Feature Selection for Microarray Data Using Least Squares SVM and Particle Swarm Optimization

E. K. Tang[1], P. N. Suganthan[*1] and X. Yao[2]
[1] School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
Email: tangke@pmail.ntu.edu.sg; epnsugan@ntu.edu.sg
[2] School of Computer Science
University of Birmingham
Birmingham, B15 2TT,
United Kingdom
Email: X.Yao@cs.bham.ac.uk

Abstract-**Feature selection is an important preprocessing technique for many pattern recognition problems. When the number of features is very large while the number of samples is relatively small as in the micro-array data analysis, feature selection is even more important. This paper proposes a novel feature selection method to perform gene selection from DNA microarray data. The method originates from the least squares support vector machine (LSSVM). The particle swarm optimization (PSO) algorithm is also employed to perform optimization. Experimental results clearly demonstrate good and stable performance of the proposed method.**

I. Introduction

Commonly addressed in the pattern classification field, feature selection methods eliminate irrelevant or redundant features and select the most informative subset of features to enhance the generalization performance. For three reasons, feature selection is even more important for the classification of gene expression microarray data, where feature selection is also referred to as gene selection. Firstly, many of the genes may be irrelevant or insignificant to a specific classification problem. Previous studies have shown that a small subset of genes may be sufficient for a particular biological classification problem [1]. Secondly, feature selection can reduce the data volume while retaining most of the discriminative information, which will reduce the computational and storage requirements. Finally, feature selection also helps to gain deeper understanding of the functions of particular genes. This is very important for designing microarray experiments for clinical diagnosis and prognosis purposes.

A typical feature selection method consists of two components - an evaluation criterion and a searching scheme. The evaluation criterion can be either the classification accuracy or some other quantities such as the Fisher's ratio, Mahalanobis class seperability [2] or norm of the classification hyperplane of a support vector machine (SVM) [3]. The searching scheme searches the space of all possible feature subsets. Various searching schemes have also been well discussed in pattern recognition literatures, such as sequential forward selection (SFS), sequential floating forward selection (SFFS), branch and bound and so on [2]. Recently, binary evolutionary algorithms (EA) have also been employed as the searching scheme [4]. The feature subsets discovered by the search algorithm are evaluated with respect to the evaluation criterion. Finally, the feature subset yielding the optimal value with respect to the evaluation criterion is chosen.

As many evaluation criteria and searching schemes are available, it is possible to develop many feature selection methods by just combining different evaluation criteria and searching schemes. Since, many of these combinations actually perform similarly, it is sufficient to perform comparisons with the most commonly used combinations instead of all possible combinations. In [1], Golub *et al.* employed the weighting factor, which is a minor variant of Fisher's ratio, as the evaluation criterion. The features are evaluated individually rather than as a subset. This scheme is also referred as feature ranking [3]. In [3], Guyon *et al.* introduced a top-down recursive feature elimination (RFE) scheme, in which features were eliminated successively according to their influence on a support vector machine (SVM) based evaluation criterion. Rakotomamonjy then extended the work by introducing more SVM-based criteria [5]. Recently, Zhou and Mao proposed a new criterion named LS bound measure to evaluate a feature subset [6]. The LS bound was combined with SFS as well as SFFS schemes to obtain competitive results.

In the context of gene expression data analysis, different feature selection methods have their own advantages and disadvantages. For example, if the data contain D features (genes) and d of them are to be selected, the correlation coefficient methods only require D evaluations, while RFE scheme requires $(2D\text{-}d\text{-}1)(D\text{-}d)/2$ evaluations, SFS scheme requires $(2D\text{-}d+1)d/2$ evaluations and SFFS scheme requires even more evaluations. If we do not consider the differences among different evaluation criteria but only the searching schemes, the correlation coefficient-based individual feature ranking methods are usually the most efficient ones, while SFFS is highly time consuming. Similarly, although EAs are powerful optimization techniques and feature selection can be

0-7803-9387-2/05/$20.00 ©2005 IEEE

viewed as a binary optimization problem, binary EAs (they are more practical than continuous EAs for binary optimization problems) are even more time consuming than SFFS scheme. Note that the comparison is under the assumption that computational costs of the evaluation criteria are generally the same for all the searching schemes. In practice, criteria combined with RFE, SFS and SFFS, such as SVM-based criterion and LS bound measure are more complex than Fisher's criterion and factor weighting measure that are usually employed in individual feature ranking methods, hence correlation coefficient methods are actually even more efficient than we have acknowledged. The efficiency is due to the fact that they select a subset of good features rather than a good feature subset. However, to achieve high classification accuracy, a good feature subset is usually more important than a subset of good features. Therefore, the correlation coefficient-based methods generally select a sub-optimal feature subset for classification thereby resulting in lower accuracy. According to different requirements, one can trade off between accuracy and efficiency by selecting a suitable feature selection method.

For the gene selection problem in microarray data analysis, D is usually very large and d is relatively very small. By viewing feature selection methods from different aspects, a good gene selection method should have the following characteristics: The computational cost for a single evaluation is low, the required number of evaluations is small and the evaluation criterion can guarantee high classification accuracy. Motivated by these considerations, we propose in this paper a novel gene selection method. The method employs an efficient calculation of the leave-one-out (LOO) error of an LSSVM as the evaluation criterion. Since the LOO error is a good estimator of the generalization performance, one can expect to achieve good generalization performance by selecting feature subsets with respect to the LOO error. After proposing the efficient calculation of LOO error, we also present a scheme to efficiently optimize the feature subset using a continuous EA. Although we preliminarily employ the standard particle swarm optimization (PSO) algorithm, other EAs can also be employed.

The remainder of this paper is organized as follows: In Section 2, we review the LSSVM classifier and the LS bound gene selection method. In section 3, we describe the proposed gene selection method. In Section 4, the proposed method is compared and evaluated against some recent gene selection methods on several microarray datasets. Discussions and conclusions are presented in section 5.

II. LSSVM AND LS BOUND FEATURE SELECTION

A. LSSVM Classifier

Proposed by Suykens et al. [7], least squares SVM is a modification of the standard SVM. In a classification problem, we are given a training dataset of n training sample pairs: $\{\mathbf{x}_i, y_i\}$, $i = 1, \ldots, n$, where $\mathbf{x}_i \in \Re^d$ is the ith training sample, and y_i is the corresponding class label, which is either +1 or -1. The LSSVM classifier employs a set of mapping functions Φ to map the input data into the reproducing kernel Hilbert space, where the mapping function is implicitly defined by the kernel function: $k(\mathbf{x}_i, \mathbf{x}_j)=\Phi(\mathbf{x}_i)\cdot \Phi(\mathbf{x}_j)$. The general framework of the LSSVM is formulated as:

$$\min_{\mathbf{w},\mathbf{e}} J(\mathbf{w},\mathbf{e}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{2}\sum_{i=1}^{n}\gamma e_i^2 \quad (1)$$

$$\text{s.t.} \quad y_i\left[\mathbf{w}\cdot\Phi(\mathbf{x}_i)+b\right]=1-e_i \quad (2)$$

where, e_i denotes regression error for sample $\mathbf{x}_i$, $\mathbf{e} = [e_1, e_2, \ldots, e_n]^T$, and γ is a given positive constant. The role of γ, just as that of the regularization constant C in the classical SVM, is to adjust the compromise between generalization and training accuracy. By introducing Lagrangian multipliers, the solution can be obtained by solving the following linear system:

$$\begin{bmatrix} 0 & \mathbf{Y}^T \\ \mathbf{Y} & \mathbf{K}+\mathbf{\Lambda}_\gamma \end{bmatrix}\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{\mathbf{1}} \end{bmatrix} \quad (3)$$

where $\mathbf{Y}= [y_1, y_2, \ldots, y_n]^T$, $\vec{\mathbf{1}} = [1,\ldots,1]^T$, $\mathbf{K} = \{y_i y_j k(x_i, x_j)\}$ and $\mathbf{\Lambda}_\gamma = diag\{1/\gamma, 1/\gamma, \ldots, 1/\gamma\}$. b and $\boldsymbol{\alpha}$ are the solutions of the LSSVM, b is a scalar and $\boldsymbol{\alpha}= [\alpha_1, \alpha_2, \ldots, \alpha_n]^T$. Similar to the standard SVM, the discriminant function achieved by solving an LSSVM takes the form:

$$\mathbf{w} = \sum_{i=1}^{n}\alpha_i y_i \Phi(\mathbf{x}_i) \quad (4)$$

$$f(\mathbf{x}) = sign\left[\sum_{i=1}^{n}\alpha_i y_i k(\mathbf{x},\mathbf{x}_i)+b\right] \quad (5)$$

The main difference between the standard SVM and the LSSVM is that for the standard SVM the equality constraints in (2) are replaced by inequality constraints:

$$y_i\left[\mathbf{w}\cdot\Phi(\mathbf{x}_i)+b\right]\geq 1-e_i \quad (6)$$

By employing equality constraints instead of inequality constraints, LSSVM can be formulated as a linear system, which requires much less computation than a quadratic programming problem used in the standard SVM. Empirical results have shown that LSSVM can generally achieve comparable classification accuracy as the standard SVM.

B. LS Bound Feature Selection

Based on the standard SVM, Guyon et al. proposed the SVM-RFE feature selection method [3], which has become very popular for gene selection. In the statistical learning theory, it is known that leave-one-out error (LOOE) gives almost an unbiased estimation of the generalization performance of a classifier. Hence, for a feature selection problem, one can employ an LSSVM and calculate its LOOE to evaluate a feature or a feature subset. The feature or feature subset yielding the smallest LOOE will be selected. However, traditionally calculating the LOOE requires repeating the whole classification procedure for n times, where n is the number of samples. This is too time consuming and constrains the application of LOOE. To overcome this problem, several approaches have been proposed to efficiently estimate the LOOE for the standard SVM [8]. Since the LSSVM can be viewed as a modification of the standard SVM, recently Zhou and Mao proposed a new evaluation criterion based on the LOOE of an LSSVM [6], named LS bound measure, to solve gene selection problems. Given n samples represented by a set of features, the LS bound measure is defined as:

$$\mathbf{M} = \sum_{i=1}^{n} \left(\alpha_i \left[\left(D_{min}^{i} \right)^2 + 2/\gamma \right] - 1 \right)_+ \tag{7}$$

where $(x)_+ = \max(0,x)$, and D^i_{min} is the distance between $\mathbf{x}_i$ and its nearest neighbor in the reproducing kernel Hilbert space. This LS bound measure is proposed based on an upper bound of the LOOE of the LSSVM, which is

$$LOOE \le \frac{1}{2n} \left(n - \sum_{i=1}^{n} sign\left(\alpha_i \left[\left(D_{min}^{i} \right)^2 + 2/\gamma \right] - 1 \right) \right) \tag{8}$$

Like many other evaluation criteria, LS bound measure can be combined with any one of the searching schemes such as SFS, SFFS, RFE and so on. The gene selection algorithm combining the LS bound with SFS is illustrated below:

```
(1) Initialize $S$ to an empty set
      /*$S$ is the set of selected genes*/
(2) Initialize $C$ to the full gene set;
      /*$C$ is the set of candidate genes to be selected from*/
(3) For $i$ to $d$
      /*$d$ genes are to be selected*/
      $p$=number of genes in $C$;
      for $j$=1 to $p$
         Take gene $j$ from set $C$ and temporarily put into set $S$;
         Calculate the measure $M$ using all genes in set $S$;
      End
      Select the gene with the minimal $M$;
      Put the selected gene into set $S$;
   End
```

Since there exists a similar criterion for the standard SVM, one common question is whether the SVM-based criterion or the LSSVM-based criterion is better. As we mentioned before, computational cost is much lower for the LSSVM than for the standard SVM. Hence, from the perspective of efficiency, LSSVM is better. Further, experiments on several microarray datasets showed that combined with SFS, LS bound measure performs only slightly worse than the SVM-RFE based on accuracy. Compared with the great improvement in efficiency, LSSVM seems useful. However, (8) is only an upper bound of the LOO error, hence (7) is valid only when the bound is tight. If the bound is loose, (7) will give a biased estimation for the generalization performance and become an inaccurate measure. As whether the bound is tight or not is likely to be an uncertain data dependent issue, LS bound should be applied with caution. Our motivation is to reduce this uncertainty of LS bound measure.

III. LSSVM BASED EVOLUTIONARY FEATURE SELECTION

We begin with an improved evaluation criterion for the LSSVM. As discussed above, although several efficient measures have been proposed based on either standard SVM or LSSVM, they are naturally estimations of the LOO error rather than an exact value. This nature will result in possible biased evaluation of the features, and degrade the accuracy of the feature selection method. Different from these measures, our leave-one-out calculation (LOOC) measure, originates from an exact calculation of the LOO error of an LSSVM. This exact calculation of LOO error is presented as the Lemma below:

Lemma 1: Given a training dataset of n training data pairs $\{\mathbf{x}_i, y_i\}$ ($i = 1, \ldots, n$), the LOO error of an LSSVM on the dataset can be calculated by

$$LOOE = \frac{1}{2n} \left(n - \sum_{i=1}^{n} sign\left(1 - \alpha_i / \left(\mathbf{H}^{-1} \right)_{ii} \right) \right) \tag{9}$$

where $\mathbf{H} = \begin{bmatrix} \mathbf{\Omega} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}$, $\Omega = \{k(x_i, x_j)\} + \mathbf{\Lambda}_\gamma$ and $\left(\mathbf{H}^{-1}\right)_{ii}$ is the ith diagonal element of the inverse matrix of $\mathbf{H}$. α_i is defined in eqn (3). The proof of Lemma 1 is presented in Appendix A.

By comparing (8) and (9), one can find the computational cost is approximately the same for the upper bound in (8) and the exact calculation in (9) since both of them only require a single training procedure on the whole dataset. Therefore (9) is obviously an improvement over (8) since the consideration of whether the bound is tight or not is no longer necessary if exact calculation can be implemented with the same computational cost. Further, we can also observe from (9) that the LOOE is actually determined by the term $1-\alpha_i/(\mathbf{H}^{-1})_{ii}$. Large positive value of this term means that sample $\mathbf{x}_i$ can be easily classified to the correct class in the LOO procedure and visa versa. Based on this observation, our LOOC measure is defined as:

$$LOOC = \sum_{i=1}^{n} \frac{1}{1 + exp\left(1 - \alpha_i / \left(\mathbf{H}^{-1} \right)_{ii} \right)} \tag{10}$$

Different from (7), we employed the logistic function in our measure. The reason is that by employing the logistic function, the LOOC ranges between (0,1), which can be viewed as a probability that represent the generalization performance of a classifier and can be useful for possible post-processing procedure. Logistic function is also commonly used to transfer output of an SVM-type classifier into a specific region. For more details on this issue, please refer to [9].

Now we have presented the evaluation criterion, a searching scheme is also required for a gene selection method. A direct application of our LOOC measure is to combine it with traditional schemes, just like the LS bound measure. However, we have discussed that those schemes will be very time consuming when the number of features is large, which is true for microarray data. An alternative is to introduce a vector $\mathbf{v}$ of scaling factors into the kernel matrix, and modify $k(\mathbf{x}_i, \mathbf{x}_j)$ as $k(\mathbf{v}^T\mathbf{x}_i, \mathbf{v}^T\mathbf{x}_j)$. Introducing scaling factors is not a totally new idea in SVM based feature selection methods. In [5], this scheme is employed for an SVM based feature selection method. The method works similarly to SVM-RFE: SVM is optimized iteratively with respect to the scaling factors, in each iteration, the feature corresponding to the scaling factor that has the smallest absolute value is removed. Some other related works are also presented in [5]. However, in these works, introducing scaling factor is simply for defining a new evaluation criteria, the searching scheme of traditional SVM-RFE method is still employed. Furthermore, the dimensionality of **v** is the number of candidate features. Hence, optimization of either SVM or LSSVM with respect to **v** will be formulated as a very high dimensional optimization problem for microarray data, which may be very difficult to solve. Therefore, in contrast to the previous works, in this work we consider employing this idea to reduce the computational cost. To settle the high dimensionality problem, a principle component analysis (PCA) is applied to the original data first. In pattern recognition field, PCA is a popular technique for dimensionality reduction. It transfers high dimensional data to a lower dimensional space. Because the number of samples in microarray data is usually very small when compared to the number of genes, the PCA procedure reduces the number of features significantly, which typically equals the number of samples. After that, scaling factors are assigned to features of the transformed data and optimization can now be carried out much more efficiently. The basic particle swarm optimization (PSO) algorithm [11] is employed in our method to optimize these scaling factors. Finally, the scaling factors of the original genes can be well approximated by simple linear algebra operations.

By computing the scaling factors, we obtain a ranking score for each gene by just taking the absolute value of the scaling factors. The larger the score, the more important the corresponding gene will be for constructing the kernel matrix, which determines the solution of LSSVM. Therefore, selection can be easily carried out based on these scores. In experimental study, we choose the features with the largest ranking scores. The gene selection method, named LSSVM based evolutionary feature selection (LSEFS) is summarized as follows:

Initialization: Let **X** and **PX** be original data and PCA transformed data. Denoting **T**, **v** and **pv** as the transformation matrix of PCA procedure, scaling vector for the original data and transformed data respectively.

(1) PCA procedure
  PX=**TX**;
(2) Introduce pv, such that $k(\mathbf{x}_i, \mathbf{x}_j)$ becomes $k(\mathbf{pv}^T(\mathbf{Tx}_i), \mathbf{pv}^T(\mathbf{Tx}_j))$. Optimize (10) with respect to **pv** using a PSO algorithm.
(3) Since $k(\mathbf{pv}^T(\mathbf{T}\,\mathbf{x}_i), \mathbf{pv}^T(\mathbf{T}\,\mathbf{x}_j))$ is the same as $k((\mathbf{pv}^T\mathbf{T})\mathbf{x}_i, (\mathbf{pv}^T\mathbf{T})\mathbf{x}_j)$, **v** can now be computed as
  $\mathbf{v}= (\mathbf{pv}^T\mathbf{T})^T=\mathbf{T}^T\mathbf{pv}$, and the ranking score vector is abs(**v**)
(4) Features corresponding to the *d* largest elements of the ranking score vector are selected.

IV. EXPERIMENTAL RESULTS

In this paper, we compare our method with three other gene selection methods, namely LS bound combined with SFS scheme, Mahalanobis class separability measure combined with SFFS scheme (MAHSFFS) and the SVM-RFE method. We apply the gene selection algorithms to three microarray datasets: ALL-AML Leukemia (Leukemia), Hepatocellular Carcinoma (Carcinoma) and High-grade Golima (Golima). Table I summarizes some basic information of these datasets. They can be obtained from [10]. All the genes are standardized to zero mean and unit standard deviation. As the dimensionality of these microarray data is huge, and many of the genes are irrelevant to the classification task, we performed a pre-selection procedure to reduce the searching space and computational time. For each dataset, we selected top 1000 genes based on Fisher's ratio. Zhou and Mao also employed the same pre-selection scheme [6]. All the simulations and experiments in this paper are based on the pre-selected genes.

TABLE I
BASIC INFORMATION OF THE 3 MICROARRAY DATASETS

Dataset	Number of Samples	Number of genes
Leukemia	72	7129
Carcinoma	60	7129
Golima	50	12625

To assess performance of different feature selection methods, in some previous works researchers randomly split the original data into a training set and a testing set. The gene selection was then performed on the training set while the selected features were assessed on the testing set. However, this approach is not reliable due to the small sample size of microarray data. In our experiment, we employ the B.632+ bootstrap estimator to assess the performance of gene selection methods [12]. The balanced bootstrap samples are generated with K=200. In particular, the training sets are generated by re-sampling with replacement from the original dataset. Samples that are not contained in a training set go to the corresponding testing set. Each sample in the original dataset is made to appear exactly K times in the balanced bootstrap training sets.

In our experiments, for all the four gene selection methods, parameters of the LSSVM are tuned by 5-fold cross-validation on the training set. The performances of the methods are evaluated on gene subsets with number of genes ranging from 1 to 50. Two parameters need to be defined in advance to employ PSO in our LSEFS algorithm. They are the number of particles, and the number of generations. Since D and d are 1000 and 50 respectively, the LS bound with SFS scheme requires 48775 evaluations to select 50 features. Number of evaluations of MAHSFFS partially depends on the dataset, but it will definitely be larger than the SFS scheme, while the computational cost for a single evaluation using Mahalanobis class separability is comparable to LS bound and our measure. SVM-RFE requires only 1950 evaluations, but the computational cost of a single evaluation is much larger for SVM-RFE than for the other three methods. Based on these considerations, we set the number of particles and the number of generations at 50 and 200 respectively. Hence, 10000 evaluations are required for our LSEFS algorithm to select 50 features. Under this situation, our method requires the least overall computational cost among the four methods.

The results are presented in Figures 1 to 3. One can observe that our LSEFS algorithm performs well on all the three datasets. For Leukemia dataset, LSEFS performs almost the same as the LS bound SFS method, but obviously better than

SVM-RFE and MAHSFFS. For Carcinoma and Golima datasets, our method performs consistently better than all the other three methods. Since the overall computational cost of our method is no more than the other three methods, our LSEFS is competitive as a gene selection algorithm.

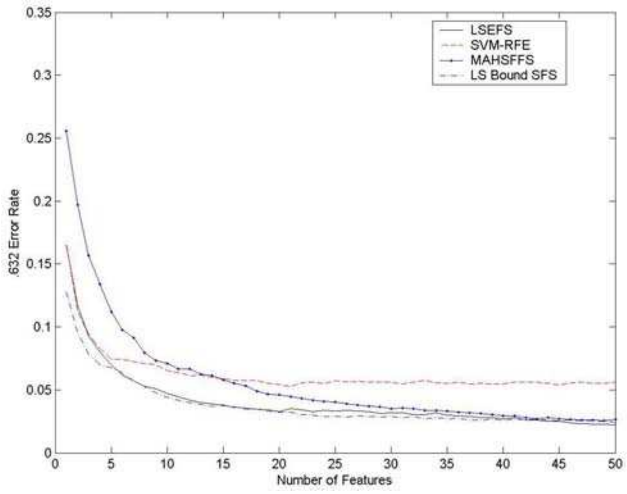

Fig. 1. .632 Error rates for the Leukemia dataset.

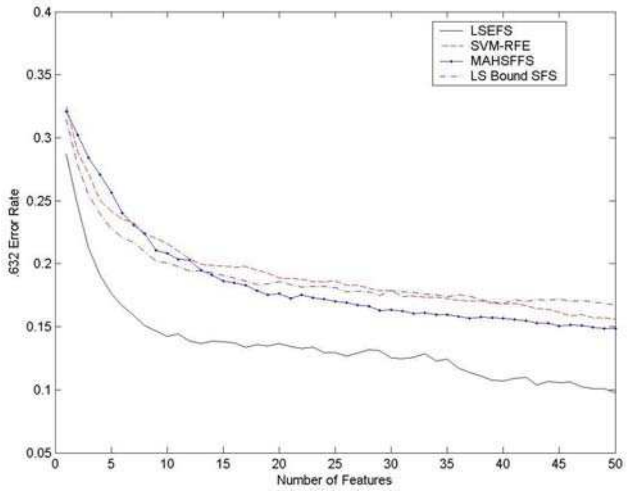

Fig. 2. .632 Error rate for the Carcinoma dataset.

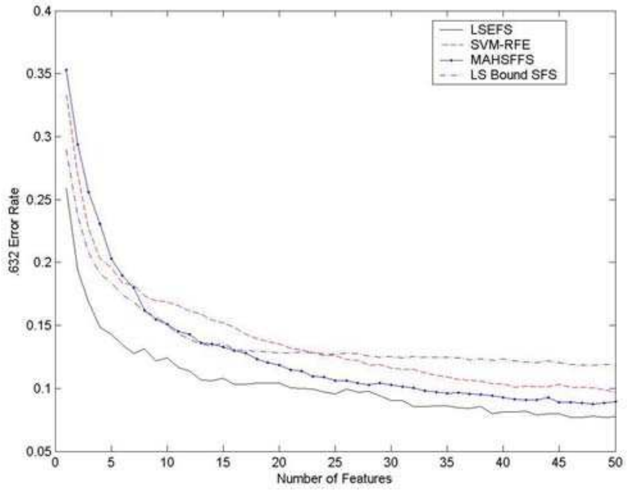

Fig. 3. .632 Error rate for the Golima dataset.

V. Discussion and Conclusion

Although LSEFS has shown good performance, there may be opportunities for further improvements. For example, by employing PCA to reduce dimensionality of the optimization problem involved, we manage to solve the problem more efficiently by PSO. However, PSO is not the only optimization method that is applicable. In practice, selection of optimization technique should be based on some underlying properties of the problem itself. For example, if the problem is unimodal and easy to solve, a simple gradient descent method, which is more efficient than EAs, is sufficient to achieve the solution. On the other hand, if the problem is multimodal and complex, a simple EA such as PSO may not work well and a more powerful EA should be considered.

Most of traditional searching schemes that we have mentioned in the previous sections require significantly more evaluations when D and d increase. For example, if we consider the Leukemia dataset with 1000 genes, selecting d=100 genes requires 95050 evaluations, much larger than 48775 for selecting d=50 genes. On the contrary, 10000 evaluations are sufficient for our method, the same as for selecting 50 features. Similar scenario can be observed when we increase the value of D. The reason lies on the underlying mechanism of LSEFS whose computational complexity is dominated by the number of samples rather than the number of features. Once the scaling factors have been computed, we can select any number of features without further evaluation. Hence our method is more scalable compared with the other methods.

In this paper, we present a new evaluation criterion for gene selection. Then we propose a new searching scheme to combine with the proposed criterion. The proposed searching scheme allows using continuous EAs more efficiently to solve the gene selection problem. The resultant method, named LSEFS, possesses both better generalization performance and lesser computational cost when compared to three state-of-the-art feature selection methods. It can also better scale to dataset with very large number of genes, which is particularly important for microarray data analysis.

APENDIX A

Note: Some denotations in this appendix may be different from those in the previous sections. Further, all results of this proof is applicable to samples in the kernel space, but for simplicity, we only use **x** instead of $k(\mathbf{x})$.

Solution of the linear system (3) consists of the Lagrangian multipliers α_i 's and b. When employing the whole training set to train the LS-SVM, we use α_p^0 to denote the corresponding Lagrangian multiplier of a training pair ($\mathbf{x}_p$, y_p). In the LOO procedure, $\mathbf{w}_p$ and b_p denote the **w** and b computed using the training set without $\mathbf{x}_p$, hence the LOO error of LS-SVM on $\mathbf{x}_p$ can be written as:

$$looe_p = \frac{1}{2} - \frac{sign\left(y_p\left(\mathbf{w}_p \cdot \mathbf{x}_p + b_p\right)\right)}{2}$$

Based on this expression, we first propose the following Lemma:

Lemma 2: For any training sample $\mathbf{x}_p$ and the corresponding α_p^0, the following equality holds:

$$1 - y_p\left(\mathbf{w}_p \cdot \mathbf{x}_p + b_p\right) = \min_{\boldsymbol{\lambda}} \alpha_p^0 \boldsymbol{\lambda}^T \boldsymbol{\Omega} \boldsymbol{\lambda}$$

where $\Omega = \left\{k\left(x_i, x_j\right)\right\} + \gamma^{-1}\mathbf{I}$, $\boldsymbol{\lambda}=[\lambda_1, \lambda_2, \ldots, \lambda_n]^T$

Subject to the constraint: $\lambda_p = -1, \sum_{i=1}^{n} \lambda_i = 0$

Proof of Lemma 2: The solution of $\boldsymbol{\alpha}$ for an LS-SVM can be achieved by solving the following optimization problem:

$$\max \mathbf{W}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j - \frac{1}{2}\sum_{i=1}^{n} \alpha_i^2 / \gamma \qquad \text{(A1)}$$

$$\text{s.t.} \ \sum_{i=1}^{n} \alpha_i y_i = 0 \qquad \text{(A2)}$$

For the whole set, the vector $\boldsymbol{\alpha}^0=[\alpha_1^0, \alpha_2^0, \ldots, \alpha_n^0]$ maximizes the optimization problem (A1) subjected to the constraint (A2). The decision function is $\mathbf{w}_0 \cdot \mathbf{x} + b_0$, where $\mathbf{w}_0 = \sum_{i=1}^{n} \alpha_i^0 y_i \mathbf{x}_i$. Let us consider the result of the leave-one-out procedure on the training data $\mathbf{x}_p$. The solution $\boldsymbol{\alpha}^p=[\alpha_1^p, \alpha_2^p, \ldots, \alpha_n^p]$ for the procedure is obtained by maximizing the function (A1) subjected to the constraint (A2) and the additional constraint

$$\alpha_p^p = 0 \qquad \text{(A3)}$$

Using the solution, we construct the separating hyperplane $\mathbf{w}_p \cdot \mathbf{x} + b_p = 0$ where $\mathbf{w}_p = \sum_{i=1}^{n} \alpha_i^p y_i \mathbf{x}_i$.

Since $\boldsymbol{\alpha}^p$ maximizes the function (A1) under constraints (A2) and (A3), the following inequality holds.

$$\mathbf{W}(\boldsymbol{\alpha}^p) \geq \mathbf{W}(\boldsymbol{\alpha}^0 - \boldsymbol{\delta}) \qquad \text{(A4)}$$

where the vector $\boldsymbol{\delta}=[\delta_1, \delta_2, \ldots, \delta_n]$ satisfies the following conditions:

$$\delta_p = \alpha_p^0 \qquad \sum_{i=1}^{n} \delta_i y_i = 0 \qquad \text{(A5)}$$

Similarly, since $\boldsymbol{\alpha}^0$ maximizes the function (A1) under constraint (A2), the following inequality holds:

$$\mathbf{W}(\boldsymbol{\alpha}^0) \geq \mathbf{W}(\boldsymbol{\alpha}^p + \boldsymbol{\beta}) \qquad \text{(A6)}$$

where $\boldsymbol{\beta}=[\beta_1, \beta_2, \ldots, \beta_n]$ is a vector satisfying the constraint

$$\sum_{i=1}^{n} \beta_i y_i = 0 \qquad \text{(A7)}$$

We obtain

$$\mathbf{W}(\boldsymbol{\alpha}^p + \boldsymbol{\beta}) - \mathbf{W}(\boldsymbol{\alpha}^p) \leq \mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^p) \leq \mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^0 - \boldsymbol{\delta}) \qquad \text{(A8)}$$

Let us denote $I_1 = \mathbf{W}(\boldsymbol{\alpha}^p + \boldsymbol{\beta}) - \mathbf{W}(\boldsymbol{\alpha}^p)$ and $I_2 = \mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^0 - \boldsymbol{\delta})$ and manipulate them separately.

$$\begin{aligned}
I_1 &= \mathbf{W}(\boldsymbol{\alpha}^p + \boldsymbol{\beta}) - \mathbf{W}(\boldsymbol{\alpha}^p) \\
&= \sum_{i=1}^{n} \beta_i - \sum_{i,j=1}^{n} \alpha_i^p \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j - \frac{1}{2}\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\
&\quad - \sum_{i=1}^{n} \alpha_i^p \beta_i / \gamma - \frac{1}{2}\sum_{i=1}^{n} \beta_i^2 / \gamma \\
&= \sum_{j=1}^{n} \beta_j (1 - y_j \mathbf{w}_p \cdot \mathbf{x}_j - \alpha_j^p / \gamma) \\
&\quad - \frac{1}{2}\left(\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \beta_i^2 / \gamma\right)
\end{aligned}$$

Since $\sum_{i=1}^{n} \beta_i y_i = 0$

$$\begin{aligned}
I_1 &= \mathbf{W}(\boldsymbol{\alpha}^p + \boldsymbol{\beta}) - \mathbf{W}(\boldsymbol{\alpha}^p) \\
&= \beta_p\left[1 - y_p(\mathbf{w}_p \cdot \mathbf{x}_p + b_p)\right] \\
&\quad - \frac{1}{2}\left(\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \beta_i^2 / \gamma\right) \qquad \text{(A9)}
\end{aligned}$$

$$\begin{aligned}
I_2 &= \mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^0 - \boldsymbol{\delta}) \\
&= \sum_{i=1}^{n} \alpha_i^0 - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i^0 \alpha_j^0 y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j - \frac{1}{2}\sum_{i=1}^{n} (\alpha_i^0)^2 / \gamma - \sum_{i=1}^{n} (\alpha_i^0 - \delta_i) \\
&\quad + \frac{1}{2}\sum_{i,j=1}^{n} (\alpha_i^0 - \delta_i)(\alpha_j^0 - \delta_j) y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \frac{1}{2}\sum_{i=1}^{n} (\alpha_i^0 - \delta_i)^2 / \gamma \\
&= \sum_{j=1}^{n} \delta_j (1 - y_j \mathbf{w}_0 \cdot \mathbf{x}_j - \alpha_j^0 / \gamma) + \frac{1}{2}\left(\sum_{i,j=1}^{n} \delta_i \delta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \delta_i^2 / \gamma\right)
\end{aligned}$$

Since $\sum_{i=1}^{n} \delta_i y_i = 0$, similarly we can obtain

$$\begin{aligned}
&\sum_{j=1}^{n} \delta_j (1 - y_j \mathbf{w}_0 \cdot \mathbf{x}_j - \alpha_j^0 / \gamma) + \frac{1}{2}\left(\sum_{i,j=1}^{n} \delta_i \delta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \delta_i^2 / \gamma\right) \\
&= \frac{1}{2}\left(\sum_{i,j=1}^{n} \delta_i \delta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \delta_i^2 / \gamma\right) \qquad \text{(A10)}
\end{aligned}$$

Let $\beta_i = \delta_i = \alpha_i^0 - \alpha_i^p \quad \forall i$, then $I_1=I_2$

$$\beta_p\left[1 - y_p(\mathbf{w}_p \cdot \mathbf{x}_p + b_p)\right] - \frac{1}{2}\left(\sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \beta_i^2 / \gamma\right)$$

$$= \frac{1}{2}\left(\sum_{i,j=1}^{n} \delta_i \delta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \delta_i^2 / \gamma\right)$$

$$(\alpha_p^0 - \alpha_p^p)\left[1 - y_p(\mathbf{w}_p \cdot \mathbf{x}_p + b_p)\right]$$

$$= \alpha_p^0 [1 - y_p (\mathbf{w}_p \cdot \mathbf{x}_p + b_p)]$$
$$= \frac{1}{2} (\sum_{i,j=1}^{n} \delta_i \delta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} \delta_i^2 / \gamma + \sum_{i,j=1}^{n} \beta_i \beta_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$
$$+ \sum_{i=1}^{n} \beta_i^2 / \gamma)$$
$$= \sum_{i,j=1}^{n} (\alpha_i^0 - \alpha_i^p)(\alpha_j^0 - \alpha_j^p) y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j + \sum_{i=1}^{n} (\alpha_i^0 - \alpha_i^p)^2 / \gamma \qquad \text{(A11)}$$

In the leave-one-out procedure, we define the set T_p as a constrained linear combination of all the n-1 training samples $\{\mathbf{x}_i\}_{i \neq p}$:

$$T_p = \left\{ \sum_{i=1, i \neq p}^{n} \lambda_i \mathbf{x}_i : \sum_{i=1, i \neq p}^{n} \lambda_i = 1 \right\}$$

We also define the quantity S_p as the distance between $\mathbf{x}_p$ and this set:

$$S_p^2 = d^2(\mathbf{x}_p, T_p) = \min_{x \in T_p} (\| \mathbf{x}_p - \mathbf{x} \|)^2$$

By setting $\lambda_p = -1$, we can rewrite S_p as:

$$S_p = \min \left\{ (\left\| \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \right\|)^2 : \lambda_p = -1, \sum_{i=1}^{n} \lambda_i = 0 \right\} \qquad \text{(A12)}$$

Finally, we define

$$S_{bp} = \min_{\boldsymbol{\lambda}} \left\{ \boldsymbol{\lambda}^T \boldsymbol{\Omega} \boldsymbol{\lambda} : \lambda_p = -1, \sum_{i=1}^{n} \lambda_i = 0 \right\} = \boldsymbol{\lambda}'^T \boldsymbol{\Omega} \boldsymbol{\lambda}',$$

let $\delta_i = -\alpha_p^0 y_p y_i \lambda_i'$,

since $\mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^p) \leq \mathbf{W}(\boldsymbol{\alpha}^0) - \mathbf{W}(\boldsymbol{\alpha}^0 - \boldsymbol{\delta})$,

then $(\alpha_p^0)^2 \boldsymbol{\lambda}^{*T} \boldsymbol{\Omega} \boldsymbol{\lambda}^* \leq (\alpha_p^0)^2 \boldsymbol{\lambda}'^T \boldsymbol{\Omega} \boldsymbol{\lambda}'$

But according to the definition of λ_i and S_{bp},

$$(\alpha_p^0)^2 \boldsymbol{\lambda}^{*T} \boldsymbol{\Omega} \boldsymbol{\lambda}^* \geq (\alpha_p^0)^2 \boldsymbol{\lambda}'^T \boldsymbol{\Omega} \boldsymbol{\lambda}'$$

Therefore,

$$(\alpha_p^0)^2 \boldsymbol{\lambda}^{*T} \boldsymbol{\Omega} \boldsymbol{\lambda}^* = (\alpha_p^0)^2 \boldsymbol{\lambda}'^T \boldsymbol{\Omega} \boldsymbol{\lambda}' = \alpha_p^0 [1 - y_p (\mathbf{w}_p \cdot \mathbf{x}_p + b_p)]$$

which gives $y_p (\mathbf{w}_p \cdot \mathbf{x}_p + b_p) = 1 - \alpha_p^0 \boldsymbol{\lambda}'^T \boldsymbol{\Omega} \boldsymbol{\lambda}' = 1 - \alpha_p^0 S_{bp}$

Hence, Lemma 2 is proved.

Proof of Lemma 1: To calculate S_{bp}, we need to solve the quadratic problem below:

$$S_{bp} = \min_{\boldsymbol{\lambda}} \boldsymbol{\lambda}^T \boldsymbol{\Omega} \boldsymbol{\lambda} \qquad \text{subject to } \lambda_p = -1, \sum_{i=1}^{n} \lambda_i = 0 \qquad \text{(A13)}$$

Since $\boldsymbol{\Omega}$ is symmetric and positive definite, (A14) can be modified as

$$S_{bp} = \min_{\lambda_i} (\lambda_p \lambda_p \boldsymbol{\Omega}_{pp} + 2\lambda_p \sum_{i=1, i \neq p}^{n} \lambda_i \boldsymbol{\Omega}_{ip} + \sum_{i,j=1, i, j \neq p}^{n} \lambda_i \lambda_j \boldsymbol{\Omega}_{ij})$$

$$= \min_{\lambda_i} (\boldsymbol{\Omega}_{pp} - 2 \sum_{i=1, i \neq p}^{n} \lambda_i \boldsymbol{\Omega}_{ip} + \sum_{i,j=1, i, j \neq p}^{n} \lambda_i \lambda_j \boldsymbol{\Omega}_{ij}) \qquad \text{(A14)}$$

subject to $\sum_{i=1, i \neq p}^{n} \lambda_i = 1$

By introducing a Lagrange multiplier μ, we further modify problem (A14) as

$$S_{bp} = \min_{\lambda_i} \max_{\mu} \left[(\boldsymbol{\Omega}_{pp} - 2 \sum_{i=1, i \neq p}^{n} \lambda_i \boldsymbol{\Omega}_{ip} + \sum_{i,j=1, i, j \neq p}^{n} \lambda_i \lambda_j \boldsymbol{\Omega}_{ij}) + 2\mu (\sum_{i=1, i \neq p}^{n} \lambda_i - 1) \right]$$

$$= \min_{\lambda_i} \max_{\mu} (\boldsymbol{\Omega}_{pp} - 2\mathbf{B}^T \tilde{\boldsymbol{\lambda}} + \tilde{\boldsymbol{\lambda}}^T \mathbf{H}_{/p} \tilde{\boldsymbol{\lambda}}) \qquad \text{(A15)}$$

where $\tilde{\boldsymbol{\lambda}} = [\lambda_1, \lambda_2, ... \lambda_{p-1}, \lambda_{p+1}, \lambda_{p+2}, ... \lambda_n, \mu]^T$, $\mathbf{H} = \begin{bmatrix} \boldsymbol{\Omega} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}$, and $\mathbf{H}_{/p}$ is the submatrix of $\mathbf{H}$ with row and column p removed. $\mathbf{B}$ is the pth column of $\mathbf{H}$ with the pth entry removed and $\mathbf{1}$=[1,1,1…1]^T.

From the fact that the optimal value of $\tilde{\boldsymbol{\lambda}}$ is $\mathbf{H}_{/p}^{-1} \mathbf{B}$, it follows:

$$S_{bp} = \boldsymbol{\Omega}_{pp} - \mathbf{B}^T \mathbf{H}_{/p}^{-1} \mathbf{B} \qquad \text{(A16)}$$

Since for a block matrix:

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}^{-1} = \begin{bmatrix} (\mathbf{A}_{11} - \mathbf{A}_{12} \mathbf{A}_{22}^{-1} \mathbf{A}_{21})^{-1} & -\mathbf{A}_{11}^{-1} \mathbf{A}_{12} \mathbf{F}_{22}^{-1} \\ -\mathbf{F}_{22}^{-1} \mathbf{A}_{21} \mathbf{A}_{11}^{-1} & \mathbf{F}_{22}^{-1} \end{bmatrix}$$

and by sorting the matrix $\mathbf{H}$ (change the position of rows and columns in the matrix, so that the pth row and column become the first row and column respectively), we get

$$\tilde{\mathbf{H}} = \begin{bmatrix} \boldsymbol{\Omega}_{pp} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{H}_{/p} \end{bmatrix}$$

then $$(\tilde{\mathbf{H}}^{-1})_{11} = (\boldsymbol{\Omega}_{pp} - \mathbf{B}^T \mathbf{H}_{/p}^{-1} \mathbf{B})^{-1} = 1 / S_{bp}$$

Because $\tilde{\mathbf{H}}$ is obtained by changing the positions of rows and columns in the matrix $\mathbf{H}$, $(\tilde{\mathbf{H}}^{-1})_{11}$ is just the entry $(\mathbf{H}^{-1})_{pp}$, which gives

$$S_{bp} = 1 / (\mathbf{H}^{-1})_{pp} \qquad \text{(A18)}$$

From equation (A18), we finally get

$$y_p (\mathbf{w}_p \cdot \mathbf{x}_p + b_p) = 1 - \alpha_p^0 / (\mathbf{H}^{-1})_{pp} \qquad \text{(A19)}$$

Therefore, Lemma 1 is proved.

REFERENCES

[1] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp.531-537, 1999.

[2] P. Devijver and J. Kittler, *Pattern recognition: a statistical approach*. Prentice Hall, London, 1982.

[3] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine Learning*, vol. 46, pp. 389-422, 2002.

[4] M. L. Raymer, W. F. Punch, E. D. Goodman, L. A. Kuhn and A. K. Jain, “Dimensionality Reduction Using Genetic Algorithms,” *IEEE Trans. Evolutionary Computation*, vol. 4, pp. 164-171, 2000.

[5] A. Rakotomamonjy, “Variable selection using SVM-based criteria,” *Journal of Machine Learning Research*, vol. 3, pp. 1357-1370, 2003.

[6] X. Zhou and K. Z. Mao, “LS Bound based gene selection for DNA microarray data,” *Bioinformatics*, vol. 21, 1559-1564, 2005.

[7] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.

[8] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, “Choosing multiple parameters for support vector machines,” *Machine Learning*, vol. 46, pp. 131–159, 2002.

[9] J. Platt, “Probabilities for support vector machines,” *Advances in large margin classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. MIT Press , Cambridge, MA, 2000.

[10] Available: http://www.esat.kuleuven.ac.be/~npochet/Bioinformatics/

[11] R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” *Proc. Sixth International Symposium on Micro Machine and Human Sciende (Nagoya, Japan)*, pp. 39-43, 1995.

[12] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2001.