Active Mining Discriminative Gene Sets  
(Invited)

Feng Chu and Lipo Wang

College of Information Engineering, Xiangtan University,  
Xiangtan, Hunan, China
School of Electrical and Electronic Engineering  
Nanyang Technological University, Singapore
elpwang@ntu.edu.sg

Abstract. Searching for good discriminative gene sets (DGSs) in microarray data is important for many problems, such as precise cancer diagnosis, correct treatment selection, and drug discovery. Small and good DGSs can help researchers eliminate “irrelevant” genes and focus on “critical” genes that may be used as biomarkers or that are related to the development of cancers. In addition, small DGSs will not impose demanding requirements to classifiers, e.g., high-speed CPUs, large memories, etc. Furthermore, if the DGSs are used as diagnostic measures in the future, small DGSs will simplify the test and therefore reduce the cost. Here, we propose an algorithm of searching for DGSs, which we call active mining discriminative gene sets (AM-DGS). The searching scheme of the AM-DGS is as follows: the gene with a large t-statistic is assigned as a seed, i.e., the first feature of the DGS. We classify the samples in a data set using a support vector machine (SVM). Next, we add the gene with the greatest power to correct the misclassified samples into the DGS, that is the gene with the largest t-statistic evaluated with only the mis-classified samples is added. We keep on adding genes into the DGS according to the SVM’s mis-classified data until no error appears or overfitting occurs. We tested the proposed method with the well-known leukemia data set. In this data set, our method obtained two 2-gene DGSs that achieved 94.1% testing accuracy and a 4-gene DGS that achieved 97.1% testing accuracy. This result showed that our method obtained better accuracy with much smaller DGSs compared to 3 widely used methods, i.e., $T$-statistics, $F$-statistics, and SVM-based recursive feature elimination (SVM-RFE).

1 Introduction

Accurate classification of homogenous cancers is a key problem for disease diagnosis, treatment selection, pathology research, and drug discovery. In recent years, gene expression profiles have been extensively applied to classifying cancers at the molecular level [5][13][14]. A typical gene expression data set can be described as a high dimensional $n \times m$ matrix $B$. In $B$, each column stands for a cancer sample (i.e., an observation) and each row stands for a gene. Here
usually ranges from several tens to over one hundred and \( n \) usually ranges from several thousands to tens of thousands. Since \( n \) is much larger than \( m \), it is of great importance to select a group genes (rather than use all of them) for classification because of the following two points. First, among all the genes, only a part of them have discriminating power. Furthermore, some genes even act as “noise” and undermine the classification accuracy. Second, some genes are highly correlated and their expression profiles behave very similarly in classification. Excluding some of such correlated genes will reduce redundancy in the discriminative gene sets (DGS).

Since mid-1990s, a number of gene selection approaches \[8,9,11,12,10,15\] have been proposed. Most of these methods can be regarded as filter schemes \[20\], which first rank genes according to their discriminative ability and then select a certain number (e.g., 20, 50, or 100) of top-ranked genes for classification. Although these top-ranked genes can lead to highly accurate classification results, they may still contain great redundancy. Some other methods use wrapper scheme \[20\]. In \[1\], a support vector machine based recursive feature elimination method (SVM-RFE) is proposed, which eliminates unimportant genes (i.e., the genes with little or no discriminating power) or redundant genes one by one from the initial gene set that includes all the genes. Since the SVM-RFE usually has to eliminate several hundreds or thousands genes to obtain a final DGS, it requires a large amount of computing time. In \[17\], a method called Markov blanket was used to reduce redundancy in DGSs. Since the Markov blanket mainly focus on reducing redundancy, it does not guarantee that the resulting DGS has very good discriminating power. In \[7\], Wang et al. proposed a method that uses unsupervised clustering to identify the redundancy in DGSs and then reduced the redundancy by “collapsing dense clusters”. They firstly rank all the genes and then select some top-ranked genes. After that, they cluster these “pre-selected” genes and pick out a representative gene for each cluster. The DGSs were formed using these representative genes. Although this method is able to reduce the redundancy of DGSs, the obtained DGSs are often not optimal because of the following reasons. (a) The cooperation among clusters and their representatives are not optimal; (b) A gene sometimes cannot represent the whole cluster, especially when the cluster contains more genes than other clusters. In \[24\], Liu et al. used entropy to reduce the redundancy of DGSs. However, the computation of entropy needs to know or estimate the very complicated probability density function of training samples, which prevents the entropy-based method becoming popular for this application.

Here we propose a simple yet very effective and efficient method of searching for DGSs that lead to high classification accuracy. Our method is a top-down forward wrapper search scheme, which is much more computationally efficient than the SVM-RFE scheme \[11\] and is able to greatly reduce the redundancy of DGSs by considering the cooperation among genes.

The rest of this paper is organized as follows. In Section 2 we introduce our SVM-based method of searching for DGSs, i.e., active mining discriminative gene sets (AM-DGS), and its related techniques. In Section 3 we apply our
SVM-based AM-DGS algorithm to the well-known benchmark gene expression data sets, i.e, the leukemia data set [5]. In Section 4, we discuss our results and conclude the paper.

2 Active Mining Discriminative Gene Sets

Recently, active learning has attracted great attention in the machine learning field because of its self-learning ability [2, 3, 22, 23]. An active learner, AL, has three components \( \{X, F, Q\} \). Here \( X \) is the input matrix. \( F \) is the mapping function from input space to output space that describes the objective (or function) of the AL. \( Q \) is a query function that is used to determine the sequence of unlabelled samples to be learned by the AL according to the current state of the AL, i.e., the AL has the ability to choose the “new things” that will “benefit” its learning. Compared to passive learners, which only contain \( X \) and \( F \) but no \( Q \), ALs are able to select data for themselves based on the learners’ present performance and therefore has the potential to obtain better learning results.

For almost all the active learning approaches proposed to date, the function \( Q \) is used to search for the unlabelled samples, i.e., observations, to be learned by the AL. In the following parts of this section, we will propose a learning scheme with a query function \( \tilde{Q} \) that is used to search for features (i.e., genes in this application) according to the current state of the learner (i.e., the SVM classifier in this application) and its objective. Hence we call our algorithm active mining as opposed to active learning. In addition, our proposed method is a forward searching scheme that is more straight-forward and efficient than backward searching schemes are.

2.1 T-Statistic

In the first step of our scheme, we rank all the features (genes) according to their \( t \)-statistics (TSs). The TS of gene \( i \) is defined as follows [16].

\[
TS_i = \left| \frac{\bar{x}_{c1} - \bar{x}_{c2}}{s_{pi} \sqrt{1/n_1 + 1/n_2}} \right|
\]

where

\[
\bar{x}_{c1} = \sum_{j \in C_1} \bar{x}_{ij} / n_1
\]

\[
\bar{x}_{c2} = \sum_{k \in C_2} \bar{x}_{ik} / n_2
\]

\[
s_{pi}^2 = \frac{\sum_{j \in C_1} (x_{ij} - \bar{x}_{c1})^2 + \sum_{k \in C_2} (x_{ik} - \bar{x}_{c2})^2}{n_1 + n_2 - 2}
\]

There are 2 classes, i.e., \( C_1 \) and \( C_2 \), which include \( n_1 \) and \( n_2 \) samples, respectively. \( x_{ij} \) and \( x_{ik} \) are the expression values of gene \( i \) in \( C_1 \) and \( C_2 \), respectively. \( \bar{x}_{c1} \) and \( \bar{x}_{c2} \) are the mean expression values of \( C_1 \) and \( C_2 \). \( s_{pi} \) is the pooled standard deviation of gene \( i \).
2.2 Seeds

After ranking all the genes with TSs, the gene with the largest TS is selected as the first feature in the discriminative gene set (DGS). We call this first feature the *seed*. The best seed that leads to the highest accuracy may not necessarily be the No.1 gene in the TS ranking result (the gene with the greatest TS). It can be the No.2 gene, the No.3 gene and so on. In our application, we use a number of top genes as seeds to search for the best DGS with the highest classification accuracy.

2.3 Support Vector Machines

We use support vector machines (SVMs) [18] [19] as our classifier, i.e., we input our DGS into an SVM to carry out training and classification.

A standard SVM classifier aims to solve the following problem. Given $l$ training vectors $\{x_i \in \mathbb{R}^n, i = 1, ..., l\}$ that belong to two classes, with desired output $y_i \in \{-1, 1\}$, find a decision boundary:

$$w^T \phi(x_i) + b = 0,$$

where $w$ is the weight vector and $b$ is the bias. $\phi(x_i)$ is the function that maps $x_i$ to a potentially much higher dimensional feature space. This decision boundary is determined by minimizing the cost function:

$$\psi = \frac{1}{2} ||w||^2 + C \sum_{i=1}^{l} \xi_i,$$

subject to:

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0.$$

where $\{\xi_i, i = 1, 2, ..., l\}$ are slack variables and $C$ is a constant that determines the tradeoff between the training error and the generalization capability of the SVM. This optimization problem has a quadratic programming (QP) dual problem:

$$\text{maximize: } Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j),$$

subject to:

$$\sum_{i=1}^{l} \alpha_i y_i = 0,$$

$$C \geq \alpha_i \geq 0,$$
where \( \{ \alpha_i, i = 1, 2, ..., l \} \) are Lagrange multipliers. For this problem, we use the sequential minimum optimization [4] as the QP-solver.

2.4 Correction Score

We define a ranking scheme, which we call correction score (CS), to measure a feature’s ability to separate the samples that are misclassified by the DGS obtained in the previous round of training. (Here we define the process of picking out a feature and adding it into a DGS as a *round* of training.) The CS of gene \( i \) is defined as:

\[
CS_i = \frac{S_{bi}}{S_{wi}}
\]  

(12)

where

\[
S_{bi} = \sum_{j \in C_1} (e_{ij} - \overline{x}_{c1})^2 + \sum_{k \in C_2} (e_{ik} - \overline{x}_{c2})^2
\]  

(13)

\[
S_{wi} = \sum_{j \in C_1} (e_{ij} - \overline{x}_{c2})^2 + \sum_{k \in C_2} (e_{ik} - \overline{x}_{c1})^2
\]  

(14)

where \( e_{ij} \) and \( e_{ik} \) are the expression values of *misclassified* samples in \( C_1 \) and \( C_2 \), respectively. \( \overline{x}_{c1} \) and \( \overline{x}_{c2} \) are defined in Eq.2 and Eq.3. \( S_{bi} \) is the sum of squares of the inter-class distances [21] (the distances between samples of different classes) among the misclassified samples. \( S_{wi} \) is the sum of squares of the intra-class distances (the distances of samples within the same class) among the misclassified samples.

2.5 Adding Features According to Misclassification

We input the feature with the largest CS into the SVM in the next round of learning. Our method of searching for the discriminating gene sets is analogous to an AL in the sense that our method has the ability to choose the feature (i.e., the gene) to be included in the next round of learning based on the present state of the learner (i.e., the SVM).

2.6 SVM-Based AM-DGS

The whole process to obtain a DGS is summarized as follows.

**Algorithm: SVM-based AM-DGS**

Inputs:

Training samples: \( X_{tr} = [x_{tr1}, x_{tr2}, ..., x_{trl}]^T \), validation samples: \( X_v \), testing samples \( X_{test} \)

Class labels for training, validation, and testing samples: \( Y_{tr} = [y_{tr1}, y_{tr2}, ..., y_{trl}]^T \), \( Y_v \), \( Y_{test} \)

The number of top-ranked genes to search for DGSs: M.

Initialize:
Initialize DGS to an empty matrix: \( \text{DGS} = [\ ] \).
Initialize the training error to 1: \( E_{tr} = 1 \).
Initialize the validation error to 0: \( E_v = 0 \).
Initialize the repeat counter to 0: \( \text{Rpt} = 0 \).
Choose a seed:
Calculate the \( TS \) for each feature in \( X_{tr} \).
\[
\text{for}(m = 1; \text{until } m < M; m++)
\]
\[
\left\{ \begin{array}{l}
\text{Select a feature with the m-th largest } TS \text{ as the seed (S).} \\
\text{S} \rightarrow \text{DGS.}
\end{array} \right.
\]
Repeat until: \( E_{tr} = 0 \) or \( E_v < E_{vpre} \) or \( \text{Rpt} > 2 \):
\[
\left\{ \begin{array}{l}
E_{trpre} = E_{tr} \\
E_{vpre} = E_v \\
\text{Train an SVM with DGS then obtain } E_{tr}.
\end{array} \right.
\]
Pick out the misclassified samples \( X_e = \{x_{e1}, x_{e2}, ..., x_{et}\}^T \).
Validate the SVM using \( X_v \) and obtain \( E_v \).
If \( E_{vpre} = E_v \), \( \text{Rpt} = \text{Rpt} + 1 \).
Calculate CS for each feature in \( X_e \).
Pick out the feature with the largest CS and put it into the DGS.
\[
\}
\]
Output DGS

3 Experimental Results

We tested our method in the well-known leukemia data set [5]. The leukemia data set [5] (http://www-genome.wi.mit.edu/cancer/) contains two types of leukemia samples, i.e., acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL). Golub et al. divided the data into 38 samples for training and the other 34 independent samples for testing. Among the 38 training samples, there are 27 ALL samples and 11 AML samples. Among the 34 testing samples, there are 20 ALL samples and 14 AML samples. The entire leukemia data set contains the expression values of 7129 genes. We normalized this data set by subtracting the mean and dividing the standard deviation across each sample.

We processed the leukemia data set with our SVM-based AM-DGS algorithm and showed the results in Table 1. Here we list the 8 DGSs whose seeds are the top 8 genes according to their TSs. For each DGS, the first gene (i.e., the first line in the DGS) is its seed. The second, third (and so on) genes are the genes included in the DGS in the corresponding round, respectively.

From these results, we found that our SVM-based AM-DGS is very effective and efficient in finding good DGSs. Let us use DGS 1 to illustrate this. Since
Table 1. Training and testing accuracies for various DGSs obtained by our SVM-based AM-DGS algorithm to the leukemia data set

<table>
<thead>
<tr>
<th>Set No.</th>
<th>Gene Sets</th>
<th>Training Accuracy (%)</th>
<th>Testing Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>U50136_rna1_at</td>
<td>92.11</td>
<td>79.41</td>
</tr>
<tr>
<td></td>
<td>X17042_at</td>
<td>100</td>
<td>82.35</td>
</tr>
<tr>
<td>2</td>
<td>X95735_at</td>
<td>97.37</td>
<td>94.12</td>
</tr>
<tr>
<td></td>
<td>M23197_at</td>
<td>100</td>
<td>94.12</td>
</tr>
<tr>
<td>3</td>
<td>M55150_at</td>
<td>97.37</td>
<td>82.35</td>
</tr>
<tr>
<td></td>
<td>M84526_at</td>
<td>92.11</td>
<td>82.35</td>
</tr>
<tr>
<td></td>
<td>M23197_at</td>
<td>97.37</td>
<td>91.18</td>
</tr>
<tr>
<td>4</td>
<td>M16038_at</td>
<td>92.11</td>
<td>79.41</td>
</tr>
<tr>
<td></td>
<td>U22376_cds2_s_at</td>
<td>94.74</td>
<td>82.35</td>
</tr>
<tr>
<td>5</td>
<td>Y12670_at</td>
<td>94.74</td>
<td>64.71</td>
</tr>
<tr>
<td></td>
<td>U22376_cds2_s_at</td>
<td>100</td>
<td>82.35</td>
</tr>
<tr>
<td>6</td>
<td>M23197_at</td>
<td>92.11</td>
<td>85.29</td>
</tr>
<tr>
<td></td>
<td>U22376_cds2_s_at</td>
<td>97.37</td>
<td>88.24</td>
</tr>
<tr>
<td></td>
<td>M63138_at</td>
<td>97.37</td>
<td>94.12</td>
</tr>
<tr>
<td>7</td>
<td>D49950_at</td>
<td>97.37</td>
<td>94.12</td>
</tr>
<tr>
<td></td>
<td>U22376_cds2_s_at</td>
<td>86.84</td>
<td>67.65</td>
</tr>
<tr>
<td></td>
<td>X04085_rna1_at</td>
<td>94.74</td>
<td>76.47</td>
</tr>
<tr>
<td></td>
<td>U50136_rna1_at</td>
<td>100</td>
<td>82.35</td>
</tr>
<tr>
<td>8</td>
<td>X17042_at</td>
<td>89.47</td>
<td>79.41</td>
</tr>
<tr>
<td></td>
<td>U22376_cds2_s_at</td>
<td>89.47</td>
<td>85.29</td>
</tr>
<tr>
<td></td>
<td>M86406_at</td>
<td>94.74</td>
<td>64.71</td>
</tr>
<tr>
<td></td>
<td>X95735_at</td>
<td>100</td>
<td>97.06</td>
</tr>
</tbody>
</table>

DGS 1 contained only 2 genes, we plotted the gene expression values of the two genes in Fig.1. In the first round of training, only the seed, i.e., gene U50136_rna1_at, was input to the SVM. Because gene U50136_rna1_at has a high TS, the SVM misclassified only three samples that were indicated with arrows. In the second round training, the algorithm selected the gene that had the best capability to separate the three misclassified samples, i.e., gene X17042_at. We found in Fig.1(a) that gene X17042_at “dragged” the misclassified samples away from the classes which these 3 samples were mistakenly assigned to in the previous round of training. Therefore, with the help of the second gene X17042_at, DGS 1 increased its training accuracy from 92.11% to 100%; the 38 training samples were perfectly separated by DGS 1.

The best testing accuracy was obtained by DGS 8, which included 4 genes. The SVM obtained 100% training accuracy and 97.1% testing accuracy (i.e., 1 errors in the 34 testing samples) using DGS 8. In this data set, we used the 8 genes with the largest TSs as the seeds (M=8 in our algorithm summarized in the previous section). If more seeds were used, more DGSs could be found.
Fig. 1. Gene expression values for the two genes in DGS 1 in the leukemia data set. (a) a plot includes only the training samples; (b) a plot includes all the training and testing samples.

4 Discussion

The results of leukemia data set visually indicate the effectiveness of our SVM-based AM-DGS algorithm. Except the seeds, all the genes in a DGS are selected according to their capability to correct misclassified samples. Therefore, the SVM-based AM-DGS can optimize the cooperation among genes and hence leads to good accuracy and smaller DGSs. Compared with the filter approaches, e.g., TS and FS, the SVM-based AM-DGS can greatly reduce the redundancy in a DGS.
In conclusion, the SVM-AMDGS proposed here is effective and computationally efficient in searching for good DGSs, the simulation using the leukemia data set shows that our algorithm leads to highly accurate classifications with the smallest gene sets found in the literature.

References