Expert Systems with Applications xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Gene selection and sample classification on microarray data based on adaptive genetic algorithm/*k*-nearest neighbor method

Chien-Pang Lee[a], Wen-Shin Lin[a], Yuh-Min Chen[b], Bo-Jein Kuo[a,*]

[a] *Biometry Division, Department of Agronomy, National Chung Hsing University, No. 250, Kuo Kuang Rd., Taichung 40227, Taiwan, ROC*
[b] *School of Nursing, China Medical University, No. 91, Hsueh Shih Rd., Taichung 40402, Taiwan, ROC*

ARTICLE INFO

Keywords:
Gene selection
Sample classification
Adaptive genetic algorithm
k-Nearest neighbor
Microarray data

ABSTRACT

Recently, microarray technology has widely used on the study of gene expression in cancer diagnosis. The main distinguishing feature of microarray technology is that can measure thousands of genes at the same time. In the past, researchers always used parametric statistical methods to find the significant genes. However, microarray data often cannot obey some of the assumptions of parametric statistical methods, or type I error may be over expanded. Therefore, our aim is to establish a gene selection method without assumption restriction to reduce the dimension of the data set. In our study, adaptive genetic algorithm/*k*-nearest neighbor (AGA/KNN) was used to evolve gene subsets. We find that AGA/KNN can reduce the dimension of the data set, and all test samples can be classified correctly. In addition, the accuracy of AGA/KNN is higher than that of GA/KNN, and it only takes half the CPU time of GA/KNN. After using the proposed method, biologists can identify the relevant genes efficiently from the sub-gene set and classify the test samples correctly.

1. Introduction

Since microarray technology has widely used on the study of gene expression in cancer diagnosis, much attention has been directed to microarray data analysis. Compared to traditional techniques, microarray technology can find patterns of normal and abnormal tissues (or cells) more easily and quickly because it can measure thousands of genes in an experiment. However, research funds for microarray technology are often limited and the sample size of microarray data is usually less than its gene size (Alon et al., 1999; Golub et al., 1999; Khan et al., 2001).

In order to identify the relevant and significant genes in microarray studies, researchers often use parametric statistical methods to test the significant genes found by cluster analysis. For instance, *t*-test and permutation *t*-test were used in the early days of microarray analysis (Dudoit, Yang, Callow, & Speed, 2002). However, there are some problems with using parametric statistical methods to analyze the microarray data. First, microarray data often cannot obey some of the assumptions of parametric statistical methods such as the assumption of normal distribution or independent. Second, microarray data contain thousands of genes, so the type I error will be over expanded. For those reasons, several statistical approaches have been performed to identify the differentially expressed genes in multiple testing, i.e., simultaneous test for each gene of the null hypothesis of equal expression. Dudoit et al. (2002) successfully used the method of maxT adjusted *p*-value to analyze the mice apo AI data. The adjusted *p*-value for the multiple testing procedures controlled the family-wise type I error rate. However, type I error may still be over expanded due to the high dimensionality of microarray data. Therefore, some researchers exploited the joint behavior of genes and sought clusters of genes to discriminate between normal and tumor tissue samples (Li, Darden, Weinberg, Levine, & Pedersen, 2001; Li, Pedersen, Darden, & Weinberg, 2002). Li et al. employed genetic algorithm/*k*-nearest neighbor (GA/KNN) to analyze the microarray data with good results. Owing to the slow evolution speed of GA, many methods have been created to improve GA such as KGA (Kuncheva & Jain, 1999) and IGA (Ho, Shu, & Chen, 1999). Our arm is to develop a method that can be used without assumption restriction and can provide dimension reduction for the microarray data. We call the method adaptive genetic algorithm/*k*-nearest neighbor (AGA/KNN).

There are some reasons to use AGA and KNN. First, most variable selection methods need some assumptions, and they are not suitable to use for high-dimensional space. AGA is a search tool of machine learning and imitates the biological system to find the near optimal solution, so it is suitable to analyze high-dimensional, noisy data. Second, KNN is one of the most widely used classification techniques (Kuncheva, 1995) due to its simplicity and effectiveness (Ho, Liu, & Liu, 2002; Kuncheva & Bezdek, 1998; Kuncheva & Jain, 1999). Each sample can be classified according to the classification of its *k* nearest neighbors which are determined by

* Corresponding author. Tel.: +886 422840777x201; fax: +886 22850886.
E-mail address: bjkuo@nchu.edu.tw (B.-J. Kuo).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.07.053

Euclidean distance (Enas & Choi, 1986). Unlike many other classifiers which assuming a multivariate normal distribution of the feature values (Raymer, Punch, Goodman, Sanschagrin, & Kuhn, 1997), KNN does not depend on the data following any particular distribution.

Although various research domains of AGA and KNN have been well characterized (Li et al., 2001a; Srinivas & Patnaik, 1994), this is the first reported use of the combination to analyze microarray data.

The paper is organized as follows: first, the data sets used in this study and AGA/KNN method are described; second, the results of gene selection and sample classification, and comparison of AGA/KNN and GA/KNN are reported; finally, the conclusion and future work are given.

2. Material and method

2.1. Data sets

In this study three data sets were used to validate the performance of AGA/KNN. The first data set is an original colon data which is a high-density oligonucleotide chip (Alon et al., 1999). This set consists of 2000 gene expression levels for each sample. There are 62 samples of 40 tumor tissues and 22 normal tissues. But five samples (N34, N36, T30, T33, and T36) were identified as likely to have been contaminated and were removed (Li, Weinberg, Darden, & Pedersen, 2001b). We log-transformed the data and then divided them into a training set (the first 40 samples) and a test set (the remaining 17 samples).

The second data set is mice apo AI data which is a cDNA chip (Callow, Dudoit, Gong, Speed, & Rubin, 2000). This data set contains 6,384 gene expression levels for each sample. There are 8 apo AI knockout mice and 8 normal mice (reference mice) samples. Because this data set only had 16 samples, a cross-validation strategy was used.

In order to verify that AGA/KNN can distinguish the data set with several categories, the cDNA microarray data of the small, round blue cell tumors (SRBCTs) of childhood (Khan et al., 2001) was also used. The SRBCTs data set includes four distinct diagnostic cancers such as neuroblastoma (NB), rhabdomyosarcoma (RMS), nonHodgkin lymphoma (NHL) and the Ewing family of tumors (EWS). Because the four distinct cancers are similar on tumor histology, it is difficult to distinguish among them (Khan et al., 2001). This data set contains 2,308 gene expression levels for each sample. There are 63 training samples of 23 tumors (13 EWS and 10 RMS) and 40 cell lines (10 EWS, 10 RMS, 12 NB and 8 NHL). Twenty test samples contain 14 tumors (5 EWS, 5 RMS and 4 NB) and 6 cell lines (1 EWS, 2 NB and 3NHL).

2.2. Adaptive genetic algorithm/k-nearest neighbor (AGA/KNN)

Holland (1975) first proposed genetic algorithm (GA) which imitated the natural evolution and selection (Li et al., 2001a). This algorithm can find the near optimal solution through imitation of the biological evolution system (Goldberg, 1989; Liu et al., 2004). The evolution speed of GA may be slow, so we developed AGA by the addition of three methods: (1) elitist strategy, (2) adaptive probabilities of crossover and mutation, and (3) extinction and immigration strategy.

Because GA just finds the nearest optimal solution, the best string of each run is often not the same, especially in high-dimensional space. To solve this problem, we generated many near optimal solutions by running the AGA/KNN procedure repeatedly and then computed the frequency of the results. The dimension of the data set was reduced by the rank of the gene frequency. There are five major components of AGA: encoding, initial population, fitness function, genetic operators (this component contains selection and the adaptive probabilities of crossover and mutation), and termination. The following subsections will describe the details of AGA. A flow chart of AGA/KNN is shown in Fig. 1.

2.2.1. Encoding

Each gene has to be encoded and transformed into characters before performing AGA/KNN. There are many encoding methods available but we used binary code in this study to allow each string to contain all combinations of genes (characters). When encoding with binary code, if a character was encoded to 1, that character would be sent to the fitness function and computed; otherwise that character would not be computed. A string is made of all characters and is analogous to a chromosome in the biological system, and a character of a string is analogous to a gene in a chromosome.

2.2.2. Initial population

Because AGA/KNN uses the binary code, $2^g - 1$ string would be generated where g is the gene size of that data set; thus AGA/KNN will need more CPU time to train when all strings are used. Therefore, 150 strings were selected into the initial population at random to decrease the computation time.

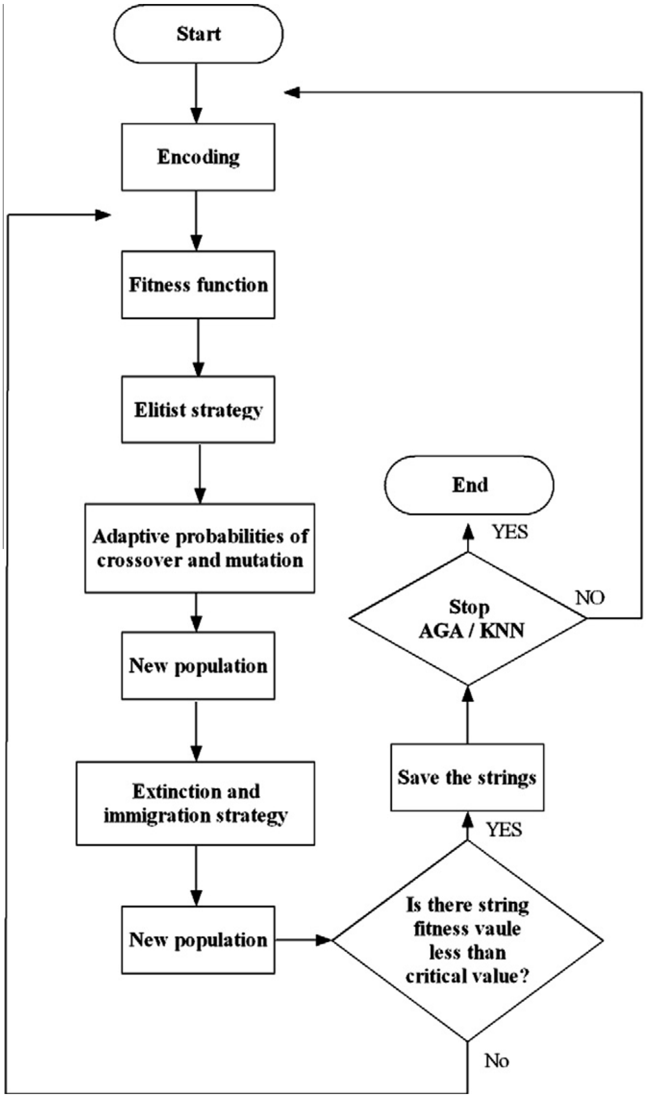
Start
Encoding
Fitness function
Elitist strategy
Adaptive probabilities of crossover and mutation
New population
Extinction and immigration strategy
New population
Is there string fitness vaule less than critical value?
No
YES
Save the strings
Stop AGA / KNN
NO
YES
End

Fig. 1. The flow chart of AGA/KNN.

Please cite this article in press as: Lee, C.-P., et al. Gene selection and sample classification on microarray data based on adaptive genetic algorithm/k-nearest neighbor method. *Expert Systems with Applications* (2010), doi:10.1016/j.eswa.2010.07.053

2.2.3. Fitness function

The fitness function is analogous to the theory of "survival of the fittest". The survival rate (or fitness value) of a string is computed based on the fitness function. We aim to find the minimum solution set for the relevant gene. Therefore, the fitness function, analogous to the definition of the report of Hernandez, Duval, and Hao (2008), was determined by

$$\text{fitness} = (1-K) + \frac{S}{g}, \tag{1}$$

where K is the classification rate determined by KNN; S is the total number of characters equal to 1 for each string; and g is the gene size of the data set. Thus, the lower the fitness value of a particular string is, the better the survival rate of that string is. Furthermore, for cancer diagnosis, if two selected gene subsets have the same classification ability, the smaller gene size is preferred (Hernandez et al., 2008). In addition, for the process of AGA/KNN, the decision of the fitness value is critically important. A plot of learning trend, the fitness value calculated in AGA/KNN through the training generations, is shown in Fig. 2. According to the limitation of the fitness value for 1000 training generations in AGA/KNN, the critical value of the fitness function in our study was set to 0.27.

2.2.4. Genetic operators

2.2.4.1. Selection and elitist strategy. When the initial population does not contain any string with fitness values less than the critical value of the fitness function, a new population (or second generation) must be generated through genetic operators. Basically, the roulette wheel method of selection is often used in the first step of the genetic operators. However, if each string is randomly selected for the second generation, the fitness values of some strings may be too low. For this reason, De Jong (1975) used the elitist strategy based on the genetic adaptive system to make sure the best fitness value of each successive generation would not be worse than the previous generation. In this method, 20% of the strings with better fitness values would be saved and put into the new population without performing the other genetic operators. Besides, the roulette wheel method of selection was also used on the remaining 80% to generate the next generation.

2.2.4.2. Adaptive probabilities of crossover and mutation. The next two steps of the genetic operator are crossover and mutation. Crossover enhances the ability of evolution by exchanging the information of the parent generation; while mutation enhances the ability of evolution by introducing new characters into the strings. In fact, the fitness value of a string may become worse when either the crossover rate or mutation rate is too high. Because the crossover rate and the mutation rate of simple genetic algorithms (SGA) were not fluctuating rates, new crossover and mutation operators were determined according to the concept of De Jong (1975). Srinivas and Patnaik (1994) determined adaptive probabilities of crossover and mutation to improve the SGA technique. Consequently, the crossover rate in our study was determined analogously by

$$\begin{cases} P_c = \frac{k_1(f'-f_{\min})}{(\bar{f}-f_{\min})}, & f' \leqslant \bar{f}, \\ P_c = k_1, & f' > \bar{f}, \end{cases} \tag{2}$$

where $f_{\min}$ is the minimum fitness value of strings in the population; $\bar{f}$ is the mean fitness value of strings in the population; f' is the small fitness value between two strings which was performed with crossover operator; k_1 is a constant.

The mutation rate was determined by

$$\begin{cases} P_m = \frac{k_2(f'-f_{\min})}{(\bar{f}-f_{\min})}, & f' \leqslant \bar{f}, \\ P_m = k_2, & f' > \bar{f}, \end{cases} \tag{3}$$

where f' is the fitness value of strings which was performed with mutation operator; k_2 is a constant. As Srinivas and Patnaik (1994) suggested, the constants, k_1 and k_2, used for AGA/KNN were chosen as $k_1 = 1$ and $k_2 = 0.5$ to result in $0 \leqslant P_c \leqslant 1$ and $0 \leqslant P_m \leqslant 0.5$.

As mentioned before, SGA involved the non-fluctuating crossover and mutation rates to improve the convergence speed. However, the crossover and mutation rates were always fixed in SGA; hence the behavior of optimal string with high fitness value might become poorer if either rate was unsuitable. In the AGA/KNN process, we used the adaptive probabilities of crossover and mutation to improve the above problem. In other words, as the process of evolution in AGA/KNN fell into local optima or could not converge, the situation could be adapted opportunely by increasing the crossover rate and the mutation rate simultaneously.

2.2.4.3. Extinction and immigration strategy. The strings of the new generation will become similar to one another after several generations. In other words, the variance of the strings will be close to 0, and the evolution will be hold. Therefore, we developed an extinction and immigration strategy to increase the variation of the population. Yao and Sethares (1994) theorized that extinction and immigration strategy should behave like the mutation operator while the mutation rate was close to 1. Therefore, we determined that when the variance of the best fitness value of the previous 5 generations equals 0, new strings should be generated to replace the strings with fitness values that are larger than the mean fitness value of the population.

2.2.5. Termination

After performing the extinction and immigration strategy, AGA/KNN must verify that at least one string has a fitness value that is less than the critical value; if not, AGA/KNN must return to the fitness function (this is called a "loop"), as shown in Fig. 1; if so, those found strings would be saved. The path from "start" to "save the strings" in Fig. 1 is called a run. After saving the strings, AGA/KNN must confirm that the number of runs matches our requested number; if not, AGA/KNN must return to encoding; if so, AGA/KNN will stop execution. In the AGA/KNN procedure, a run can be generated by many loops, and a run can find one or more strings with fitness values that are less than the critical value. After repeating AGA/KNN run several times requested, many strings would be saved.

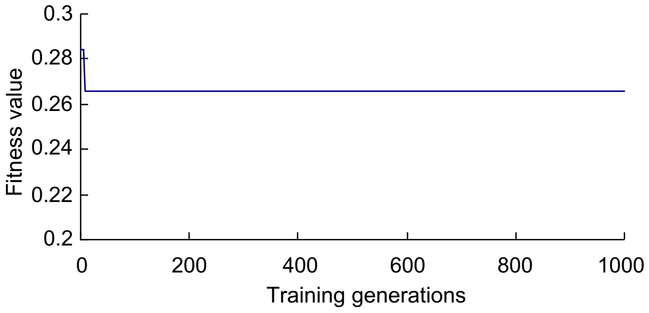

Fig. 2. The learning trend of fitness value in AGA/KNN.

3. Results and discussion

3.1. Gene selection for colon data

The training set of colon data (Alon et al., 1999) was analyzed by AGA/KNN. After repeating AGA/KNN by 100 runs, 133 strings were

Please cite this article in press as: Lee, C.-P., et al. Gene selection and sample classification on microarray data based on adaptive genetic algorithm/*k*-nearest neighbor method. *Expert Systems with Applications* (2010), doi:10.1016/j.eswa.2010.07.053

saved. The frequency of each gene was computed according to those 133 strings, and each gene was ranked through their frequency. We assume that the 50 most frequently appearing genes should contain the relevant genes. The following statistical approaches were employed to confirm it.

3.2. Visual display of results

Since cluster tree analysis can group genes or samples according to their similarity, we used this technique on the results with three gene groups to verify the classification ability of the selected gene. The three gene groups are described below:

- Gene group 1: The 50 most frequently selected genes of AGA/KNN.
- Gene group 2: The 50 smallest *p*-value genes of maxT adjusted *p*-value.
- Gene group 3: A random selection of 50 genes.

As the result of the cluster trees shown (see http://web.nchu.edu.tw/~bjkuo/AGA/), the gene group 1 using the 50 most frequently selected genes by AGA/KNN could resolve samples in the training set of colon data into two clusters (tumor and normal tissues samples) obviously. However, the gene group 2 using the 50 smallest *p*-value genes from MaxT adjusted *p*-value could not easily get the pattern of samples. The gene group 3 using 50 randomly selected genes did not find any patterns of samples at all.

Next, principal component analysis (PCA) was applied to the colon data. The plots of the first two components were used to reveal the patterns of the data set. We also used the 3 gene groups described previously to classify between the normal and tumor samples by projecting samples onto the plot of the first vs. second principal component.

Fig. 3 shows that the first two components explained approximately 67% of the total variation of gene group 1. Obviously, tumor and normal samples were separated, with the normal tissue samples on the top and tumor tissue samples on the bottom with one outlier, T3. In contrast, the first two components explained approximately 72% of the variation of gene group 2, and normal tissue samples and tumor tissue samples were separated with one outlier, N12 (Fig. 4). For Gene group 3 (Fig. 5), the first two components explained approximately 64% of the variation, and two distinct clusters separating the groups were not easily found.

The results of cluster tree analysis and PCA confirmed that the patterns of samples could not be found by using randomly selected genes.

3.3. Classification analysis

The ability of finding the patterns of samples by AGA/KNN could be better than maxT adjusted *p*-value and random selection had been shown in Section 3.2. Because visual display result is subjective, an objective approach should be presented to verify this result. Therefore, for gene selected by AGA/KNN on the colon (binary categories) and the SRBCTs (four categories) data sets, classification analyses, including KNN ($k = 3$) and support vector machine (SVM) (Brown et al., 2000), were employed to compare the classification rate.

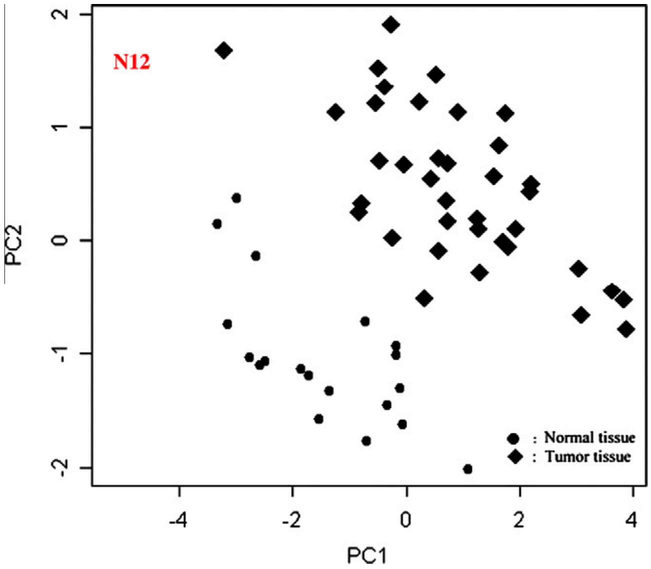

Fig. 4. Plot of the first vs. second principal component using the 50 smallest *p*-value genes of MaxT adjusted *p*-value. *Note:* N12 is an outlier.

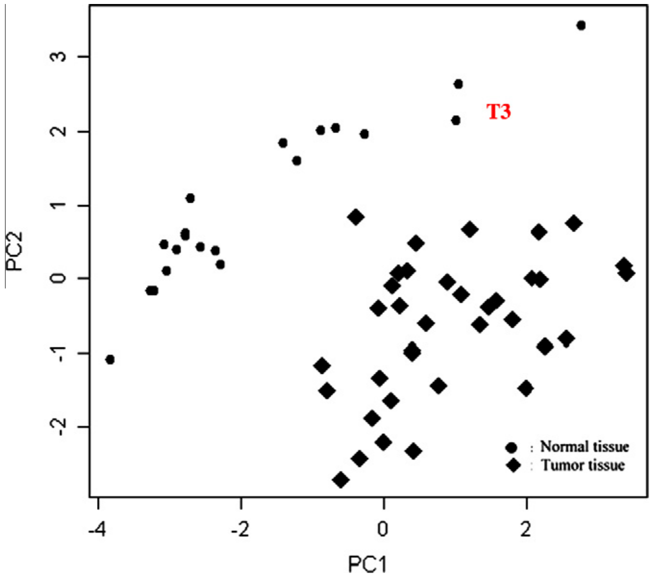

Fig. 3. Plot of the first vs. second principal component using the 50 most frequently selected genes by AGA/KNN. *Note:* T3 is an outlier.

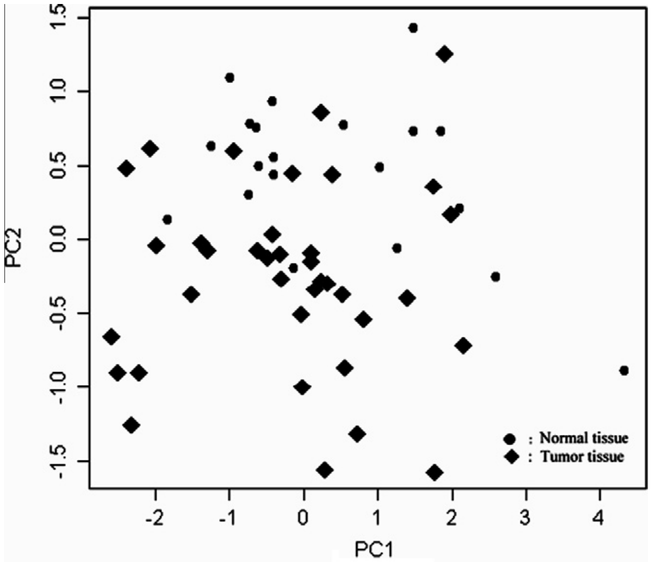

Fig. 5. Plot of the first vs. second principal component using 50 randomly selected genes.

Please cite this article in press as: Lee, C.-P., et al. Gene selection and sample classification on microarray data based on adaptive genetic algorithm/*k*-nearest neighbor method. *Expert Systems with Applications* (2010), doi:10.1016/j.eswa.2010.07.053

Firstly, KNN was employed to classify the samples. For the colon data set, all samples were classified correctly when using the 50 most frequently selected genes by AGA/KNN (Table 1). However, when only the most frequently selected gene was used to classify samples, 8 samples were classified correctly, and the classification rate was only 47.06%. Although the classification rate increased to 82.35% when all genes were used to classify samples, the classification rate dropped to 58.82% (11 samples were classified correctly) when using the 50 least frequently selected genes.

Table 2 shows that 8 samples were classified correctly when the gene with the smallest maxT adjusted p-value was used. Sixteen samples were classified correctly when the 50 smallest maxT adjusted p-value genes were used. Three samples were classified incorrectly when using all genes and the 50 largest maxT adjusted p-value genes, respectively.

For the validation, the classification rate of AGA/KNN using the 50 most frequently selected genes (Table 1) and the classification rate of maxT adjusted p-value using 50 smallest maxT adjusted p-value genes (Table 2) were 100% and 94.12%, respectively. This result also indicates that AGA/KNN performs better than maxT adjusted p-values. Furthermore, all samples were classified correctly when using the most frequently selected genes ranging from 50 to 250 (Fig. 6). This implies that those genes should contain the relevant genes. Hence, AGA/KNN could reduce the dimension of the data set more easily than maxT adjusted p-values and it could be used without assumption restriction.

Table 1
Classification of the test set for AGA/KNN.

Test set[a]	Experiment[b]	Top 1	Top 50	All genes	The least 50
T28	0	1	0	0	0
N28	1	1	1	1	1
N29	1	0	1	0	0
T29	0	0	0	0	1
T31	0	0	0	0	0
T32	0	0	0	0	1
N32	1	1	1	1	1
N33	1	0	1	1	1
T34	0	0	0	0	1
T35	0	0	0	1	1
N35	1	0	1	1	1
T37	0	1	0	0	0
T38	0	1	0	0	0
T39	0	1	0	1	1
N39	1	0	1	1	1
T40	0	0	0	0	1
N40	1	0	1	1	1
Classification rate		47.06%	100%	82.35%	64.71%

[a] "0" denotes a normal sample and "1" denotes a tumor sample.
[b] Original classification based on Alon et al. (1999).

Table 2
Classification of the test set for MaxT adjusted p-value.

Test set[a]	Experiment[b]	Smallest	50 Smallest	All genes	50 Largest
T28	0	1	0	0	0
N28	1	1	1	1	1
N29	1	0	1	0	0
T29	0	0	0	0	0
T31	0	0	0	0	0
T32	0	0	0	0	1
N32	1	1	1	1	1
N33	1	0	1	1	1
T34	0	0	0	0	0
T35	0	0	0	1	0
N35	1	0	1	1	1
T37	0	1	1	0	0
T38	0	1	0	0	0
T39	0	1	0	1	1
N39	1	0	1	1	1
T40	0	0	0	0	0
N40	1	0	1	1	1
Classification rate		47.06%	94.12%	82.35%	82.35%

[a] "0" denotes a normal sample and "1" denotes a tumor sample.
[b] Original classification based on Alon et al. (1999).

Subsequently, SVM was also used to verify the classification rate of the most frequently genes selected by AGA/KNN. For the colon data set (binary categories), the classification rate of SVM reaches to 100% when using 9 genes selected by AGA/KNN in both training and test data sets (not shown here). Likewise, for the SRBCTs data set (four categories), the classification rate could also achieve 100% when using 14 genes selected by AGA/KNN in both training and test data sets (Fig. 7). This result indicates that the most frequently selected genes by AGA/KNN can be applied to classify multiple categories successfully.

As the above results shown, AGA/KNN cannot only reduce the dimension of the data set but also have the excellent ability of classification regardless of using KNN or SVM method to classify the samples. Because the idea of AGA/KNN is based on GA/KNN, two algorithms were compared in the next subsection.

3.4. Comparison of AGA/KNN and GA/KNN

3.4.1. Accuracy of gene selection for mice apo AI data

Two algorithms were compared using mice apo AI data (Callow et al., 2000). From this data, eight significant genes (Table 3)

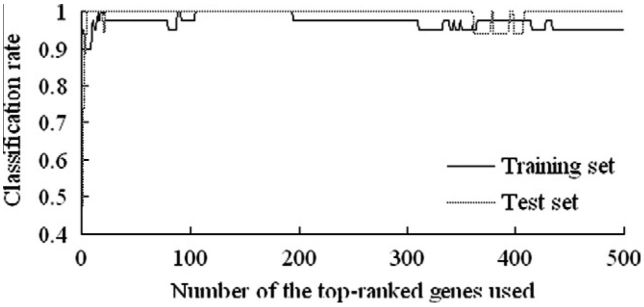

Fig. 6. The graph of classification rate using KNN vs. the number of the top-ranked genes selected by AGA/KNN for the training set and test set samples (colon data set).

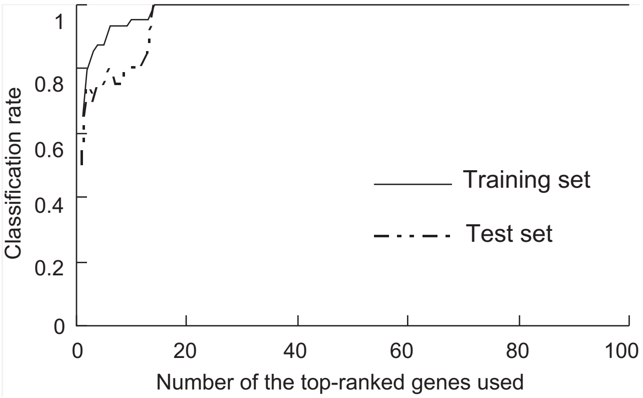

Fig. 7. The graph of classification rate using SVM vs. the number of the top-ranked genes selected by AGA/KNN for the training set and test set samples (SRBCTs data set).

Please cite this article in press as: Lee, C.-P., et al. Gene selection and sample classification on microarray data based on adaptive genetic algorithm/k-nearest neighbor method. *Expert Systems with Applications* (2010), doi:10.1016/j.eswa.2010.07.053

Table 3
Genes with MaxT adjusted p-value < 0.05. For mice apo AI knockout data.

Rank	Gene ID	Gene name
1	2149	Apo AI, lipid-Img
2	540	EST, highly similar to A
3	5356	CATECHOLO–METHYLTRAN
4	4139	EST, weakly similar to C
5	1739	ApoCIII, lipid-Img
6	2537	ESTs, highly similar to
7	1496	Est
8	4941	Similar to yeast sterol

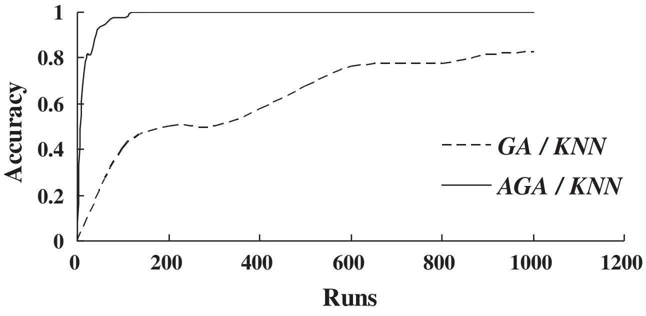

Fig. 8. Accuracy of AGA/KNN vs. GA/KNN.

previously identified and verified by Q–Q plot and maxT adjusted p-value (Dudoit et al., 2002) were used to compare the selection accuracy of genes selected by AGA/KNN and GA/KNN, respectively. For instance, AGA/KNN and GA/KNN selected 20 genes from the mice apo AI data individually and then the selection accuracy of those 20 genes was compared. As shown in Fig. 8, the accuracy of AGA/KNN was around 90% after 40 runs, and then it climbed to about 100% after 70 runs. In contrast, the accuracy of GA/KNN was only around 80% when more than 1,000 runs were executed. According to these results, it is obvious that the accuracy of AGA/KNN is better than that of GA/KNN. The parameters of AGA/KNN and GA/KNN are presented in Table 4.

3.4.2. Comparison of the CPU time

The selection accuracy of genes of AGA/KNN was found to be close to 100% after more than 120 runs and the selection accuracy of genes of GA/KNN was around 80% after more than 1000 runs. Therefore, for comparison, both algorithms were executed 10 times. The AGA/KNN algorithm was set to 120 runs while the GA/KNN was set to 1000 runs (Fig. 9). Since the mean termination time of AGA/KNN was about 10 min and that of GA/KNN was about 26 min, the evolution speed of AGA/KNN should be better than that of GA/KNN.

Table 4
The parameters of GA/KNN and AGA/KNN.

Operator	GA/KNN	AGA/KNN
Algorithm runs	50–1200	10–130
Encoding	Integer code	Binary code
Population size	150	150
String length	50	6226
Selection method	Roulette wheel	Roulette wheel
Crossover ratio	None	$0 \leqslant P_c \leqslant 1$
Mutation ratio	0.01	$0 \leqslant P_m \leqslant 0.5$
Elitist strategy	Preserving the best string	Preserving the 20% best strings
Extinction and immigration strategy	No	Yes

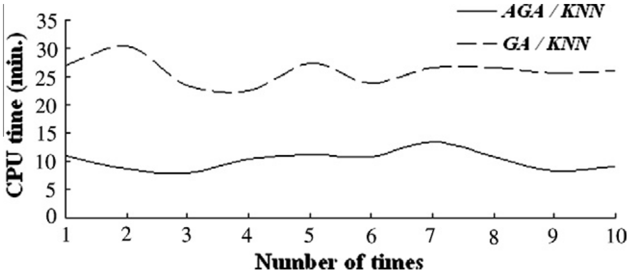

Fig. 9. CPU time of GA/KNN vs. AGA/KNN.

4. Conclusion and future work

The results of this study indicate that AGA/KNN can be provided as a useful tool with excellent performance for dimension reduction and gene selection on gene expression data. Comparing AGA/KNN and GA/KNN, it can be concluded that both algorithms are good for dimension reduction. However, the efficiency and classification rate of AGA/KNN is better than that of GA/KNN when the most frequently selected genes are used. Thus, researchers can use AGA/KNN to perform dimension reduction when analyzing microarray data. After using this proposed method, biologists can identify the relevant genes efficiently from the sub-gene set and classify the test samples correctly. The following recommendations will be made for further study: to confirm the correlation between runs and the dimension of the data set; and to confirm the correlation between the number of gene selected and the dimension of the data set.

References

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., et al. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probe by oligonucleotide arrays. In *Proceedings of the national academy of sciences of the United States of America* (Vol. 96, pp. 6745–6750).

Brown, M. P. S., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C. W., Furey, T. S., et al. (2000). Knowledge-based analysis of microarray gene expression data by using support vector machines. In *Proceedings of the national academy of sciences of the United States of America* (Vol. 97, pp. 262–267).

Callow, M. J., Dudoit, S., Gong, E. L., Speed, T. P., & Rubin, E. M. (2000). Microarray expression profiling identifies genes with altered expression in HDL-deficient mice. *Genome Research, 10*, 2022–2029.

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive system*. Doctoral dissertation. Ann Arbor: Department of Computer and Communication Sciences, University of Michigan.

Dudoit, S., Yang, Y. H., Callow, M. J., & Speed, T. P. (2002). Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica, 12*, 111–139.

Enas, G. G., & Choi, S. C. (1986). Choice of the smoothing parameter and efficiency of *k*-nearest neighbor classification. *Computers & Mathematics with Applications, 12A*, 235–244.

Goldberg, D. E. (1989). *Genetic algorithms in search optimization and machine learning*. New York: Addison-Wesley Publishing Company. pp. 1–88.

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

Hernandez, J. C., Duval, B., & Hao, J.-K. (2008). A study of crossover operators for gene selection of microarray data. *Lecture Notes in Computer Science, 4926*, 243–254.

Ho, S. Y., Shu, L. S., & Chen, H. M. (1999). Intelligent genetic algorithm with a new intelligent crossover using orthogonal arrays. In *GECCO-99: Proceedings of the genetic and evolutionary computation conference* (pp. 289–296).

Ho, S. Y., Liu, C. C., & Liu, S. (2002). Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. *Pattern Recognition Letters, 23*, 1495–1503.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press.

Khan, J., Wei, J. S., Ringnér, M., Saal, L. H., Ladanyi, M., Westermann, F., et al. (2001). Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine, 7*, 673–679.

Kuncheva, L. I. (1995). Editing for the *k*-nearest neighbors rule by a genetic algorithm. *Pattern Recognition Letters, 16*, 809–814.

Please cite this article in press as: Lee, C.-P., et al. Gene selection and sample classification on microarray data based on adaptive genetic algorithm/*k*-nearest neighbor method. *Expert Systems with Applications* (2010), doi:10.1016/j.eswa.2010.07.053

Kuncheva, L. I., & Bezdek, J. C. (1998). Nearest prototype classification: Clustering, genetic algorithms or random search. *IEEE Transactions on System, Man and Cybernetics Part C – Applications and Reviews, 28*, 160–164.

Kuncheva, L. I., & Jain, L. C. (1999). Nearest neighbor classifier: Simultaneous editing and feature selection. *Pattern Recognition Letters, 20*, 1149–1156.

Li, L., Darden, T. A., Weinberg, C. R., Levine, A. J., & Pedersen, L. G. (2001). Gene assessment and sample classification for gene expression data using a genetic algorithm/*k*-nearest neighbor method. *Combinatorial Chemistry & High Throughput Screening, 4*, 727–739.

Li, L., Pedersen, L. G., Darden, T. A., & Weinberg, C. R. (2002). Class prediction and discovery based on gene expression data. *Biostatistics branch and 2 laboratory of structural biology*. North Carolina: National Institute of Environmental Health Sciences, Research Triangle Park.

Li, L., Weinberg, C. R., Darden, T. A., & 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*, 1131–1142.

Liu, D., Shi, T., DiDonato, J. A., Carpten, J. D., Zhu, J., & Duan, Z. H. (2004). Application of genetic algorithm/*k*-nearest neighbor method to the classification of renal cell carcinoma. In Vicky Markstein (Ed.), *2004 IEEE computational systems bioinformatics conference (CSB'04)* (pp. 558–559).

Raymer, M. L., Punch, W. F., Goodman, E. D., Sanschagrin, P. C., & Kuhn, L. A. (1997). Simultaneous feature extraction and selection using a masking genetic algorithm. In T. Back (Ed.), *Seventh international conference on genetic algorithms (ICGA-97)* (pp. 561–567).

Srinivas, M., & Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on System, Man and Cybernetics, 24*, 656–666.

Yao, L. M., & Sethares, W. A. (1994). Nonlinear parameter estimation via the genetic algorithm. *IEE Transactions on Signal Processing, 42*, 927–935.