Diffuse Large B-cell Lymphoma Classification Using Genetic Programming Classifier

Supoj Hengpraprohm and Prabhas Chongstitvatana
Department of Computer Engineering Chulalongkorn University, Thailand
supojn@yahoo.com, prabhas@chula.ac.th

Abstract – **Diffuse large B-cell lymphoma (DLBCL) is the most common subtype of non-Hodgkin's lymphoma. It is possible to classify normal and DLBCL patients using the data from cDNA microarrays technique that monitoring gene expression. Machine learning techniques are well-known methods for classification tasks. In this paper, we propose a Genetic Programming based method to generate classifiers with high accuracy. The proposed method employs cluster of classifiers to vote for the result. Furthermore, the classifier is presented in form of a mathematical equation which is amendable to human interpretation.**

I. Introduction

Diffuse large B-cell lymphoma (DLBCL) is the most common subtype of non-Hodgkin's lymphoma. Less than 50% of DLBCL patients respond well to current therapy and have prolonged survival [1]. Many researchers attempt to study their special feature and try to identify normal and DLBCL patients automatically.

Alizadeh et al. [2] showed that there is diversity in gene expression among the tumours of DLBCL patients using complementary DNA (cDNA) microarrays technique. They identified two molecularly distinct forms of DLBCL such as germinal centre B-like (GC B-like) DLBCL and activated B-like DLBCL.

Azuaje [3], [4] proposed an automated approach to prediction and discovery of classes of cancer based on the processing of gene expression data generated by Alizadeh et al. Using artificial neural learning known as Simplified Fuzzy ARTMAP (SFAM), it can provide an effective and efficient method for the prediction and discovery of cancer categories.

Many researchers [5], [6], [7] tried to develop cancer classification and clustering systems using machine learning techniques based on gene expression data. The systems are able to classify the data with high accuracy or cluster the data significantly. However, the knowledge embedded in these classifiers are difficult to understand by human.

In this paper, we used Genetic Programming to generate classifiers for identifying normal and DLBCL from data generated by Alizadeh. The classifier is formed as a mathematical formula which makes it more understandable for human. In conjunction with Genetic Programming, we have used cluster of classifiers with a voting strategy to improve the accuracy of classification.

The paper is organized as follows. Section II presents an introduction to Genetic Programming. Section III describes the data and method implemented in this research. Section IV shows the result of the experiment and conclusions are presented in Section V.

II. Introduction To Genetic Programming

Genetic Programming [8] is a search method that imitates natural evolution and natural selection. It is developed from Genetic Algorithms [9] and is differed by the way the solution is represented in a tree structure instead of a fixed length binary string. The solution comprises of nodes from a function set and a terminal set. A function set is a set of operators designed for the problems such as arithmetic operators, logical operators, control functions, etc. A terminal set is a set of operands of function such as constant, variable, etc. The algorithm of Genetic Programming is shown in Fig. 1 and details of each step are as follows:

A. Generate an initial population of solutions

The initial solutions are created to full the population. The structure of a solution is a tree. There will be a large variation of solution structures through the process of this random generation (Fig. 2).

B. Evaluate each solution by a fitness function

Each solution is evaluated to determine its fitness. The evaluation function, called "fitness function", is an important element in Genetic Programming. The fitness function is problem specific. For example, for a symbolic regression task, the fitness function usually is the minimization of prediction error in the training set. Each solution will have a measure of goodness associated with it.

C. Create a new population by genetic operators

Genetic operations on the population have the goal of generating a new population that has better quality solutions. There are three genetic operators: reproduction, crossover, and mutation.

Reproduction

A number of good solutions are selected to be reproduced to the next generation. This process conserves good solutions.

Crossover

This operator recombines parts from two good solutions, called "parents", to create new solutions, called "offspring". Two good solutions are selected, the probability of a solution being selected is proportional to its fitness. The crossover points, which determine the location to exchange parts, are randomly selected. The subtrees from parents are exchanged. This process creates two new offspring (Fig. 3).

Mutation

To maintain diversity in the population and to encourage exploration of different solutions, the mutation operator changes some part of a solution randomly. A solution is selected randomly and a location to be changed is selected. A part is mutated by replacing it with a small random tree (Fig. 4).

These steps are repeated until the termination criteria are met. The termination criterion for the run may be defined by the best fitness value or a maximum number of generations. Throughout generations, the quality of solutions is improved. The result from each run is different as the search for a solution is probabilistic and the solution for this problem is not unique.

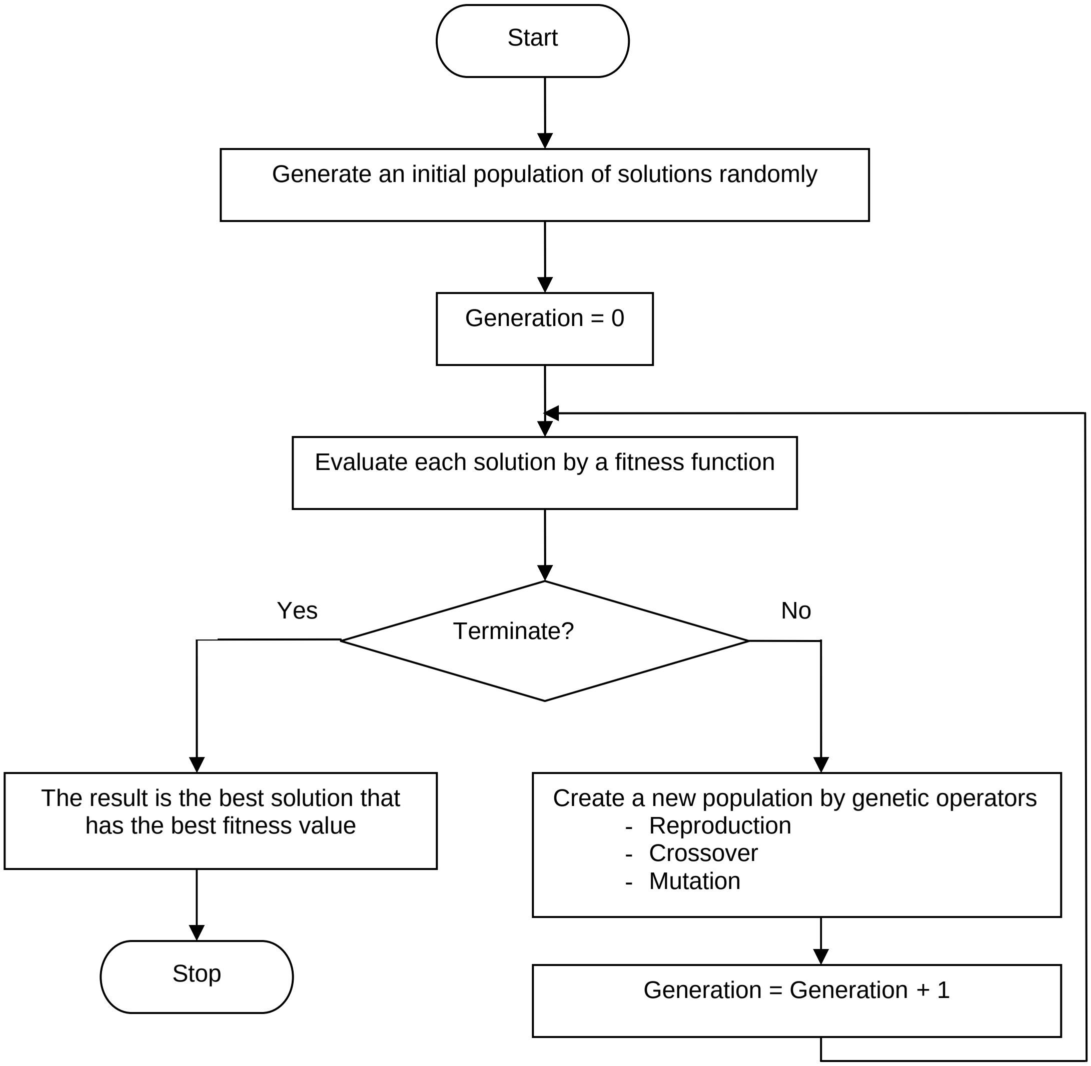
Start
Generate an initial population of solutions randomly
Generation = 0
Evaluate each solution by a fitness function
Terminate?
Yes
No
The result is the best solution that has the best fitness value
Stop
Create a new population by genetic operators
- Reproduction
- Crossover
- Mutation
Generation = Generation + 1

Fig.1. Genetic Programming algorithm

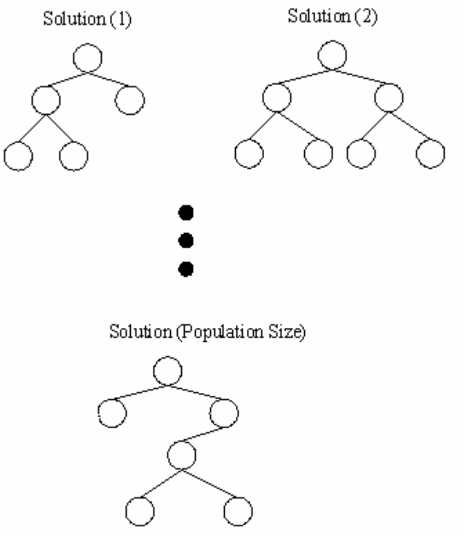

Fig.2. Solution in Population of Genetic Programming

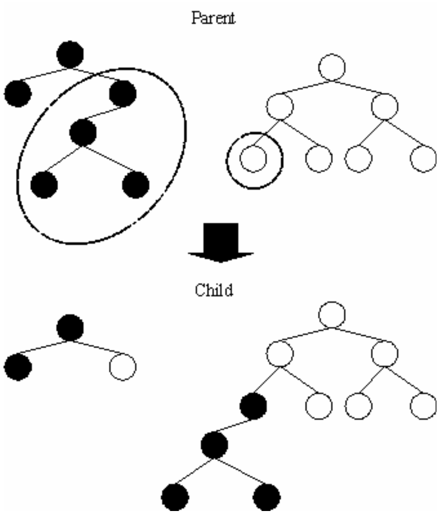

Fig.3. Crossover operator

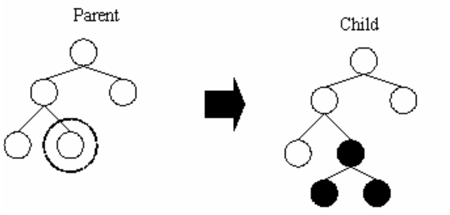

Fig.4. Mutation operator

III. The Data And Methods Implemented

In this research, we used the data generated by Alizadeh et al. [2]. The data is the expression levels from a number of genes using cDNA microarray technique. The data is described in Section A. Section B presents the feature of Genetic Programming classifier. Section C shows the details of parameters. The evaluation of the result will be discussed in Section D.

A. cDNA microarray data

The data consisted of 63 cases (45 DLBCL and 18 normal) described by the expression level of the genes: CD10, BCL-6, TTG-2, IRF-4 and BCL-2, which were used in the experiment of Azuaje [4]. These 5 genes have 13 environments in expression levels. Their values are defined as follows:

$$gene\_\exp ression = \log_2 \frac{Int(Cy5)}{Int(Cy3)}. \qquad (1)$$

where Int(Cy5) and Int(Cy3) are the intensities of red and green colors which are scanned after the hybridization of the samples with the arrayed DNA probes. The full data and experimental methods are available on the web of Alizadeh et al. (http://llmpp.nih.gov/lymphoma)

B. Genetic Programming classifier

A classifier is represented by a classification tree. The tree represented an arithmetic equation (Fig 5). It consists of symbols from the function set *F* and the terminal set *T*. The function set *F* comprises of arithmetic operators and the terminal set *T* comprises of 10 constants and 13 variables defined as follows: $F = \{+, -, *, /\}$ and $T = \{ 0.. 9,\ x1.. x13 \}$. The variables represent the value of the expression level of genes in each environment. Each variable is defined in Table I.

TABLE I
Definition Of Variable Used In GP Classifier

Variable	Gene	Clone_ID
x1	CD10	200814
x2	CD10	1286850
x3	CD10	701606
x4	BCL-6	712395
x5	BCL-6	1340526
x6	TTG-2	712829
x7	TTG-2	685456
x8	IRF-4	270770
x9	IRF-4	1272196
x10	BCL-2	232714
x11	BCL-2	342181
x12	BCL-2	1336385
x13	BCL-2	342181

C. Parameters and Methods

The parameters of Genetic Programming runs used in the experiment are shown in Table II. Initial population of equations is generated randomly. Genetic operators are applied to create a new generation of population as described in Section 2. The details of the process are as follows:

Fitness Function

To evaluate the fitness of a candidate, its equation is evaluated. Its variables are instantiated as follows. The variables (x1-x13) are from the cDNA microarray data. If the result of evaluating an equation is more than 0, it will be classified as Class 1 (DLBCL group). Otherwise it will be classified as Class 2 (normal group). An equation is evaluated with data in the training set and the total number of the correct classification is counted as the fitness value of the equation. The term 1/size is included as a penalty for a large solution and to encourage a compact solution. The higher fitness value indicates the better solution. The fitness function defined as follow:

$$fitness\_value = Total(Correct\_Classification) + \frac{1}{Size(Tree)} \qquad (2)$$

Selection

The Tournament Selection [8] is used in the experiment. The tournament size is 20.

Reproduction

The top 10% of high fitness value individuals will be selected to create a new generation.

Crossover

Two equations are selected with the selection method as described above. The crossover points are selected randomly to swap the structure of each equation at the crossover points. After crossover, if the size of equation is within the limit of the maximum size, it will be accepted to be in the new generation. Otherwise, it will be discarded. Crossover is repeated until the offspring are created equal to the target number.

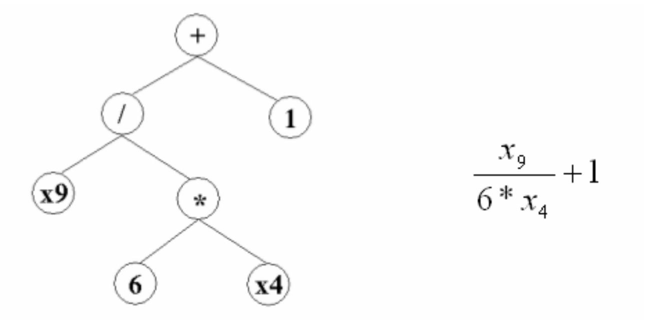

Fig.5. (left) The tree represented an arithmetic equation
(right) The equation derived from the tree

TABLE II
THE PARAMETERS WHICH BE USED TO CREATE THE CLASSIFIERS

Population Size	1,000
Maximum Size of Tree	Not more than 20 times of the number of variables (20 x 13 = 260) nodes.
Maximum number of Generation	500
Reproduction Rate	10%
Crossover Rate	80%
Mutation Rate	10%
Termination Criteria	Correctly classify the training data 100% or exceed the maximum number of generations

Mutation

An equation is selected and mutation is applied. There are two types of mutation, they are defined as follows:

1) *Type1 Structure change mutation:* a node in a tree is selected and replaces with a terminal node (0-9 or x1-x13) randomly chosen.
2) *Type2 Value change mutation:* a node is selected and replaces with a random choice from the symbol set of its own type. For example: if the value of node selected is an operator, it will be replaced by a value in function set randomly chosen.

D. Evaluation criteria

To evaluate the performance of a classifier, we used a method known as round robin or leave one out method [10]. There are 63 records of data, 62 records are used as training set and one record is used as a test. We exchange a test data through to 63 records and evaluate an equation in terms of its accuracy, sensitivity and specificity which are defined as follows:

$$Accuracy = \frac{(TP + TN)}{N} \qquad (3)$$

$$Sensitivity = \frac{TP}{(TP + FN)} \qquad (4)$$

$$Specificity = \frac{TN}{(TN + FP)} \qquad (5)$$

where N is a total number of tested cases, TP is a total number of DLBCL subjects correctly classified, TN is a total number of normal subjects correctly classified, FP is a total number of normal subjects classified as DLBCL and FN is a total number of DLBCL subjects classified as normal.

Accuracy indicates the effectiveness of a classifier for classifying all data correctly. Sensitivity indicates the effectiveness of classifier to classify DLBCL data correctly. Specificity indicates the effectiveness of a classifier for classifying normal data correctly.

IV. Experimental Result

Each run of the experiment consists of 63 runs of different training data. A genetic programming run consists of 63 runs, each run generates a best classifier for its training set. Genetic Programming is a randomized algorithm and the result of each run is different. The experiment is repeated 10 times and averaged value of accuracy, sensitivity and specificity. To improve the accuracy further we used a majority voting strategy from a number of different classifiers, all of which were from different run. The number of classifiers is varied from 1, 3 and 5. The result is shown in Table III. The machine used in the experiment is a PC with 1.5GHz processor, 512 Mbytes memory. Each run takes on the average, 190 seconds. The total amount of time of the whole experiment is 190×63×10 = 119700 seconds or 33.25 hours.

We compare this result with the result of Azuaje's experiment in the best case (vigilance value (ρ) is equal to 0.95) and found that the Genetic Programming classifier is more effective to classify the data. The best result comes from using 5 classifiers to vote. The comparison of results is shown in Fig 6. An example of a Genetic Programming classifier is shown in Fig 7. Fig 8 shows the same classifier in a mathematical equation form. The average size of a solution is 89.5 nodes.

TABLE III
THE EXPERIMENTAL RESULT

Number of Classifiers	Accuracy (%)	Sensitivity (%)	Specificity (%)
1	78.72	83.10	67.77
3	84.91	88.88	74.99
5	88.09	92.22	77.77

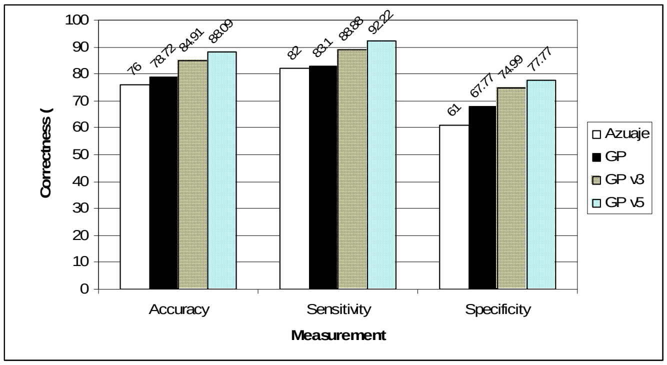

Fig. 6. The comparison of experimental result

```
If ( ( ( x13 - ( ( 4 - ( ( ( x3 - 1 ) + ( x2 * x7 )
      ) - x9 ) ) * x12 ) ) + 9 ) + ( ( ( ( x9 * 6 )
      + 9 ) / ( ( ( x13 * 6 ) + ( x10 / x6 ) ) + 1 )
      ) - x8 ) ) > 0
Then
      Class 1: DLBCL
Else (include divide by 0)
      Class 2: normal
End If
```

Fig. 7. An example of a Genetic Programming classifier

V. Discussion and Conclusion

These results suggest that Genetic Programming classifiers can be useful for classifying DLBCL out of normal data. Furthermore, when a majority voting strategy is used in conjunction with Genetic Programming classifiers, the correctness of classification is improved. However, it takes more computation time to create many classifiers.

When a classifier is presented in an equational form, its meaning is more amendable to human interpretation. The equation shows the relation of expression of each gene. These relationships may help us to understand which gene is important for the treatment of the disease.

$$\left[[[\, x13 - [[\, 4 - [[(x3 - 1) + (x2 \cdot x7)] - x9]\,] \cdot x12]\,] + 9] + \left[\left[\frac{[(x9 \cdot 6) + 9]}{\left[\left[(x13 \cdot 6) + \left(\frac{x10}{x6} \right) \right]\right] + 1} \right] - x8 \right]\right]$$

Fig. 8. A Genetic Programming classifier in a mathematical equation form

REFERENCES

[1] Margaret A. Shipp et al., “Diffuse large B-cell lymphoma outcome prediction by gene expression profiling and supervised machine learning”, Nature Medicine, vol. 8, no. 1, Jan. 2002, pp. 68 – 74.

[2] Ash A. Alizadeh et al., “Distinct type of diffuse large B-cell lymphoma identified by gene expression profiling”, Nature, vol. 403, 3 Feb. 2000, pp. 503 – 511.

[3] Francisco Azuaje, “Making Genome Expression Data Meaningful: Prediction and Discovery of Classes of Cancer Through a Connectionist Learning Approach”, Proceeding of the IEEE Symposium on Bio-Informatics and Biomedical Engineering (BIBE), 2000, pp. 208 – 213.

[4] Francisco Azuaje, “A computational neural approach to support the discovery of gene function and classes of cancer”, IEEE Transaction on Biomedical Engineering, vol. 48, 2001, pp.332 – 339.

[5] Chanho Park and Sung-Bae Cho, “Evolutionary Ensemble Classifier for Lymphoma and Colon Cancer Classification”, The 2003 Congress on Evolutionary Computation, vol.4, 2003, pp.2378 – 2392.

[6] Sung-Bae Cho and Hong-Hee Won, “Machine Learning in DNA Microarray Analysis for Cancer Classification”, the Proceedings of the First Asia-Pacific bioinformatics conference on Bioinformatics 2003, vol. 19, 2003, pp.189 – 198.

[7] Jin-Hyuk Hong and Sung-Bae Cho, “Lymphoma Cancer Classification Using Genetic Programming with SNR Features”, Proceeding of Genetic Programming: 7th European Conference, vol. 3003, 2004, pp.78 – 88.

[8] Koza, J. , “Genetic Programming”, MIT Press, 1992.

[9] Holland, J., “Adaptation in Natural and Artificial System”, Ann Arbor, Michigan : University of Michigan Press, 1975.

[10] F. Tourassi and C. Floyd, “The effect of data sampling on the performance evaluation of artificial neural networks in medical diagnosis”, *Medical Decision Making,* 17, 186-192, 1997