist of keeping the same, adding a new gene, and removing a gene in the predictor. The selection process is done using the statistical replication algorithm. The termination condition is met when the all predictors are the same over a specific number of generations. An overview of the evolutionary algorithm is shown in figure 3.1.

The main different from genetic algorithm is the evolutionary algorithm focuses on a single parent-single offspring search approach. A single offspring is created from a

```
Build the initial genes pool
Initialize the population of the predictor
Evaluate initial population
    :using KNN classifier and clustering performance
While (the maximum fitness is still changing)
begin
    Apply mutation operator to generate new predictors
    Evaluate new predictors
    Perform statistical replication of the new predictors
end
```

Figure 3.1: Evolutionary Algorithms for microarray multiclass classification

single parent using mutation operator and both are placed in competition for survival, with selection eliminating the poorer solution (Fogel, 1994).

3.2 The data preprocessing

The dataset that I used in this thesis is the leukemia dataset reported by Golub et al. (1999). The leukemia samples were taken from bone marrow and peripheral blood using oligonucleotide microarray technique. It contains an initial training set composed of 27 samples of acute lymphoblastic leukemia(ALL) and 11 samples of acute myeloblastic leukemia(AML), and an independent test set composed of 20 ALL and 14 AML samples. Originally, the dataset was built and analyzed for binary classification: ALL and AML. However, it can be separated into three or four classes by using sub types of the two main classes. To perform a multiclass classification task, 72 samples in the dataset are divided into three classes: ALL B-CELL(38), ALL T-CELL(9), and AML(25). The numerical value of the gene expression data is corresponding to the absolute intensity level. Golub et al. (1999) have normalized the dataset by re-scaling intensity values to make the overall intensities for each chip equivalent and Golub et al. also fitted the data with a linear regression model. From 6,817 genes, the baseline genes were cut off before further analysis. The number of genes that will be used in the multiclass classification is 6,129.

The NCI60 dataset, constructed using cDNA microarray technique, is provided without normalization. The gene expression data value is the relative intensity level of

the mRNA samples and their references. The NCI60 contains 61 samples of 9 classes: breast (7), central nervous system(5), colon(7), leukemia(6), melanoma(8), non-small-cell-lung-carcinoma or NSCLC (9), ovarian(6), renal(9) and reproductive(4). Cutting off the missing value (-intensities), and the background intensities value which is much larger than the foreground intensities, the number of genes is reduced from 9,703 to 7,375 genes.

Due to the noisy nature of dataset provided by microarray experiment, preprocessing is an important step in the analysis of microarray data. The raw intensities have a wide dynamic range. Both dataset have to be normalized to decrease the variation before submitting the dataset to the evolutionary algorithm. The global normalization method to the gene expression data to eliminate the systematic variance. For NCI60 dataset built by cDNA microarray technology, the analysis of relative gene level is done by using the log-ratios between a certain gene (labelled in red or Cy5) and a reference gene (labelled in green or Cy3) before the normalization is applied to the data. From (3.1), the red and green intensities are computed to form the log-ratios.

$$M' = log_2(R/G) \tag{3.1}$$

The method of global normalization is applied to the dataset. Then, the normalization is calculated using the constant c estimated from the mean for the log-ratio M, as shown in (3.2).

$$M = M' - c \tag{3.2}$$

After the preprocessing, the dataset is written into file in order to perform the evolutionary algorithm in multiclass classification in the following step. The data format used in the project is showed in table 3.1 and 3.2.

gene description	gene access number	sample#1	...	sample#N
Transcription factor Stat5b (stat5b) mRNA	U48730_at	10.30	...	30.30
Breast epithelial antigen BA46 mRNA	U58516_at	12.67	...	18.19
...	...	...	...	...

Table 3.1: The training dataset and testing dataset file format

sample no	class label
1	B-CELL
2	AML
N	...

Table 3.2: The label file format

3.3 The evolutionary algorithm for microarray multiclass classification

The algorithm can be divided into several steps. First step is to build the initial gene pool. In this step, the number of genes is reduced from thousands to hundreds, ranked by their predictive values. This will narrow the search spaces and let the evolutionary algorithm search on the more significant genes in the next step.

Next step, the evolutionary algorithm is applied as a gene subset optimization process. The population of the gene selector(or predictors) is constructed randomly. Each predictor is represented in the form of a subset of genes selected from the initial gene pool from the first step. The evolution is performed to determine the performance of the predictors. The performance of a predictor can be distinguished by the degree of goodness of classifying and clustering task. The goodness can be defined by fitness function. In this task, the fitness value is calculated following the method used in Deutsch (2003), the leave-one-out cross validation (LOOCV). Moreover, Deustsch suggested to add an additional score to a fitness function. The suggested score was introduce in order to reflects the clustering performance of the predictor. If the subset of genes in a predictor can group the data into well separated clusters, a predictor will be given the additional score. A predictor that gives higher correctly classified will be given a higher fitness value. For each generation, The basic idea is a predictor with higher fitness will have more chance to be survive in the next generation.

The selection and evolution process is mostly based on the statistical replication described in Deutsch (2003). The termination criteria is defined using both the maximum number of generation and the criteria of no improvement of maximum fitness value of the population. The predictor with highest fitness will be the one that give the best subset of genes for the classification task.

3.4 The initial gene pool

The initial gene pool was build to reduce number of genes or features in a data. Many researches have revealed that when the number of features is large, the performance of the learning methods degrades. Constructing the initial genes pool can get rid of some genes that have low predictive value are noisy to the system. Only genes with high predictive value will be selected into the initial set. Many methods have been devised to construct the initial genes pool. Deutsch (2003) applied the filter method suggested by Xing et al. (2001) to build the primary gene set. Next, these filtered genes are ranked using the method that is similar to Ben-Dor et al.(2000).

In this dissertation, two steps described in Deutsch (2003) are combined into only one step. The initial gene pool will be built using different gene ranking methods in RankGene software provided by Su et al. (2003). The RankGene is a program that has been developed for computing a predictive power of gene expression data in distinguishing between different classes of samples. There are six widely used methods that can be applied for multiclass dataset: information gain, twoing rule, sum minority, max minority, gini index, and sum of variances. All ranking methods are applied to rank the genes and select the top ranked genes into the initial gene pool in order to investigate the most suitable method for each dataset. The aim of the investigation is to account for different characteristics of the data such as the up- or down-regulated genes can vary between datasets.

3.5 The predictor

The initial gene pool contains a complete set of genes $\mathcal{G}_t$ which is a collection of genes 1 through N. The terminology of Deutsch (2003) is used to explain how the algorithm work. We denoted the samples of microarray dataset as $\mathcal{D} \equiv \{D_1, D_2, \dots\}$. Each sample D consists of N genes. A set of possible types in dataset is denoted $\mathcal{T}$. Each sample D can be classified into type T which can take from one of $\mid \mathcal{T} \mid$. A predictor P is built using a subset of genes selected as a feature of classification task. Deutsch defined P as a function that takes sample data D and calculate the output as a classified type T. That is $\mathcal{P}(D) \rightarrow T$.

The initial population of predictors is constructed by selecting genes from the initial gene pool randomly. For leukemia dataset, the initial number of gene(feature dimension) in a predictor is 10 (10 dimensions). The k-nearest neighbor(KNN) classifier

is applied to determine the performance of the predictor with k=3 differently from Deutsch. The number of k is from the research by Li et al. (2001) who perform genetic algorithm and k-nearest neighbour with gene expression data, and suggest the way to find the k value by using the learning curve of the training data to see that which number of k gives best performance in classification task. Li et al. found that setting k to be 3 gave best result among other numbers. Moreover, the discussion in the paper showed that $k = 3$ is large enough to form tight clusters even if the dataset has subtypes and the sample size is limited.

In the evolutionary algorithm process, the KNN classifier uses the Euclidean distance function to calculate the distance between the target sample D and the rest of the training data $\mathcal{D}_t$. The Euclidean distance can be computed using the formula in equation (3.3) where p_i, q_i is the feature value of target sample(D_p) and another training sample(D_q) in dimension i. The sample will be classified according to the class membership of its k nearest neighbors. With the use of the majority vote scheme to KNN, the sample will be classified to the class that has the more membership in the k nearest samples. If there is no class has more membership than the others, the sample is considered unclassifiable.

$$Euclidean\ Distance(D_p, D_q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{3.3}$$

3.6 The scoring function

Choosing the scoring function is an essential step for an application based on an evolutionary algorithm. The scoring function consists of two scoring schemes corresponding to classification and clustering performance. The first one is using the LOOCV result of the KNN. Every sample in the training data will be selected and tested by the KNN classifier using the rest of the samples as a training dataset. If KNN assigns correct class to that sample, the predictor will be added 1.

In additional to the LOOCV of training data, another component of the score is given by the term that maximizes the separation between different classes. Following Deutsch's approach, the shortest distance of every pair of samples within the same type is computed. For each type, we get the shortest distances called $d_1, d_2, \ldots, d_{|\mathcal{T}|}$. With all these distances, the two shortest distances are selected, d_i and d_j and add $C|d_i^2 - d_j^2|$ where C is a constant chosen to make the value of these term less than 1.

This scoring function depends on the predictor and can determine the performance of predictor according to the training data and the subspace of genes $\mathcal{G}$.

3.7 The statistical replication

To determine rules to evolve a predictor to a new generation, a measure of the goodness of a predictor is calculated using scoring function described above. The statistical replication method will use the score value to decide which predictors are kept, which are mutated, and which are eliminated. The statistical replication has to ensure the diversity of predictors in a population and also try to get rid of local minima problem in predictor space by allow the less fit survive occasionally.

Let the system start with an ensemble of n predictors, or n gene subspaces, $\mathcal{E} \equiv \{G_1, G_2, \ldots, G_n\}$, the replication algorithm described in Deutsch's work to replicate and modify each of the G_i's is implemented with some changes. The replication approach is modified by introducing an elitist strategy. The elitist strategy is a mechanism that guarantees that some number of the fittest solutions will be retained at each generation. Here, only one predictor with highest score is duplicated into the next generation. The pseudo code is described here:

1. For each predictor $G \in \mathcal{E}$, a new predictor is produced as follows.
   (a) A predictor G has subset of genes $\{g_1, g_2, \ldots, g_m\}$. For all predictors, those genes in a subset will be randomly mutated according to the three possibilities:
      i. Add an additional gene chosen randomly from the initial gene pool $\mathcal{G}_i$, producing new predictor G' of genes $\{g_1, g_2, \ldots, g_m, g_r\}$.
      ii. Delete a gene: a gene in the subspace of predictor is randomly deleted, producing a new set with total gene equals to $m-1$.
      iii. Keep G the same.
   (b) The scoring function of new predictor G' is computed: $\mathcal{S}_{G'}$.
   (c) The difference of the score value between the new predictor and old predictor is computed $\delta\mathcal{S} = \mathcal{S}_{G'} - \mathcal{S}_G$
   (d) The weight for G' is computed: $w = \exp(\beta \cdot \delta\mathcal{S})$ where β is the inverse temperature.
2. Let Z is the sum of these weights. The weights is normalized by multiplying them by n/Z.
3. All predictors are replicated according to their weights after normalization. With a weight w, the predictor is replicated $[w]$ times and additional time with probabilities $w - [w]$ where $[w]$ is the largest integer that less than w.
4. Using the elitism technique, the worst predictor of the population in new generation is replace by the best predictor from the previous generation.

The statistical replication described above makes the evolutionary algorithm different from the genetic algorithm. One major different is there is no crossover process between the solutions. The evolutionary believes that the mutation operator within each solution itself can improve performance of the solution. In best case, mutation may help move a predictor away from local minima by the replacement of the new genes into the predictor (Ghanea-Hercock, 2003). Using this statistical replication method, every predictor in the system is mutated and replicated in accordance with how much fitter it was than its predecessor. The each generation can be guaranteed that the best predictor from the previous generation will survive into the next generation because of the elitism rule. The number of predictors in system can also be varied in every generation but with the step of normalization in the pseudo code, this helps the number of predictor in the system stays close to *n* theory. However, in practice, the number of predictor can vary from *n* depending on the method of setting the temperature variable described below.

3.8 Annealing

The expected behavior of this evolutionary algorithm is that the scoring function will give similar scores for all predictors in the populations. In order to lead the system into convergence process, the measure of temperature is defined. The temperature technique is applied to the predictor and by looking at the fluctuation between predictors. The original work by Deutsch (2003) suggested that after exploring many calculation methods to the system, the evolutionary algorithm works well when the temperature scale is set to be proportional to the standard deviation of score value of all predictors in each generations adaptively. Despite of the use of temperature variable directly, the β variable appeared as the inverse temperature in the statistical replication algorithm is used. The standard deviation of the score value of predictors in each generation is used to determine the value of β variable adaptively. The formula in equation (3.4) which is similar to Deutsch but it is changed to be more suitable for the system. The *SD* donates the standard deviation of the scores for all predictors, and the β_{last} refers to the β from the previous generation. This temperature scheme gives the diversity to the evolutionary algorithm. So, the system does not get trapped in the local minima.

$$\beta = \sqrt{\beta_{last} * \frac{1}{SD}} \tag{3.4}$$

Due to the temperature factor in the algorithm, the system always changes in order to keep diversity in the population. The original termination condition is changed from looking at an unchanged system for ten consecutive iterations to looking at standard deviation of the overall score of the system. When the standard deviation is less than 0.01 for ten consecutive iterations, the algorithm is terminated. Furthermore, the additional termination condition is added to the algorithm to reduce the cost of computation time. That is the maximum generation of algorithm is set to 200. If the algorithm can not meet the standard deviation criteria, it has to stop at 200 generations.

3.9 The evaluation method

In this dissertation, two resampling evaluation techniques are used to evaluate the performance of the predictor: leave-one-out cross validation(LOOCV) and .632 bootstrap. The LOOCV has been recommended for problems where relatively small sample sizes are available. In the evolutionary algorithm process, only the training set is used to evaluate the performance. For a given training sample size, n, a KNN classifier classifies on the single samples by comparing the distance with (n-1) samples. After getting the best predictor with the best performance on training set, the classifier is applied on test set to determine the performance by comparing the Euclidean distance of each sample in test set with all samples in training set to find the class with highest membership according to the majority scheme.

The newer resampling method, bootstrapping, has shown to be the better performance estimator in many papers. According to the research by Braga-Neto and Dougherty (2004), the .632 bootstrap introduced by Efson(1979) has proved that it perform well in error estimation technique because of its low variance estimation in comparing with the LOOCV that has a high variance for small samples. Given a dataset of size n, the n training samples are created by sampling with replacement from the data. Sampled with replacement means that the training samples are drawn from the dataset and placed back after they are picked, so their repeated use is allowed. Since the dataset is sampled with replacement, the probability of any given instance not being chosen after n samples is $(1-1/n)^n \approx e^{-}1 \approx 0.368$ and the expected number of distinct instances from the original dataset appearing in the test set is thus $0.632n$. The .632 bootstrap estimate is defined as the equation (3.5).

$$acc_{bootstrap} = \frac{1}{b}\sum_{i=1}^{n}(0.632 * \varepsilon 0_i + .368 * acc_s) \quad (3.5)$$

Where the ε0 accuracy estimate is the error rate on the test data. The acc_s is the resubstitution accuracy estimate on the full dataset. The estimated error rate is the average of the error rates over a number of iterations *b*. Usually, the bootstrap estimators need about 200 iterations to obtain a good estimate (Bao, 2004). Thus, this is computationally considerably more expensive compare to the LOOCV estimator. With this reason, the bootstrap is not suitable in order to use for investigate the performance of the algorithm given by different parameters that will be tested on the evolutionary algorithm. The bootstrap estimator will be performed on the evolutionary algorithm with the selected parameters given after the LOOCV performance analysis is applied to various parameters. The final performance of the multiclass classification on both leukemia and NCI60 dataset will be reported by using .632 bootstrap estimator with 200 iterations of data sampling.

3.10 Experimental Setup

The study aims to evaluate the performance of the evolutionary algorithm combined with the ranking methods provided by RankGene software. Some evolutionary algorithm's parameters are fixed. The probability of the predictor will be mutated is set to 0.7. The probability of adding new gene to the chromosome and the probability to delete any random gene from any random position in predictor are set to 0.5. This will give the equal chance for adding and deleting gene into/within predictor.

Since the evolutionary algorithm is a stochastic search, it is impossible to report the performance of the algorithm with in one trial. The results from the evolutionary algorithm are the average accuracy over a number of trials. We have to limit the trial number to balance the cost of the computational time of the algorithm because the experiment has to perform on many parameters. Here, the performance reported by an average of the accuracy rate within 10 trials.

Some parameters are varied to determine the effects of them on the evolutionary algorithm in classification task. The parameters are tested on the leukemia dataset with the initial gene pool that is built using information gain ranking method which is the popular method which is used in many researches. Here is the list of a number of parameters that will be varied: (i) population size (10,30,50) , (ii) the number of initial

genes (100,200,500), (iii) the feature size(10,30,50), and (iv) 6 rank methods. In the parameter analysis phase, the performance is evaluated using LOOCV estimator on training samples and test error on test samples.

Although the LOOCV is not the best error rate estimator for the small size sample, an advantage of using LOOCV is it give almost unbias estimation and the most important thing is its computational time is faster than the .632 bootstrap. The LOOCV also has been used in many studies in microarray classification. It is acceptable to perform the LOOCV for parameter analysis.

Firstly, the experiment is performed using the three parameters with one ranking method: information gain. After the analysis, the set of suitable parameters from the analysis are used to determine the best ranking method for each dataset. In the final step, parameters that give highest accuracy will be chosen to perform the evolutionary algorithm again with the .632 bootstrap estimator in order to achieve more reliable performance in multiclass classification task.

Chapter 4

Experimental Results

In this section we present and analyse the performance results on the leukemia and NCI60 dataset for multiclass classification based on the evolutionary algorithm. The experiments performed on 1.6GHz Redhat Linux machine with 256MB are following this step:

- The baseline system is tested on the leukemia dataset without the use of any ranking methods.
- Various parameter configurations are investigated using the initial genes pool constructed by information gain ranking method. The performance results are reported based on the leave-one-out error estimator.
- The range of parameters that give good performance will be selected to perform the performance comparison with other ranking methods provided by the RankGene software.
- The reproducibility of the evolutionary algorithm is investigated to select the top-ranked genes by z-score and evaluate the performance of the top-ranked gene set.
- The .632 bootstrap error estimator is applied to the evolutionary algorithm on the datasets using the most appropriate parameters and ranking method to report the final performance.

4.1 Baseline

The evolutionary algorithm is run on the baseline system to see the basis performance of the evolutionary algorithm without using the RankGene software. We will compare the baseline system to the evolutionary algorithm with different ranking methods provided by the RankGene software later on. In baseline configuration, the evolutionary algorithm has to search for subset of predictive genes from all genes set given by the microarray experiment.

The baseline system is built on the leukemia dataset with 7,129 genes. The initial predictors in population are built by randomly selecting 10 genes to be an initial feature of the predictors. This means the evolutionary algorithm has to search for 10 optimal predictive genes set from the $\frac{7129!}{(7129-10)!}$ possible subsets. Performance of the predictors is evaluated using KNN classifier to determine the LOOCV on training samples. After the best predictor is found in each generation, it will be tested again on test samples to give the performance based on the out-of-samples estimation manner. The KNN classifier will classify the each sample in test data using all training samples for the Euclidian distance comparison.

The first investigation is to determine the average score change as the solutions evolve because the optimal predictive gene set may be found by chance. Figure 4.1 shows the average scores in each iteration from several trails of the experiment. The maximum iteration for each trial is given by the termination condition. The evolutionary algorithm will stop when the score of all predictors in the population give the standard deviation less than 0.01 for ten consecutive generations or the evolutionary algorithm reach the defined maximum generation(200 generations). We can see that the score increases as iteration goes. This graph confirms that the evolutionary algorithm can find the better solutions as the iteration increases. Especially at the first few generations, the graph shows that the average score increase the the larger scale compared with the last few generations. Figure 4.1 also shows that predictors quickly reach the perfect classification on training samples. The evolutionary algorithm converges and terminates within less than 50 generations.

The performance of the evolutionary algorithm with different feature sizes and population sizes is also addressed. Table 4.1 reports the maximum and average accuracy on 38 training samples and 34 test samples of the baseline system. The result shows that the evolutionary algorithm gives predictors with perfect classification on the training samples but those predictors can not classify the test data perfectly.

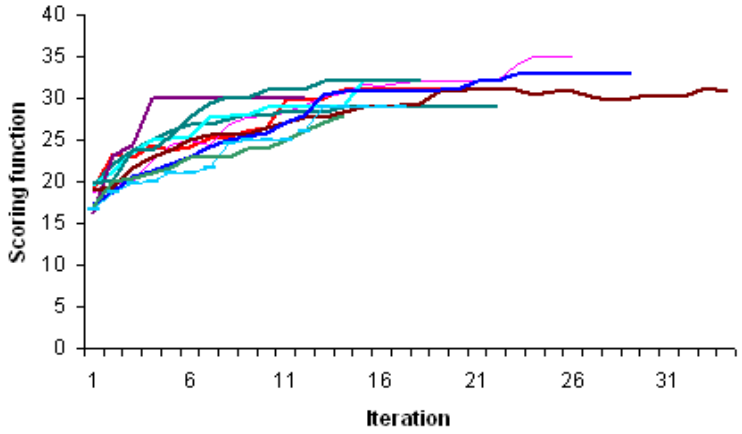

Figure 4.1: The average score in each iteration for several trails of the evolutionary algorithm on the leukemia dataset. Each trial starts with different population of the predictors.

The average accuracy on test data is approximately 66 percent when the average accuracy on training data is around 94 percent. By increasing the number of genes in predictor(feature size) and population size, it does not greatly affect the performance of the algorithm. These may be because of the large size of the initial gene pool give large search space to the evolutionary algorithm. Some search spaces may never be covered before the algorithm is terminated. Another reason may be because of the overfitting problem which will be discussed in next chapter.

Population size	Feature size	Training data [%]		Test data [%]	
		Max	Average	Max	Average
10	10	97.37	81.84	76.47	61.18
	30	100.00	88.95	79.41	66.76
	50	97.37	88.16	79.41	66.47
30	10	97.37	97.11	76.47	67.64
	30	100.00	98.42	70.59	68.53
	50	100.00	97.10	76.47	64.70
50	10	100.00	99.73	79.41	71.18
	30	100.00	97.89	79.41	68.53
	50	97.37	98.42	76.47	66.17

Table 4.1: The accuracy of the base line system built by randomly selecting genes from 7,129 genes in the leukemia dataset.

Another analysis on the baseline system is the relationship between the training

and test data performance. The relationship can be determined by plotting graph of the average accuracies of training and test data as shown in figure 4.2. Although the accuracy on test data is lower than training data, the performance can increase respectively to the improvement of the predictors on the training data regardless to different feature sizes in the predictors or the initial population size the evolutionary algorithm uses.

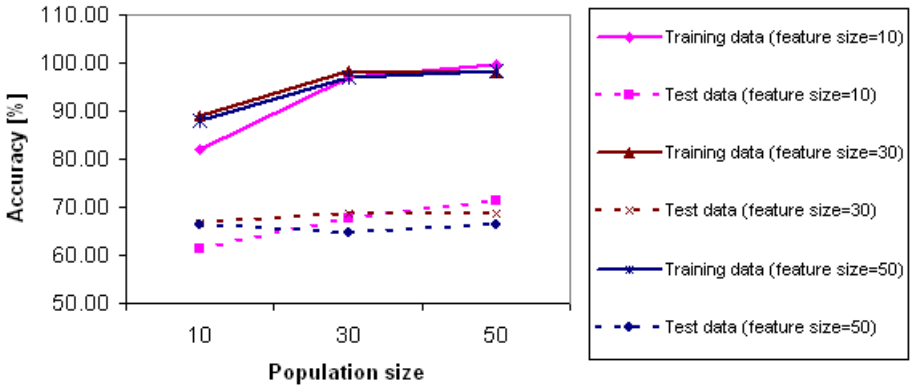

Figure 4.2: The change of average accuracy on training and test set according to the population size.

4.2 The evolutionary algorithm's parameters on leukemia dataset

The leukemia dataset is chosen to be tested on various parameters because it provides both training and test samples. The proposed parameters are performed on the original data structure of 38 training samples and 34 test samples: population size(10, 30, 50); feature size(10, 30, 50); and initial gene set 100, 200, 500 constructed by information gain ranking method. The KNN classifier uses the subset of gene in predictors to classify the class of training samples. In each generation, the best predictor that give highest score is selected calculate testing error on the test samples.

The classification results of all parameters are summarised in table 4.2. With ten trials of each experiment, all methods predict classes almost 100% correctly for the 38 training samples using leave-one-out cross validation. With the new initial gene pool constructed by the information gain ranking method, the accuracy on test data of the predictor is higher than the all accuracy on the baseline system. The prediction of the test samples using KKN classifier based on the training set varies on different

parameters but it does not varies in the big scale. If we looking at the specific predictor, there are some predictors discovered by the evolutionary algorithm that give 100% correct classification is marked with (*) in the table.

Initial gene size	Population size	Feature size	Training data (LOOCV)[%]	Test data (out-of-sample)[%]
100	10	10	100.00	84.11
		30	100.00	92.35
		50	100.00	94.41
	30	10	100.00	84.12
		30	100.00	93.23*
		50	100.00	95.00*
	50	10	100.00	89.71*
		30	100.00	89.71
		50	100.00	93.82
200	10	10	100.00	87.35*
		30	100.00	90.00*
		50	100.00	92.65
	30	10	100.00	83.24
		30	100.00	86.18
		50	100.00	90.00
	50	10	100.00	84.41
		30	100.00	90.00
		50	100.00	88.82
500	10	10	98.68	84.41
		30	100.00	83.53
		50	100.00	86.76
	30	10	100.00	81.47
		30	100.00	87.35
		50	100.00	85.88
	50	10	100.00	81.76
		30	100.00	83.53
		50	100.00	83.53

Table 4.2: The average accuracy measured in percentage on different parameters performed by the evolutionary algorithm. The accuracy is computed using LOOCV to count the correct classified samples within the original 38 training samples and out-of-sample prediction for the 34 test samples. The symbol (*) means that there is some perfect predictors found by the algorithm.

The evolutionary algorithm gives low performance on the 500 initial genes because of the search space is too large to cover all possible subset within the limited population and feature size. The parameters that give optimal performance which mean it can

balance between computational cost and the test error rate.

From the results on the leukemia dataset, the performance analysis is performed in order to choose the set of parameters that perform well in classification task. The chosen parameters will be applied the algorithm with other ranking methods in the RankGene software.

4.2.1 Initial gene pool

The initial genes pool provides a search space to the evolutionary algorithm. If the search space is too large, it is possible that the algorithm can not discover the predictive genes with in the search space. On the other hand, if the initial gene size is too small, it is possible that some predictive genes are not included in the search space. The small search space decrease performance of the evolutionary algorithm before it performs its work.

In order to distinguish the performance according to the size of the initial gene pool, the graph according to different size of gene pool is plotted as shown in figure 4.3. Considering only the affect of the size of initial gene, all different feature sizes are plotted by varying the size initial genes pool. The population size is fixed to 30 for every curves. The graph shows that the performance of the evolutionary algorithm in two cases out of three (feature size 10 and 30) are decreased when the size of initial gene pool increases. For 100 initial genes, the accuracy rates are high compare to 200 and 500 initial genes. From the graph, the number of 100 genes seems to give better performance for all predictors. The initial gene pool with size 100 will be chosen to perform the further analysis.

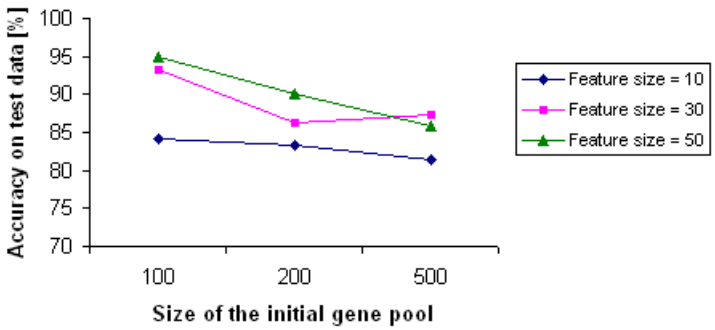

Figure 4.3: The average accuracy on test data according to the size of the initial gene pool.

4.2.2 Feature size

The feature size is the number of genes included into the predictor. This parameter is important for the optimal number of predictive genes that will be discover through the evolutionary algorithm.

Due to the design concept of this evolutionary algorithm, the number of genes in predictors or feature size can be varied across predictors and generations. There are two probabilities that can affect the subset of genes in the predictor. The first probability determines whether the predictors will be kept the same across the generation or will be mutated by changing the subset of genes within the predictor itself. In this experiment, we want to evolve the predictors across generations. The predictors will be randomly selected with the probability of 0.7 to be mutated, otherwise it will be kept the same. The low probabilities is useful in such case that assigning the mutation operator may destroy the good subset of genes that already found in the predictors in the generation.

If the predictor is selected to be mutated, the next probability is assigned to the predictor offering the choices of the mutation operators. Two mutation operators are implemented and assigned to the predictor: adding a new gene into the predictor and randomly deleting a gene from subset in the predictor. The evolutionary algorithm is implemented with the fixed probability of 0.5 in order to give an equal chance in selecting a mutation operator to the predictors.

To determine the effect of mutational probability, an average of feature size over the generations is plotted as shown in figure 4.4.

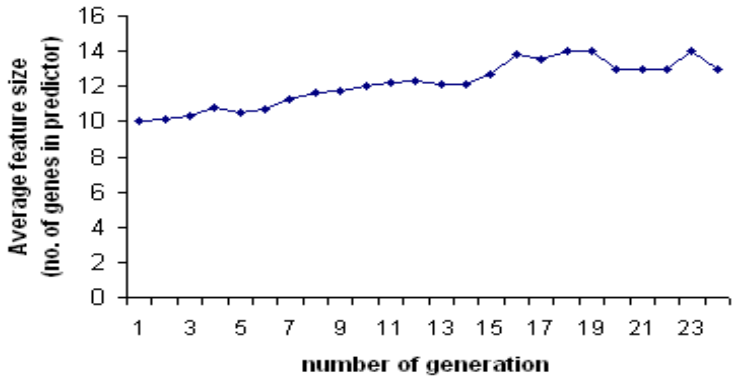

Figure 4.4: The average feature size of best predictors in each generation. The initial feature size and initial genes pool are set to 10 and 100 respectively.

The evolutionary algorithm starts with the initial feature size 10 and the algorithm is run until the terminate condition is met. The result shows that the number of genes involved in the best predictor is not remarkably different from the initial gene size in the starting point of the algorithm. However, the best predictors tend to increase the number of genes in order to obtain the better classification result in the next generation.

The feature size is varied in three values(10, 30, 50). To select appropriate feature size that give the good classification result, the graph of an average accuracy with different feature size is plotted to clearify the affect of the feature size on the classification performance. Figure 4.5 shows the average accuracy when the initial feature size is increased. The large feature size can improve the performance of the predictor. The feature size 30 and 50 will be used to perform classification task using other ranking methods.

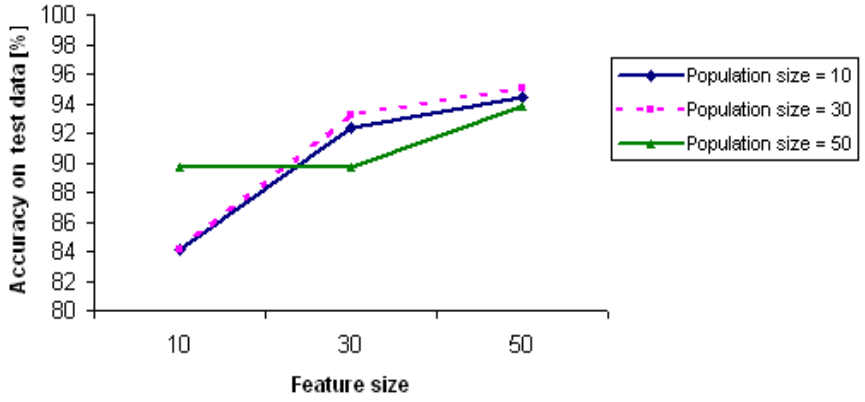

Figure 4.5: The average accuracy on test data varying on the feature size.

4.2.3 Population size

The population size is one of the important choices faced by any use of evolutionary algorithms. This parameter causes the balancing problem between the convergence time and computational time. If the population size is too small, the evolutionary algorithm may be converge too quickly. If it is too large, the evolutionary algorithm may waste computational resources: the waiting time for an improvement might be too long. The experiment determines three different sizes of population(10, 30, 50). Moreover, the size of population in this evolutionary algorithm can be flexible according to the statistical replication technique described in chapter 3. The predictors with higher weight will have a probability to be selected into the next generation more than one time.

To compare the affect of population size to the test error rate and the computational time, the graph is plotted as shown in figure 4.6. The number of the feature size and initial genes set are fixed to shows the performance of the evolutionary algorithm on different initial number of the population size. The results show that the average time usage greatly increase exponentially as the population size increases but the test error rate of the predictors does not significantly goes down. Some predictors in the experiment give less performance when the population increases. This gives us the difficulty to determine and select the most suitable parameters to determine the performance of other ranking methods. For the graph, the experiment on the population size 10 and 30 is still acceptable to perform. Therefore, the number of predictors 10 and 30 will be used in further analysis.

The population size 50 gives the less error rate among other population size but it is not significantly different from the others. Theoretically, the expensive computational cost comes with the diversity of the evolutionary algorithm. We wish to include the population size 50 into the further analysis to determine the tradeoff between computational time and the convergence of the evolutionary algorithm. The population size of 50 will be tested on only the feature size 50 which give the less error rate to reduce the work load in the experiment.

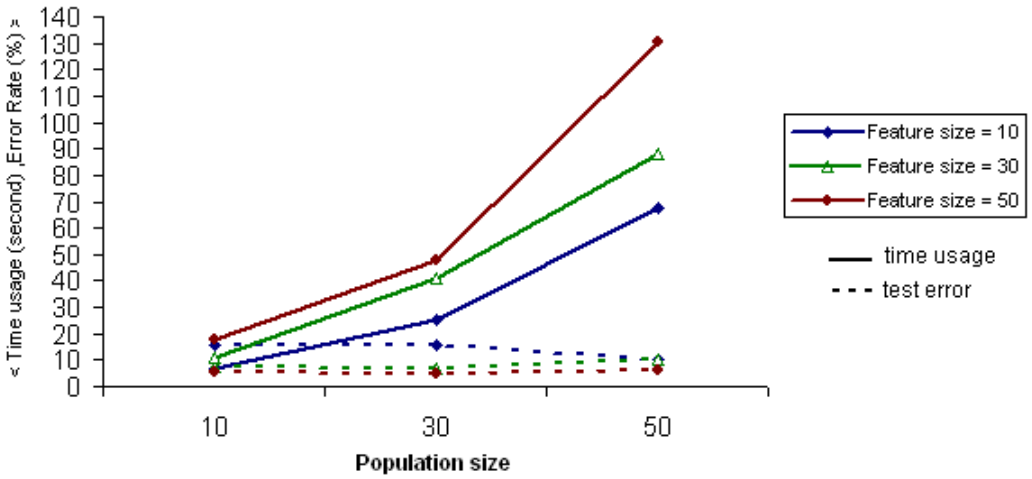

Figure 4.6: The change of population size affects the computational time and accuracy of the evolutionary algorithm on leukemia dataset

Finally, the analysis of all parameters gives us the range of suitable parameters that are used to. We decide to choose only five subset of parameters to investigate on other ranking methods in the next step.

- **The initial gene size** 100 top-ranked genes from ranking methods will be used.
- **The feature size** 30 and 50 number of initial feature size are chosen.
- **The population size** The population size 10 and 30 will be chosen on feature size 30 and 50. The population size 50 will be run with 50 initial numbers of feature.

4.3 The rank methods

Many ranking methods provided by the RankGene software are used to build the initial gene pool and investigate that which method gives the best performance for leukemia dataset. Using the experience with the information gain rank method, five sets of parameters are performed to determine the performance given by other different rank methods. All methods that are used to build the initial gene set are list below. The details of each method can be found at http://genomics10.bu.edu/yangsu/rankgene/.

1. Information gain
2. Twoing rule
3. Gini index
4. Sum minority
5. Max minority
6. Sum of variances

4.3.1 The leukemia dataset

Table 4.3 shows the average accuracy on the test data of the leukemia dataset. With the initial gene constructed by all rank methods, the performance is better than the baseline system. The rank method that gives the best performance is the twoing-rule with the population size 10 and feature size 50.

Initial gene size	Population size	Feature size	The accuracy of different rank methods on the Test data (out-of-sample)[%]					
			#1	#2	#3	#4	#5	#6
100	10	30	92.53	93.53	88.24	86.18	94.41	89.70
		50	94.41*	**96.18***	**96.18***	87.94	94.71*	93.82
	30	30	93.23*	94.12*	85.59	90.00	90.59*	89.41
		50	95.00*	94.71*	90.88	90.29	94.12*	92.06
	50	50	93.82	91.47*	88.53	88.82	93.23	96.11

Table 4.3: The average accuracy on 5 sets of parameters with 6 ranking methods measured by using out-of-sample prediction on the 34 leukemia test samples. The symbol (*) means that there is some perfect predictors found by the algorithm. The highest accuracy is written in bold.

4.3.2 The NCI60 dataset

The same sets of parameters are given from the experience on the leukemia dataset to perform on the NCI60 dataset. Six ranking methods are tested to determine which method gives best performance in multiclass classification. Due to the very small sample size of the NCI60 dataset, it is difficult to divide the data into training and test set.

The accuracy of predictors in the table 4.4 is given by using the LOOCV error rate estimation on the whole dataset. The aim of this investigation is only to find best parameters and ranking method. To assess more reliable performance of the evolutionary algorithm on the NCI60 dataset, the .632 bootstrap estimator will be used later.

The results from the NCI60 is not very impressive for all the ranking methods. The computational time for the experiment is also significantly expensive compare to the same set of parameters on the leukemia dataset. To complete the run, each trial takes approximately 1 minute for population size 10, 25 minutes for population size 30, and 1 hours for population size 50.

No predictor classifies all data 100% correctly. The best performance found in the comparison comes from the use of the information gain technique at the population size and feature size are equal to 30.

Initial gene size	Population size	Feature size	The accuracy of different rank methods on all dataset (LOOCV)[%]					
			#1	#2	#3	#4	#5	#6
100	10	30	66.72	63.77	60.00	54.43	62.78	69.34
		50	67.86	62.78	61.63	52.62	62.62	65.90
	30	30	**76.23**	72.29	72.02	65.90	74.26	75.41
		50	73.44	72.46	71.15	63.11	73.44	73.93
	50	50	75.08	72.29	71.96	71.97	73.77	74.16

Table 4.4: The average accuracy on 5 sets of parameters and 6 ranking methods measured by the LOOCV on 61 samples of the NCI60 dataset. The highest accuracy is written in bold.

4.4 Discrimination method

To assess the reproducibility of the algorithm, the frequency of the specific genes that are members of the best predictor across 100 independent trials with different initial populations. The number of genes that are consistently preferentially chosen by the evolutionary algorithm, suggesting that the gene selection operation executed by the algorithm is highly reproducible despite the initial solutions are generated randomly.

Li et al. (2001) suggested the way to determine the number of top-ranked genes that found in predictors by using the statistical z-score based on normalizing the frequency with which each of the initial genes was selected in all predictors that classify the training and test data perfectly. The Z score can be calculated using the equation (4.1).

$$Z = \frac{S_i - E(S_i)}{\sigma} \tag{4.1}$$

Where S_i denotes the number of times genes i was selected, $E(S_i)$ is the expected number of time for gene i being selected, and σ denotes the square root of the variance. The calculating of $E(S_i)$ can be done by: let A = number of perfect predictors found in the experiment, P_i = (number of genes) / (number of genes in the initial gene pool). Then, $E(S_i) = P_i * A$.

For the leukemia dataset, the discrimination method performs 100 trials of experiment using the best set of parameters: 100 initial genes constructed by the twoing rule and feature size = 50. There are 511 predictors (ignoring the duplication) that classify all training and test data correctly. Figure 4.7 shows a plot of z-score applied to the top ranked genes that are frequently selected by the evolutionary algorithm.

The z-score decreases quickly for the first 5 to 10 genes. The decrease is much

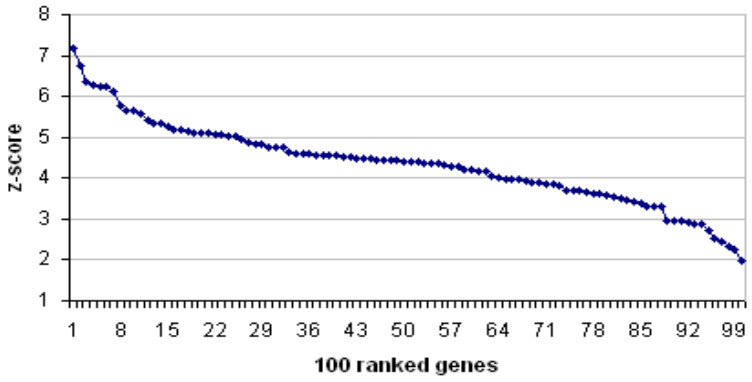

Figure 4.7: A plot of z-scores for 100 ranked genes on the leukemia dataset.

slower after 25 genes. In this case, it seems reasonable to choose 30 to 40 top-ranked gene as the most discriminative genes. In this experiment, the 30 top-ranked based on the statistic z-score analysis are selected to determine the performance by submitting the top-ranked genes to KNN classifier and let it classifies the test data using training data.

The results of the classification with the 30 top-ranked gene are correctly 100 % on both training and test set on the leukemia dataset. Table 4.5 shows the 30 top-ranked genes with its frequency found in 511 best predictors including the z-score value. This set of genes will be used in the further evaluation using the .632 bootstrap estimator.

Gene Index	Gene Access Number	Frequency	z-score value
5543	D00749_s_at	432	7.189
3773	U79274_at	405	6.740
1834	M23197_at	381	6.340
4951	Y07604_at	377	6.273
4177	X15949_at	376	6.256
1926	M31166_at	375	6.240
5552	L06797_s_at	369	6.140
6803	M19045_f_at	348	5.790
1247	L08177_at	340	5.657
3108	U36922_at	339	5.640
4913	X99584_at	335	5.573
5039	Y12670_at	325	5.407
1078	J03473_at	320	5.323
4586	X77094_at	320	5.323
412	D42043_at	317	5.273
4050	X03934_at	312	5.190
3072	U33839_at	311	5.173
6797	J03801_f_at	309	5.140
5688	L33930_s_at	308	5.123
5300	L08895_at	307	5.107
760	D88422_at	306	5.090
3281	U48251_at	305	5.073
4318	X58529_at	305	5.073
6225	M84371_rna1_s_at	302	5.023
4484	X69398_at	302	5.023
1120	J04615_at	298	4.957
962	HG3998-HT4268_at	292	4.857
2050	M58297_at	290	4.823
4535	X74262_at	290	4.823
6218	M27783_s_at	286	4.757

Table 4.5: The list of 30 top-ranked genes order by the frequency that gene is selected into the predictors.

4.5 The .632 bootstrap error estimation

After the analysis various parameters, the most suitable parameters evaluated by the LOOCV estimator will be tested again on the evolutionary algorithm using the .632 bootstrap error estimation. The results are shown in table 4.6.

For the leukemia dataset, the twoing-rule ranking method with population size 10 and feature size 50 gives the best performance. The evolutionary algorithm is applied with these parameters for reporting the final performance. The algorithm takes approximately 18 hours to complete a trial. The best predictor with 50 predictive genes found in the algorithm gives 96.40% of accuracy. The .632 bootstrap estimator is also used to determine the performance of the predictor which consists of top-ranked genes based

on the discrimination method. This predictor gives 98.61% accuracy which is slightly higher than the evolutionary algorithm with rank method.

The accuracy evaluated by the .632 bootstrap estimator					
Rank method	Initial gene size	Population size	Feature size	No. genes	accuracy[%]
Twoing rule	100	10	50	50	97.40
discrimination	100	-	30	30	98.61

Table 4.6: The accuracies of the best predictor discover by the evolutionary algorithm on the leukemia dataset. The performance is evaluated by the .632 bootstrap estimator.

4.6 The comparison of the evolutionary algorithm with literatures

Table 4.7 presents the performance results on the leukemia dataset by the evolutionary algorithm compare to results from other literatures. Although the results give less performance than the original work by Deutsch (2003), the performance measure is more reliable than the test set error rate. The accuracy on the leukemia dataset is relatively high compare with several literatures. Furthermore, the predict with only 30 top-ranked genes getting from the discrimination method provides higher accuracy than the use of the evolutionary algorithm with the RankGene software.

author	method		no. genes	accuracy[%]
		gene selector	classifier	
Keedwell et al.(2002)	GA	NN	50	88.00
Dudoit et al. (2000)	BSS/WSS	KNN	40	97.06
Dudoit et al. (2000)	BSS/WSS	DLDA	40	97.06
Li et al. (2001)	GA	KNN	50	97.06
EA	**EA+Twoing rule**	**KNN**	**50**	**97.40**
EA	**EA+Twoing rule+discrimination**	**KNN**	**30**	**98.61**
Deutsch (2003)	EA	KNN	9	100.00

Table 4.7: The comparison of the results on the leukemia dataset obtained from the dissertation(written in bold) with other published results.

Chapter 5

Discussions

The results as given in chapter 4 are satisfactory, improving on several of the previously published error rates especially on the leukemia dataset. However, there is room for improvement. Furthermore, by use of the .632 bootstrap estimator, we get a better performance measure in contrast with reports of a single best run of an algorithm found in literature.

5.1 The baseline system

In the baseline system, the evolutionary algorithm easily discover the predictors that can classify the the training samples 100% correctly. In contrast with the test error which is relatively low. Because the evolutionary algorithm leads the predictors to perfect classification on training set very quickly. This characteristic of the algorithm causes the overfitting problem. The problem generally occurs when the classifier has only small training dataset (Raudys, 2001).

The overfitting affects directly to the algorithm convergence mechanism. When numbers of the predictor fit all training samples, the similar score will be given to those predictors. Due to the additional score is in scale of 1, it does not significantly affect the diversity of the predictors with in the generation. This will lead the algorithm to the terminate condition quicker than expected. Finally, the test error rate can not be improved within the short number of generation.

5.2 The evolutionary algorithm

The evolutionary algorithm has been shown to give satisfactory performances on the datasets with the different parameter configurations.

For the leukemia dataset, we have shown that the evolutionary algorithm finds some predictors that can correctly classify 38 training samples and 34 test samples based on leave-one-out cross validation (LOOCV) and out-of-sample estimation respectively. The accuracy by using .632 bootstrap also high (97.40%). In spite of the high performance on the leukemia data, the evolutionary algorithm does not give the impressive performance on the NCI60 dataset.

The RankGene software helps the evolutionary algorithm by cutting off the less predictive gene in the whole set of genes. The reduced size of initial genes also reduces the search space for the evolutionary algorithm. This creates more chance to the algorithm in order to discover only the predictive in the small search space. From the results of many experiments on different parameters, the initial size of 100 gives the best performance on the leukemia dataset. Furthermore, the evolutionary algorithm discovers some predictors that give 100% correctly classified on all dataset. The initial gene pool constructed by the RankGene software lead to the improved classification of the evolutionary algorithm. All ranking methods provided by the RankGene software give similar performance results on leukemia and NCI60 datasets.

The different behaviour in the different dataset is that the algorithm converge more slower on the NCI60 dataset. The reason is not only the larger size of training samples (61 samples in NCI60 vs. 38 samples in leukemia dataset), but also the difficulty of classifying large number of classes(9 classes in NCI60 vs. 3 classes in leukemia dataset). The KNN classifier can not classify all NCI60 samples correctly. This eventually helps the overall system to be more diversity. But the diversity in the system is still not enough to improve the classification performance on the NCI60 dataset.

5.3 The discrimination method

Another topic to address in the experiment is the number of genes to be included in a predictor. Several research have tried to find the optimal predictive genes set. Due to the large search space, different researches have obtained different optimal genes set.

The researches by Li and Grosse (2003) suggested that for the leukemia dataset that contains only 38 training samples, the number of predictive genes included in a predictor should be less than 50. The discrimination method is applied in order to rank the frequently selected genes in the best predictors and use the statistical z-score analysis. On the leukemia dataset, the 30 top-ranked genes are obtained to be included to the predictor. The performance estimated by the .632 bootstrap confirms that these top-ranked genes can distinguish different classes of the samples.

However, the use of z-score analysis still give the large number of genes compare to the work by Deutsch (2003). Therefore, more genes added into the predictors might be more useful because the predictor will be less sensitive to the quality of data, since the current microarray technology still provides data with variances.

Chapter 6

Conclusion

The results of the microarray multiclass classification using an evolutionary algorithm are satisfactory on two datasets used in the dissertation. The evolutionary algorithm can improve the performance of classification when the RankGene software is used to build the initial gene pool. With the concept of the evolutionary algorithm, the solutions are lead to better performance over the generations. By use of the .632 bootstrap estimation, the evolutionary give the satisfy result on the leukemia dataset. In contrast with the NCI60 dataset where the classification performance has to be improved.

The number of genes included in a predictor is another issue that plays important rule to classification task. The method presented in the dissertation is to use the z-score analysis for ranking the most frequently selected genes that are found in the best predictors over 100 trials of the experiment. The discrimination method reduce the number of predictive genes in the predictor and also improves the classification performance.

The improvement of the evolutionary algorithm may be done by tuning up the parameters within the evolutionary algorithm such as designing new scoring function that can lead the solution to the decided goal and examining the mutation probabilities to give the flexibility to the algorithm.

The future work apart will investigate whether or not the predictors have biological significance. The correlation between genes and the related concept of gene clusters are also an interesting issue to study.

Bibliography

Bao, H. T. (2004). Knowledge discovery and data mining techniques and practice. http://www.netnam.vn/unescocourse/knowlegde/knowlegd.htm.

Braga-Neto, U. M. and Dougherty, E. R. (2004). Is cross-validation valid for small-sample microarray classification? *Bioinformatics*, 20(3):374–380.

Dasgupta, D. and Michalewicz, Z. (1997). *Evolutionary algorithms in engineering applications*, chapter Evolutionary Algorithms - An Overview, pages 3–23. Springer-Verlag Berlin Heidelberg.

Deb, K. and Reddy, A. R. (2003). Reliable classification of two-class cancer data using evolutionary algorithms. *Biosystems*, 72:111–129.

Deutsch, J. M. (2003). Evolutionary algorithms for finding optimal gene sets in microarray prediction. *Bioinformatics*, 19(1):45–52.

Dudoit, S., Fridlyand, J., and Speed, T. P. (2000). Comparison of discrimination methods for the classification of tumors using gene expression data. Technical report 576, Mathematical Sciences Research Institute, Berkeley, CA.

Fogel, D. B. (1994). An introduction to simulated evolutionary optimization. In *Neural Networks*, volume 5, pages 3–13. Transactions, IEEE.

Ghanea-Hercock, R. (2003). *Applied evolutionary algorithms in Java*. Springer-Verlag New York, Inc.

Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., and Caligiuri, M. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537.

Keedwell, E. and Narayanan, A. (2002). Genetic algorithms for gene expression analysis. *Applications of Evolutionary Computation: Proceedings of the 1st European Workshop on Evolutionary Bioinformatics (EvoBIO 2003)*, pages 76–86.

Li, L., Weinberg, C. R., Darden, T. A., and Pedersen, L. G. (2001). Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics*, 17(12):1131–1142.

Li, T., Zhang, C., and Ogihara, M. (2004). A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*.

Li, W. and Grosse, I. (2003). Gene selection criterion for discriminant microarray data analysis based on extreme value distributions. In *Proceedings of the seventh annual international conference on Computational molecular biology*, pages 217–223. ACM Press.

Nguyen, D. V. and Rocke, D. M. (2002). Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50.

Ooi, C. H. and Tan, P. (2003). Genetic algorithms applied to multi-class prediction for the analysis of gene expression data. *Bioinformatics*, 19(1):37–44.

Park, C. and Cho, S.-B. (2003). Genetic search for optimal ensemble of feature-classifier pairs in dna gene expression profiles. In *Neural Networks*, volume 3, pages 1702– 1707. Proceedings of the International Joint Conference on, IEEE.

Quckenbush, J. (2002). Microarray data normalization and transformation. *Nature Genetics*, 32(doi:10.1038/ng1032):496–501.

Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.-H., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J. P., Poggio, T., Gerald, W., Loda, M., Lander, E. S., and Golub, T. R. (2001). Multiclass cancer diagnosis using tumor gene expression signatures. *PNAS*, 98(26):15149–15154.

Raudys, S. (2001). *Statistical and Neural Classifiers: An Integrated approach to design*. Advances in Pattern Recognition. Springer.

Ross, D. T., Scherf, U., Eisen, M. B., Perou, C. M., Rees, C., Spellman, P., Iyer, V., Jeffrey, S. S., de Rijn, M. V., Waltham, M., Pergamenschikov, A., Lee, J. C.,

Lashkari, D., Shalon, D., Myers, T. G., Weinstein, J. N., and Brown, D. B. . P. O. (2000). Systematic variation in gene expression patterns in human cancer cell lines. *Nature Genetics*, 24(3):227–235.

Su, Y., Murali, T., Pavlovic, V., Schaffer, M., and Kasif, S. (2003). Rankgene: identification of diagnostic genes based on expression data. *Bioinformatics*, 19(12):1578–1579.

Xing, E. P., Jordan, M. I., and Karp, R. M. (2001). Feature selection for high-dimensional genomic microarray data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 601–608. Morgan Kaufmann Publishers Inc.

Yang, Y. H. and Thorne, N. P. (2003). *Science and Statistics: A Festschrift for Terry Speed*, volume 40 of *Monograph*, ims lecture notes Normalization for Two-color cDNA Microarray Data, pages 403–418.