

Virtual Gene: Using Correlations Between Genes to Select Informative Genes on Microarray Datasets ^{*}

Xian Xu and Aidong Zhang

{xianxu|azhang}@cse.buffalo.edu

Department of Computer Science and Engineering
State University of New York at Buffalo, Buffalo, NY 14260, USA

Abstract. Gene Selection is one class of most used data analysis algorithms on microarray datasets. The goal of gene selection algorithms is to filter out a small set of informative genes that best explains experimental variations. Traditional gene selection algorithms are mostly single-gene based. Some discriminative scores are calculated and sorted for each gene. Top ranked genes are then selected as informative genes for further study. Such algorithms ignore completely correlations between genes, although such correlations is widely known. Genes interact with each other through various pathways and regulative networks. In this paper, we propose to use, instead of ignoring, such correlations for gene selection. Experiments performed on three public available datasets show promising results.

1 Introduction

Microarray experiments enable biologists to monitor expression levels of thousands of genes or ESTs simultaneously [1, 8, 15]. Short sequences of genes or ESTs tagged with fluorescent materials are printed on a glass surface. The slice is then exposed to sample solution for hybridization (base-pairing). mRNA molecules are expected to hybridize with short sequences matching part of their complement sequences. After hybridization the slice is scanned and goes through various data processing steps including image processing, quality control and normalization [4]. The resulting dataset is a two dimensional array with thousands of rows (genes) and tens of columns (experiments). Element at i^{th} row and j^{th} column in such an array is the expression level measure for gene i in experiment j . When tissue samples used in the experiments are labeled (e.g., sample is cancer tissue or normal tissue), sample classification can be performed on such dataset. New samples are classified based on their gene expression profiles.

Such dataset poses special challenge for pattern recognition algorithms. The main obstacle is the limited number of samples due to practical and financial concerns. This results in the situation where the number of features (or genes) well outnumbers

^{*} This research is partly supported by National Science Foundation Grants DBI-0234895, IIS-0308001 and National Institutes of Health Grant 1 P20 GM067650-01A1 All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation or the National Institutes of Health.

the number of observations. The term “curse of dimensionality” and “peaking phenomenon” are coined in the machine learning and pattern recognition community, referring to the phenomenon that inclusion of excessive features may actually degrade the performance of a classifier if the number of training examples used to build the classifier is relatively small compared to the number of features [12]. Typical treatment is to reduce the dimensionality of the feature space before classification using feature extraction and feature selection. Feature extraction algorithms create new features based on transformation and/or combination of original features while feature selection algorithms aim to select a subset of original features. Techniques like PCA and SVD have been used to create salient features [9, 10] for sample classification on microarray datasets. Feature selection, or in our case, gene selection generates a small set of informative genes, which not only leads to better classifiers, but also enables further biological investigation.

In order to find the optimal subset of features that maximizes some feature selection criterion function (we assume the higher value the criterion function, the better the feature subset), straightforward implementation would require evaluation of the criterion function for each feature subset, which is a classic NP hard problem. Various heuristics and greedy algorithms have been proposed to find sub-optimal solutions. Assuming independence between features, one attempt is to combine small feature subsets with high individual scores. This heuristic is widely used for gene selection. A class of gene selection algorithms calculates discriminative scores for individual genes and combines top ranked genes as selected gene set. We refer to this class of algorithms single gene based algorithms. Various discriminative scores have been proposed, including statistical tests (t-test, F-test) [3], non-parametric tests like TNoM [2], mutual information [18, 19], S2N ratio (signal to noise ratio) [8], extreme value distribution [13] and SAM [16] etc. Although simple, this class of algorithms is widely used in microarray data analysis and proven to be effective and efficient.

However, the assumption of independence between genes over simplifies the complex relationship between genes. Genes are well known to interact with each other through gene regulative networks. As a matter of fact, the common assumption of cluster analysis on microarray dataset [6] is that co-regulated genes have similar expression profiles. Bø [3] proposed to calculate discriminant scores for a pair of genes instead of each individual gene. Several of recent researches on feature selection especially gene selection [11, 17, 19] took into consideration the correlation between genes explicitly by limiting redundancy in resulting gene set. Heuristically, selected genes need to first have high discriminative scores individually and secondly not correlate much with genes that have already been selected. Generic feature selection algorithms like SFFS (sequential forward floating selection), SBFS (sequential backward floating selection), etc. have also been used for selecting informative genes from microarray datasets.

In this paper, we propose a totally different approach. Instead of trying to get rid of correlation in the selected gene set, we examine whether such correlation itself is a good predictor of sample class labels. Our algorithm is a supervised feature extraction algorithm based on new feature “virtual gene”. “Virtual genes” are linear combinations of real genes on a microarray dataset. Top ranked “virtual genes” are used for further analysis, e.g., sample classification. Our experiments with three public avail-

able datasets suggest that correlations between genes are indeed very good predictors of sample class labels. Unlike typical feature extraction algorithms, the “virtual gene” bears biological meaning: the weighted summation or difference of expression levels of several genes.

The rest of this paper is organized as follows. We present the concept of “virtual gene” and the “pairwise virtual gene” algorithm in Sec. 2. Both a synthetic and a real example from Alon dataset [1] are given. In Sec. 3, extensive experimental results are reported using three public available datasets. We give our conclusion and future work of this paper in Sec. 4.

2 Virtual Gene: A Gene Selection Algorithm

2.1 Gene Selection For Microarray Experiments

In this section we formalize the problem of gene selection for microarray datasets. Symbols used in this section will be used throughout this paper.

Let \mathcal{G} be the set of all genes that are used in one study, \mathcal{S} be the set of all experiments performed, \mathcal{L} be the set of sample class labels of interest. We assume $\mathcal{G}, \mathcal{S}, \mathcal{L}$ are fixed for any given study. Let $n = |\mathcal{G}|$ be the total number of genes, $m = |\mathcal{S}|$ be the total number of experiments and $l = |\mathcal{L}|$ be the total number of class labels. A gene expression dataset used in our study can be defined as $\mathcal{E} = (\mathcal{G}, \mathcal{S}, \mathcal{L}, E)$, where L is a list of sample class labels such that for $s \in \mathcal{S}$, $L(s) \in \mathcal{L}$ is the class label for sample s ; expression matrix E is an $n \times m$ matrix of real numbers. $E(g, s)$, where $g \in \mathcal{G}, s \in \mathcal{S}$, is the expression level of gene g in experiment s . For simplicity of presentation, we use a subscripting scheme to refer to elements in \mathcal{E} . Let $\mathcal{E}(G, S) = (G, S, L', E')$ where $G \subseteq \mathcal{G}$ and $S \subseteq \mathcal{S}$. L' is a sublist of L containing class labels for samples S , E' is the subarray of E containing values of expression levels for genes G and experiments S . We also write $E' = E(G, S)$. We further use $L(S)$ to denote a list of class labels for the set of experiments S . Given training expression data $\mathcal{E}_{train} = (\mathcal{G}, \mathcal{S}_{train}, \mathcal{L}_{train}, E_{train})$, the problem of sample classification is to build a classifier that predicts L_{new} for new experiment result $\mathcal{E}_{new} = (\mathcal{G}, \mathcal{S}_{new}, \mathcal{L}_{missing}, E_{new})$. $\mathcal{L}_{missing}$ indicates that the class labels of samples \mathcal{S}_{new} have not been decided yet. The problem of gene selection is to select a subset of genes $G' \subset \mathcal{G}$ based on \mathcal{E}_{train} so that classifiers built from $\mathcal{E}_{train}(G', \mathcal{S}_{train})$ predict L_{new} more accurately than classifiers built from \mathcal{E}_{train} . We use n' as the number of features being selected, or $n' = |G'|$.

2.2 An Example

Consider the following two examples as shown in Figure 1. In each figure, the expression levels of two genes are monitored across several samples. Samples are labeled either cancerous or normal. In both cases, the expression levels of the selected genes vary randomly across the sample classes. However, their correlation is a good predictor of class labels. *Virtual gene* expression level is obtained using the Def. 2. In the case of Alon [1] dataset, the expression levels of H09719 are generally higher than that of L07648 in cancer tissues. In normal tissues, on the contrary, L07648 expresses consistently higher except in one sample. Such correlations could be good predictors of

sample class labels. However, all feature selection algorithms listed in the previous section can not find and use such correlations. Single gene based algorithms will ignore both genes since neither of them is a good predictor of sample class labels in its own right. Correlation based algorithms will actually remove such correlations, should any of the genes have been selected.

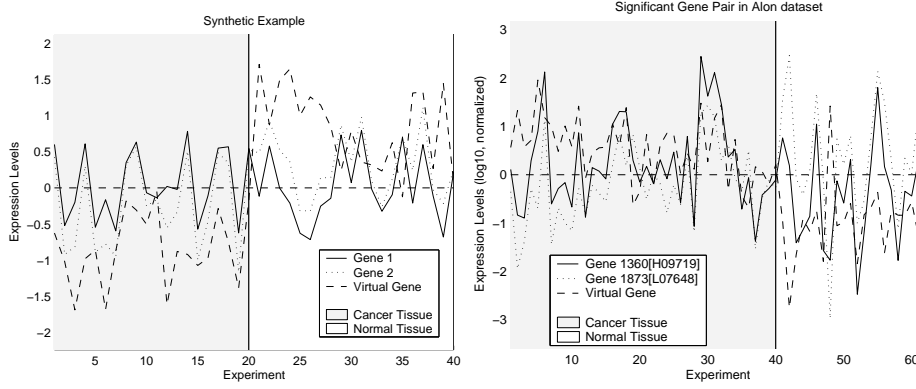


Fig. 1. Examples of gene pair being better predictor of class labels than single gene.

2.3 Virtual Gene Algorithm

Definition 1. *Virtual Gene* is a triplet $VG = (G_v, W, b)$ where $G_v \subseteq \mathcal{G}$ is a set of constituent genes, $|G_v| = n_v$, W is a matrix of size $n_v \times 1$, b is a numeric value. The expression levels of a *virtual gene* is determined using Definition 2.

Definition 2. (Virtual Gene Expression) Given a *virtual gene* $VG = (G_v, W, b)$ and gene expression matrix E , where $|G_v| = n_v$, E is an $n_v \times m_v$ expression matrix, the *virtual gene expression* VE of a virtual gene VG is a linear combination of expression matrix E . $VE(VG, E) = W^T \times E + b$, where W^T is the transpose of W .

A *virtual gene* is a triplet $VG = (G, W, b)$ as defined in Def. 1. Parameters W and b are chosen using FLD (fisher linear discriminant) to maximize linear separability between sample classes as listed in Algorithm 1. Discriminative power of a *virtual gene expression* with respect to sample classes can be measured using normal single gene based scores. We use t-score in this paper for this purpose. *Pairwise virtual gene* is a special case of *virtual gene* where the number of genes involved is limited to two. In this case, only the correlations between a pair of genes are considered. By limiting *virtual gene* to gene pairs, computation can be carried out efficiently. According to our experiments, it performs well on three public available datasets.

Definition 3. *Pairwise virtual gene and its expression* are special cases for *virtual gene* and its expression, where the number of genes involved is limited to two.

Algorithm 1 *gen_vg* : Calculating Virtual Gene From Training Data

Require: $\mathcal{E} = (G, S, L, E)$ as gene expression data.

Ensure: $VG = (G, W, b)$ as a virtual gene.

- 1: $(W, b) \leftarrow fld(E, L)$, (W, b) is the model returned by *fld* algorithm.
 - 2: **return** (G, W, b)
-

Exhaustive examination of all *pairwise virtual genes* requires $O(n^2)$ computation where n is the number of genes. For a large number of genes, exhaustive search of all gene pairs becomes inefficient. Such exhaustive search also invites unwanted noise since not all gene pairs bare biological meaning. For example, for genes that are expressed in different locations in a cell, in different biological processes, without biological interactions, their relative abundance may not be biologically significant. Ideally, only gene pairs with some biological interaction shall be examined. We approximate this using a gene clustering approach. Each gene cluster corresponds roughly to some biological pathways. By limiting search among the gene pairs from the same gene cluster, we not only focus ourselves on these gene pairs that are more likely to interact biologically, but also make our gene selection algorithm much faster.

Algorithm 2 *pairwise_vg* : Pairwise Virtual Gene Selection

Require: $\mathcal{E} = (G, S, L, E)$; k as the number of genes to be selected; $\alpha; \beta$

Ensure: VGS: as set of pairwise virtual genes $VG = (G, W, b)$

- 1: Initialize VGS to be an empty set. Initialize *pair_score* to be a sparse $n \times n$ array.
 - 2: Cluster genes based on their expression levels in E . Result stores in *Clusters*.
 - 3: **for** each gene cluster $G' \in Clusters$ **do**
 - 4: **for all** gene $g1 \in G'$ **do**
 - 5: **for all** gene $g2 \in G'$ and $g2 \neq g1$ **do**
 - 6: $vg \leftarrow gen_vg(\mathcal{E}((g1, g2), S))$
 - 7: $ve \leftarrow VE(vg, E((g1, g2), S))$
 - 8: $pair_score[g1, g2] \leftarrow t\text{-score}(ve, L)$
 - 9: **end for**
 - 10: **end for**
 - 11: **end for**
 - 12: **for** $i = 1$ to k **do**
 - 13: $(g1, g2) \leftarrow \underset{(g1, g2)}{\operatorname{argmax}}(pair_score[g1, g2])$
 - 14: $vg \leftarrow gen_vg(\mathcal{E}((g1, g2), S))$
 - 15: add vg to VGS
 - 16: multiply *pair_score* that involves $g1$ or $g2$ by α .
 - 17: multiply *pair_score* that involves genes in same cluster of $g1$ or $g2$ by β .
 - 18: $pair_score[g1, g2] \leftarrow$ minimum value
 - 19: **end for**
 - 20: **return** VGS
-

Algorithm 2 details the *pairwise virtual gene selection* algorithm. Genes are first clustered based on their expression levels. For each pair of genes in the same cluster,

virtual gene expression is calculated according to Def. 2. A single gene discriminative score with respect to the sample class labels is then derived from the virtual gene expression. All within-cluster pairwise virtual gene expression scores are calculated and stored for the next stage of analysis. The best scored virtual gene is then selected and pairwise scores are modified by two parameters. Pairwise scores of virtual genes that share constituent genes with the selected virtual gene are degraded by a constant α ranging $[0, 1]$. This dampens the effect of a single dominant salient gene. In the extreme case where α is set to 0, once a virtual gene is selected all virtual genes sharing constituent genes will not be further considered. The second parameter affecting the virtual gene selection is β , which controls how likely virtual genes in the same gene cluster are selected. Different gene clusters correspond to different regulative processes in a cell. Choosing genes from different gene clusters broadens the spectrum of the selected gene set. β also ranges $[0, 1]$. In the extreme situation where $\beta = 0$, only one virtual gene will be selected for each gene cluster. After modifying pairwise scores, the algorithm begins next loop to find the highest scored virtual gene. This process repeats until k virtual genes have been selected. For performance comparison of the *pairwise virtual gene* algorithm and single gene based algorithms, each *pairwise virtual gene* counts for two genes. For example, the performance of selecting 50 genes using single gene based algorithms would be compared to performance of selecting top 25 *pairwise virtual genes*.

2.4 Complexity of the Pairwise Virtual Gene Algorithm

The *pairwise virtual gene selection* algorithm runs in three stages: (1) cluster genes based on expression profile (lines 1-2), (2) calculate discriminative scores for the *pairwise virtual gene* (lines 3-11), and (3) select pairwise virtual genes with best discriminative scores (lines 12-20). We assume gene cluster number to be θ and n, m, k, α, β as discussed above.

In the first stage of analysis, k-means algorithm runs in $O(\theta n)$. In the second stage, the actual number of gene pairs examined is $O(\frac{n^2}{\theta})$, assuming gene clusters obtained in the previous stage are of roughly the same size. For each gene pair, the calculation of the pairwise virtual gene and its discriminative score require $O(m^2)$. Time complexity of the second stage is $O(\frac{m^2 n^2}{\theta})$. Stage three requires $O(k(\frac{n^2}{\theta} + m^2 + n + \frac{n}{\theta}))$ time. Putting them together, we have time complexity of $O(\theta n + \frac{m^2 n^2}{\theta} + k(m^2 + \frac{n^2}{\theta}))$. The most time consuming part in the previous expression is the term $O(\frac{m^2 n^2}{\theta})$. In our experiments, we choose $\theta \sim \Theta(n)$. Considering the fact that $k < n$, the time complexity of Algorithm 2 becomes $O(n^2 + nm^2)$. The $O(n^2)$ term is for k-means clustering, which runs rather quickly. If no clustering is performed in stage 1 (or $\theta = 1$, one gene cluster), the time complexity becomes $O(n^2 m^2 + kn^2)$. The savings in computation time is obvious.

Majority of space complexity for the *pairwise virtual gene selection* algorithm comes from stage 2 in the algorithm where pairwise discriminative scores are recorded. The space needed for that is $O(\frac{n^2}{\theta})$ using sparse array. Under typical situation if we choose $\theta \sim \Theta(n)$, space complexity of Algorithm 2 becomes $O(n)$, although with a large constant.

3 Experiments

In this section, we report extensive experimental results on three publicly available microarray datasets [1][8][15]. In each case, we study the gene selection problem in the context of two class sample classification.

3.1 Colon Cancer Dataset

Data Preparation This dataset was published by Alon [1] in 1999. It contains measurements of expression levels of 2000 genes over 62 samples, 40 samples were from colon cancer patients and the other 22 samples were from normal tissue. The minimum value in this dataset is 5.8163, thus no thresholding is done. We perform base 10 logarithmic transformation and then for each gene, subtract mean and divide by standard deviation. We will refer to this dataset as Alon dataset in the rest of the paper.

Experiments We performed three experiments on this dataset to evaluate performance of the four feature selection algorithms. The main purpose of each experiment is listed as follows:

1. Compare classification accuracy and the stability of classification accuracy between *single gene t-score* [3], *single gene S2N score* [1], *clustered pairwise t-score* [3] (their all pair method modified by limiting computation within gene clusters), *pairwise virtual gene*. We refer this experiment as *alon.1* in this paper.
2. Study how the choice of number of clusters in the *pairwise virtual gene* algorithm affects classification accuracy and the stability of classification accuracy. We refer this experiment as *alon.2* in this paper.
3. Study how the choice of initial cluster centers in the *pairwise virtual gene* algorithm affects gene selection performance. The *pairwise virtual gene* algorithm uses the k-means clustering algorithm to first divide genes into gene clusters. K-means algorithm is not stable in the sense that by supplying different initial cluster centers, different clustering results will be returned. We refer this experiment as *alon.3* in this paper.

For experiment *alon.1*, we use three classification algorithms to measure the performance of the feature selection algorithms. The classification algorithms we use are *knn* (k-nearest neighbor classifier, with $k = 3$), *svm* (support vector machine, with radial kernel) [5] and a linear discriminant method *dld* (diagonal linear discriminant analysis) [14]. For cross validation of classification accuracy, we use a 2-fold cross validation method, which is the same as leave-31-out method used in [3]. We run 2-fold cross validation 100 times to obtain an estimate of classification accuracy. Standard deviation of classification accuracy is also reported here. The number of genes to be selected is limited to 100, as it is reported in the literature[1] that even top 50 genes produce good classifiers.

For experiment *alon.2*, we use *knn* with $k = 3$ as classifier. We experimented with clustering genes into 8, 16, 32, 64, 128, 256 clusters in stage one of *pairwise virtual gene*

algorithm and then measure 2-fold classification accuracy as stated in the previous paragraph.

For experiment *alon.3*, we use knn with $k = 3$ as classifier. Same experiments are repeated for 20 times with randomly generated initial cluster centers for stage one of *pairwise virtual gene* algorithm. Performance of our feature selection methods is reported.

In all experiments, we measure performance of selecting from 2 to 100 genes, increasing by 2 at a time. We set $\alpha = 0, \beta = 1$ in all these experiments. As stated before, when comparing single gene-based gene selection algorithm with *pairwise virtual gene* algorithm, we treat each *pairwise virtual gene* as two genes. Thus the performance of classifiers built from top n genes are compared with the performance of classifiers built from top $\frac{n}{2}$ *pairwise virtual genes*.

Results Results of our experiment *alon.1* are summarized in Figures 2,3,4. In each figure, the left part plots classification accuracy against number of genes used to build classifier and the right part shows standard deviations of classification accuracy. By calculating the standard deviation, we can roughly estimate how close the mean classification accuracy is to the real classification accuracy. Each figure shows classification accuracy we archived using different classification methods.

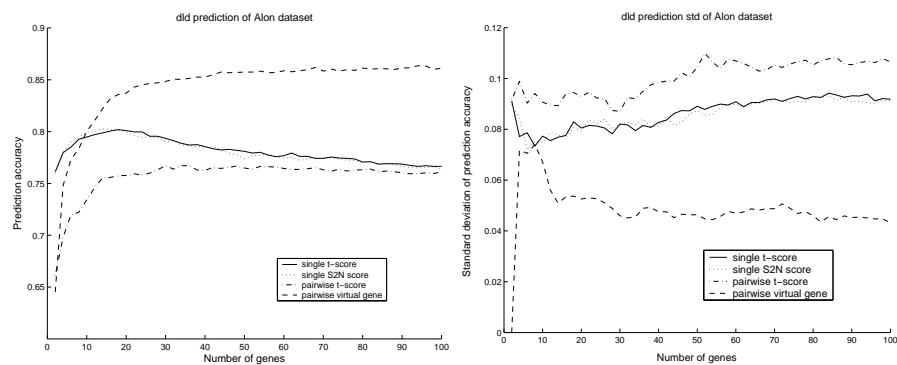


Fig. 2. Result of experiment *alon.1*. Prediction accuracy of four feature selection methods on Alon dataset using DLD classifier. Left figure shows prediction accuracy against the number of genes used to build DLD classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

From these experiments, we conclude that on Alon dataset, the *pairwise virtual gene* algorithm performs the best. When DLD and KNN classifiers are used, *pairwise virtual gene* algorithm is significantly better than other feature selection methods we tested. When SVM is used, all FSS methods produce comparable prediction accuracy with *pairwise virtual gene* algorithm enjoying small advantage over single gene based

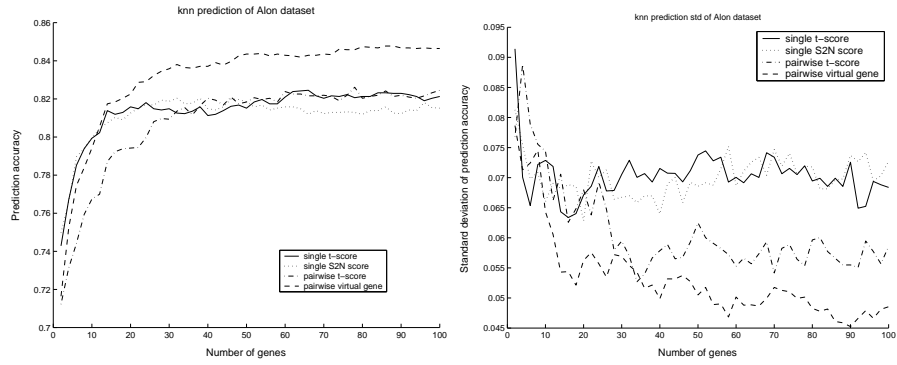


Fig. 3. Result of experiment alone.2. Prediction accuracy of four feature selection methods on Alon dataset using knn classifier ($k=3$). Left figure shows prediction accuracy against the number of genes used to build knn classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

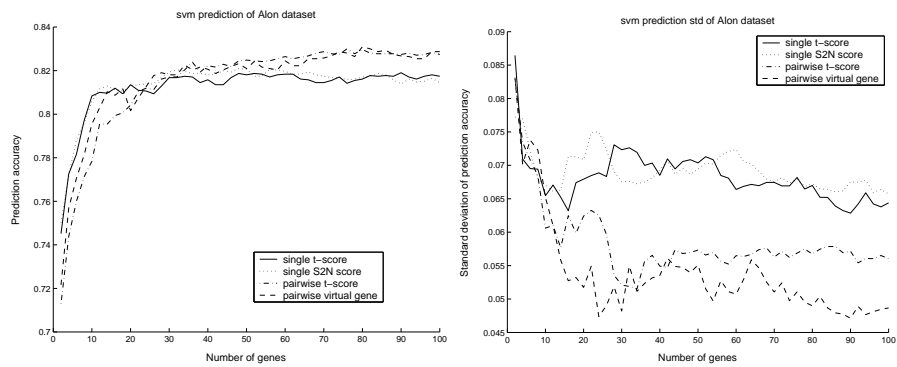


Fig. 4. Result of experiment alone.3. Prediction accuracy of four feature selection methods on Alon dataset using SVM classifier. In this experiment, we used a radial kernel for SVM. Left figure shows prediction accuracy against the number of genes used to build SVM classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

algorithms. The *pairwise virtual gene* algorithm is also most stable, in the sense that it has the smallest standard deviation of classification accuracy.

When testing using DLD classifier, the *pairwise virtual gene* algorithm results in 5%-10% increase in prediction accuracy over other FSS methods and almost 50% decrease in its standard deviation. The experiment with KNN classifier generates similar result with the *pairwise virtual gene* algorithm leading other FSS methods in classification accuracy by 2% and having the smallest variance.

Experiments with SVM generate more mixed results in which all four FSS methods having comparable classification accuracy. The *single gene t-score* and *single gene S2N* gene selection algorithms perform better than the *pairwise virtual gene* and *pairwise t-score* algorithms when the number of genes selected is less than 20. When more genes are selected, the *pairwise virtual gene* and *pairwise t-score* algorithms perform constantly better than the *single t-score* and *single gene S2N* algorithms. When the number of genes selected is more than 50, the *pairwise virtual gene* and *pairwise t-score* algorithms outperform the other two FSS algorithms by 1% in classification accuracy. The variations in classification accuracy still favors strongly towards pairwise methods with the *pairwise virtual gene* algorithm having the smallest variation.

For experiment along.2, we measure the performance of the *pairwise virtual gene* algorithm setting the number of cluster in stage 1 of the algorithm to be 8,16,32,64,128,256. The results are summarized in Figure 5. We see an overall trend of decline in performance as the number of clusters increases. The classification performance peaks when 8/16 clusters are used, indicating cluster numbers suitable for this dataset in that range. Compare two extremes, the 8-cluster version and the 256-cluster version, *pairwise virtual gene* algorithm performs about 2% better in classification accuracy using knn ($k = 3$) classifier when 8 clusters are used. This is somewhat we have expected since when using 256 clusters, compared to the 8-clusters version, the computed pairwise score is around $\frac{1}{32^2}$ or around 0.1%.

It is worth noting that we used a rather crude cluster analysis algorithm, the k-means algorithm. By computing only 0.1% (or omitting 99.9%) of all possible pairs in a 8-cluster version of the algorithm, we still get strong prediction accuracy, only losing about 2% of it. This indicates that correlations between genes within clusters generated by the k-means algorithm carry much more information on sample class distinction. We also expect to further improve *pairwise virtual gene* algorithm by using more sophisticated cluster analysis algorithms.

Since the k-means cluster algorithm is not stable, in the sense that initial cluster center assignments will affect clustering result, we perform experiment along.3 to determine how the *pairwise virtual gene* algorithm is affected by it. We run 2-fold cross validation 100 times. Each time, the *pairwise virtual gene* algorithm is run 20 times with randomly generated initial gene clusters to select 20 different sets of virtual genes. The performance of 3-nn classifier using each of the 20 virtual gene sets is measured. Figure 6 plots the mean value of the classification accuracy, with its standard deviation. From this experiment, we conclude although k-means cluster algorithm is not stable, it performs well enough to capture important gene pairs. Twenty different initial cluster centers result in twenty different *pairwise virtual gene* selection. However, the final classification accuracy measured with 3-nn (3 nearest neighbor) classifier using these

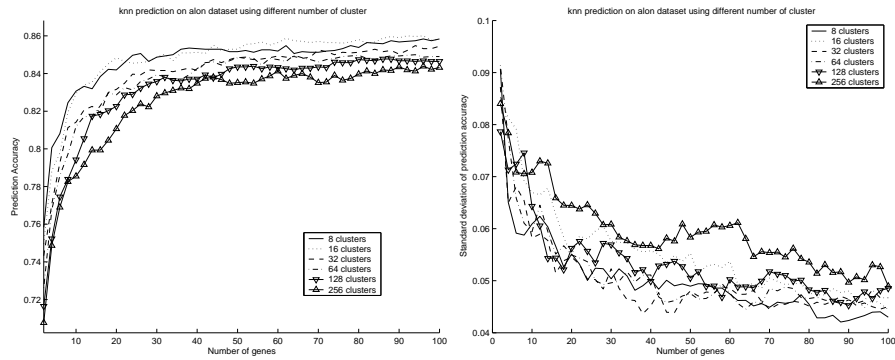


Fig. 5. Prediction accuracy and its standard deviation of knn ($k=3$) using different number of clusters in k-means algorithm (stage 1 of algorithm 2). Prediction accuracy degrade as the number of clusters increase. However, the within-cluster gene pairs (256-cluster version vs. 8-cluster version) retain much information as a reduction of 99.9% of pairs results only around 2% decrease in prediction accuracy.

twenty different *pairwise virtual gene* selections does not vary much (having standard deviation of 0.3% to 0.5%). This justifies the use of the unstable k-means algorithm in our algorithm.

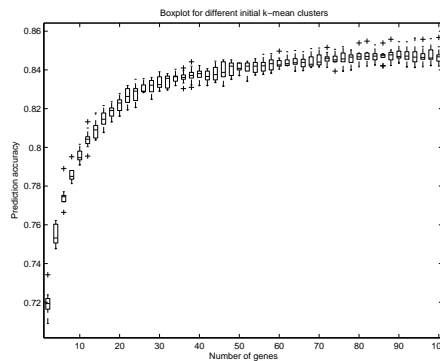


Fig. 6. The boxplot of mean 3-nn classification accuracy using *pairwise virtual gene* algorithm with 20 different initial clusters.

3.2 Leukemia Dataset

Data preparation This dataset was published by Golub etc. [8] in 1999. It consists of 72 samples, of which 47 samples were acute lymphoblastic leukemia (ALL) and rest

25 samples were acute myeloid leukemia (AML). 7129 genes were monitored in their study. This dataset contains a lot of negative intensity values. We use the following steps (similar to Dudoit etc.[7]) preprocessing the dataset before feed to our algorithm. First we threshold the data set with floor of 1 and ceiling of 16000. Then we filter out genes with $max/min \leq 5$ or $(max - min) \leq 500$, where max and min are the maximum and minimum expression values of a gene. After these two steps, the resulting 3927 genes are transformed using base 10 logarithmic and then the expression levels for each gene are normalized. We will refer to this dataset as Golub dataset in the rest of this paper.

Experiments We perform experiments to compare feature selection performance on Golub dataset. Two classifiers (KNN, DLD) are used. Classification accuracies of these classifiers using four feature selection algorithms (*single gene t-score*[3], *single gene S2N score*[8], *pairwise t-score*, *pairwise virtual gene*) are reported here. In all experiments, we measure performance of selecting from 2 to 100 genes, increasing by 2 at a time. We set $\alpha = 0, \beta = 0.8$ in all experiments.

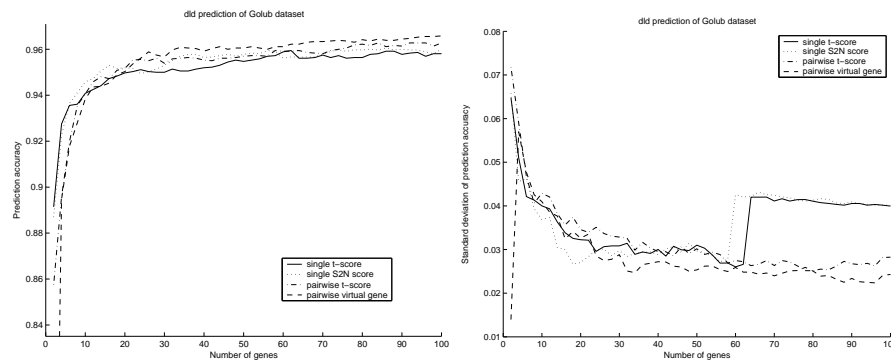


Fig. 7. Prediction accuracy of four feature selection methods on Golub dataset using DLD classifier. Left figure shows prediction accuracy against the number of genes used to build DLD classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

Results This dataset contains roughly four times the number of genes of Alon dataset. Straightforward computing of all gene pairs becomes intractable. Based on results obtained in the previous section on Alon dataset, we set the number of clusters to be 256. Results are shown in Figures 7,8. For DLD classifier, when the number of selected genes is larger than 20, *pairwise virtual gene* algorithm performs consistently better than single gene based algorithms, though not by a large margin. For knn classifier, *pairwise virtual gene* algorithm performs consistently better than all other methods we tested. Standard deviations of the classification accuracy declines as number of genes increase

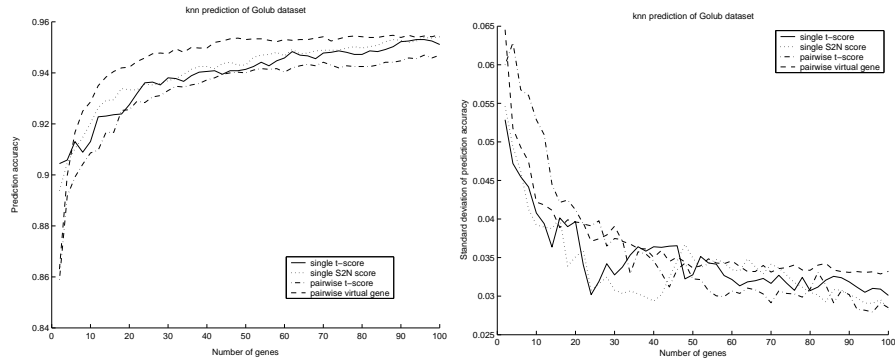


Fig. 8. Prediction accuracy of four feature selection methods on Golub dataset using knn classifier ($k=3$). Left figure shows prediction accuracy against the number of genes used to build knn classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

with one abnormal jump for single gene based methods using DLD classifier. All feature selection methods have similar variations in the classification accuracy. Overall, the *pairwise virtual gene* algorithm performs better than the single gene based algorithms on this dataset.

3.3 Multi-class Cancer Dataset

Ramaswamy etc. [15] reported study of oligonucleotide microarray gene expression involving 218 tumor samples spanning 14 common tumor types and 90 normal tissue samples. The expression levels of 16063 genes and expressed sequence tags were monitored in their experiments. The author separated the tumor samples into training set (144 samples) and testing set (54 samples). The rest 20 samples are poorly differentiated adenocarcinomas, which we did not include in our study. The training tumor set of 144 samples and 90 normal tissue samples are combined together for our study. We refer this data set as multi-class dataset in the rest of our paper.

Data preparation Like the Leukemia data set, multi-class dataset contains a lot of negative values. As a data preprocessing step, we apply a thresholding of 1 and filter out genes with $max/min \leq 5$ or $(max - min) \leq 500$. The resulting dataset has 11985 genes. Logarithmic transformation and normalization are then performed before data are fed to gene selection algorithms. It is worth nothing that in the original paper by Ramaswamy, etc.[15] all 16063 genes (or ESTs) were used for classification. For our study of feature selection, the application of $max/min \leq 5$ or $(max - min) \leq 500$ filter makes sense since we are only interested in several top ranked genes.

Experiments We measure performance of four feature selection algorithms using knn and dld classifiers. 2-fold cross validation is performed 100 times. Experiments are

in the same setting as for Alon and Golub datasets. In all experiments, we measure performance of selecting from 2 to 100 genes, increasing by 2 at a time. We set $\alpha = 0, \beta = 0.8$ in all experiments.

Results In this experiment, we set the number of clusters to be used to 400. Result of knn classifier shows single gene based algorithms performs better, but within 1% of accuracy compared to *pairwise virtual gene* algorithm. *Clustered pairwise t-score* algorithm performs as good as single gene based algorithms. As the number of genes selected increases, the differences in performance gradually converge.

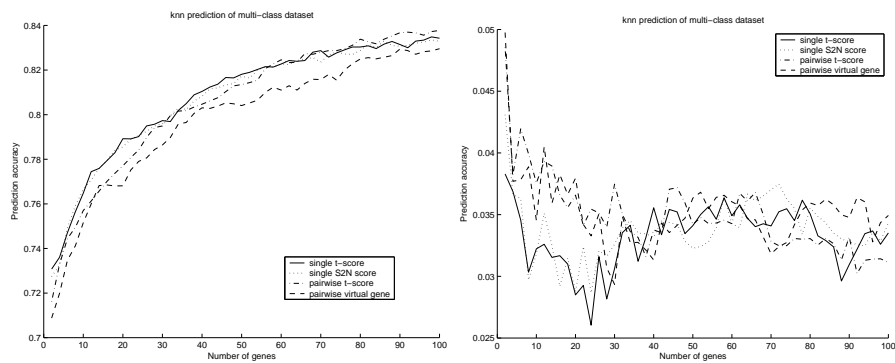


Fig. 9. Prediction accuracy of 4 feature selection methods on multi-class dataset using knn classifier ($k=3$). Left figure shows prediction accuracy against the number of genes used to build knn classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

4 Conclusion and Future Work

Gene selection is crucial both for building a good sample classifier and for selecting smaller gene set for further biological investigation. Feature extraction algorithms (PCA, SVD, etc.), single gene based discriminative scores (t-score, S2N, TNoM, information gain, etc.) and correlation based algorithms have been proposed for this purpose. In this paper, we proposed a totally different approach. Instead of trying to minimize correlations within the selected gene set, we examined whether such correlations are good predictors of sample class labels. *Virtual gene* is a linear combination of a set of real genes. Our experiments confirm our assumption that the correlations between genes are indeed good predictors of sample class labels, better in many cases than single gene based discriminative scores. There are biological explanation for this: genes interact with each other. The relative abundance of genes is a better predictor than the absolute values. Using gene clustering algorithms to limit gene pair selection seems promising.

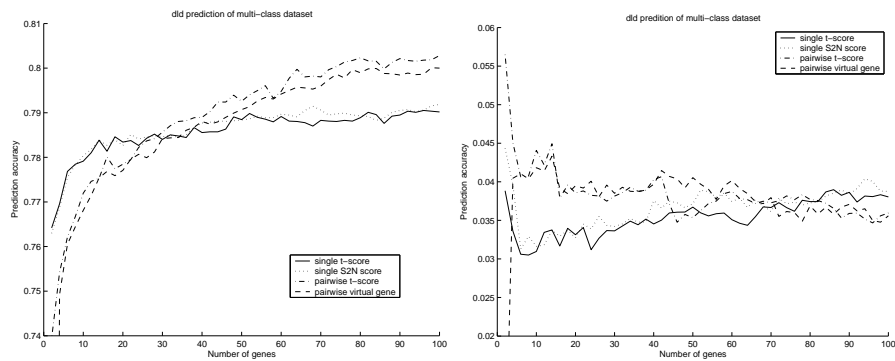


Fig. 10. Prediction accuracy of 4 feature selection methods on multi-class dataset using DLD classifier. Left figure shows prediction accuracy against the number of genes used to build knn classifier. Right figure shows the standard deviation of prediction accuracy against the number of genes.

Our experiments show that by calculating pairwise scores for only a very small portion (0.5%) of all possible gene pairs, decent classification performance can be achieved. This in turn shows most useful pairwise correlations are contained within gene clusters.

Our algorithm still has space for improvement. First but not least, we are interested in combining single gene based scores and virtual gene. In contrast to correlation based gene selection approaches, we can select top genes with high individual scores and top correlations between genes. We also want to examine larger virtual genes, virtual genes that combine more than two genes. Gene clustering is only a crude way of grouping co-regulated genes. We are currently working on using gene ontology as a way to group genes. Our algorithm is quite open, several other algorithms (e.g., cluster analysis and discriminative power of single gene) can be plugged into our algorithm without much modification. We leave this as future work as well.

References

1. U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissue probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. U.S.A.*, 96(12):6745–50, 1999.
2. A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. volume 7, pages 559–83, 2000.
3. T. Bø and I. Jonassen. New feature subset selection procedures for classification of expression profiles. *Genome Biology*, 3(4):research0017.1–0017.11, 2002.
4. G. V. Bobashev, S. Das, and A. Das. Experimental design for gene microarray experiments and differential expression analysis. *Methods of Microarray Data Analysis II*, pages 23–41, 2001.
5. C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines.
6. C. T. D. Jiang and A. Zhang. Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

7. S. Dudoit, J. Fridlyand, and T. P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97(457):77–87, 2002.
8. T. R. Golub et al. Molecular classifications of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–7, 1999.
9. T. Hastie, R. Tibshirani, M. Eisen, A. Alizadeh, R. Levy, L. Staudt, W. Chan, D. Botstein, and P. Brown. 'gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, 1(2), 2000.
10. j. Khan, J. Wei, M. Ringner, L. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. Antonescu, and C. Peterson. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–9, 2001.
11. J. Jaeger, R. Sengupta, and W. L. Ruzzo. Improved gene selection for classification of microarrays. In *Proc. PSB*, 2003.
12. A. K. Jian, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.
13. W. Li and I. Grosse. Gene selection criterion for discriminant microarray data analysis based on extreme value distributions. In *Proc. RECOMB*, 2003.
14. K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis*. Academic Press, 1979.
15. S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. Golub. Multiclass cancer diagnosis using tumor gene expression signatures. *PNAS*, 98(26):15149–15154, 2001.
16. V. G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *PNAS*, 98(9):5116–5121, April 2001.
17. Y. Wu and A. Zhang. Feature selection for classifying high-dimensional numerical data. In *IEEE Conference on Computer Vision and Pattern Recognition 2004*, volume 2, pages 251–258, 2004.
18. E. P. Xing, M. I. Jordan, and R. M. Karp. Feature selection for high-dimensional genomic microarray data. In *Proc. 18th International Conf. on Machine Learning*, pages 601–608. Morgan Kaufmann, San Francisco, CA, 2001.
19. L. Yu and H. Liu. Redundancy based feature selection for microarray data. In *Proc. of SIGKDD*, 2004.