Journal of Biomedical Informatics 129 (2022) 104070

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

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

An efficient model selection for linear discriminant function-based recursive feature elimination

Xiaojian Ding, Fan Yang, Fuming Ma

College of Information Engineering, Nanjing University of Finance and Economics, Nanjing 210023, China

ARTICLE INFO

Keywords:
Support vector machine
Recursive feature elimination
Model selection
Alpha seeding

ABSTRACT

Model selection is an important issue in support vector machine-based recursive feature elimination (SVM-RFE). However, performing model selection on a linear SVM-RFE is difficult because the generalization error of SVM-RFE is hard to estimate. This paper proposes an approximation method to evaluate the generalization error of a linear SVM-RFE, and designs a new criterion to tune the penalty parameter *C*. As the computational cost of the proposed algorithm is expensive, several alpha seeding approaches are proposed to reduce the computational complexity. We show that the performance of the proposed algorithm exceeds that of the compared algorithms on bioinformatics datasets, and empirically demonstrate the computational time saving achieved by alpha seeding approaches.

1. Introduction

Gene selection is now widely used in genomic sciences, such as distinguishing driver mutations from driver genes [10,24], discovering significant features for drug sensitivity prediction [1,18], or identifying panels of biomarkers to aid in cancer prognosis [30,37], etc. Given microarray data with thousands of genes and tens or hundreds of samples, the gene selection task is to select the most informative genes that are relevant to a specific classification task.

Many gene selection algorithms have been studied in the literature. From a computational perspective, these approaches can be broadly divided into three groups, namely, filter, wrapper, and embedded. Filter methods rank genes by measuring the relevance between genes and class labels. They are independent of classifiers and are often computationally more efficient compared to wrapper and embedded methods. However, the performance of an inductive algorithm is not guaranteed [6]. In contrast, wrapper and embedded methods rely on a classifier, using classification accuracy as an indication of feature quality. They use the prediction provided by a classifier to evaluate subsets of features. Wrapper methods incorporate the performance of a classifier as a evaluation criterion to choose the best gene subset. However, they often suffer from overfitting problems and the computational complexity is large [34]. Embedded methods select genes as part of the model construction process, and are specific to given learning algorithms, and therefore may be more efficient than the other two types methods [3]. Recursive feature elimination (RFE) algorithms are typical approaches in the literature [22,15]. Among them, the most widely used RFE algorithm is support vector machine-based recursive feature elimination (SVM-RFE) [22], which is first introduced for microarray data analysis. However, it is a significant challenge to estimate parameters of SVM-RFE from a limited number of samples. SVM-RFE recursively eliminates redundant genes using coefficients computed by an SVM classifier. Therefore, parameters of SVM-RFE are exactly same as the parameters of SVM. As SVM is very sensitive to model parameters, the performance of SVM-RFE should also be sensitive to the model parameters. If inappropriate values are selected, SVM-RFE may fail to generate the optimal subset of genes. Therefore, model selection is an important step in SVM-RFE.

Generally, SVM-RFE has linear and nonlinear versions, depending on the type of SVM classifier it incorporated. In the case of kernel SVM, SVM-RFE at least has two model parameters: the kernel parameter and the penalty parameter *C*. If random kernels are used [13,12], more kernel parameters should be considered. In the case of a linear SVM, model selection is the process of tuning regularization parameter *C* such that SVM achieves the best generalization performance for a complete set of features. General speaking, generalization performance is measured by the error rate of a classifier over the test set [27,45]. In this paper, we focus on the model selection problem of linear SVM-based RFE algorithm termed as linear SVM-RFE. Over the past two decades, a large number of model selection methods have been introduced to improve the classification accuracy. Intelligent and bioinspired methods such as genetic algorithm (GA) [44], particle swarm optimization (PSO)

https://doi.org/10.1016/j.jbi.2022.104070
Received 23 November 2021; Received in revised form 2 April 2022; Accepted 5 April 2022
Available online 15 April 2022

[5], gravitational search algorithm (GSA) [26] and gradient-based algorithms [4,31], are popular choices to tune hyperparameters of SVM. Although they operate model selection in a small parameter space, they suffer from model selection problems of additional algorithm parameters (e.g., fitness function, learning rate and stopping criteria) and low convergence speed. The work in [40] used cross-validation (CV) to optimize the hyperparameters of SVM to minimize the CV error estimate. The works in [7,39,23,38] suggested to tune the hyperparameters of SVM using internal metrics, such as Xi-alpha bound, approximate span bound, radius-margin bound and distance between two classes (DBTC). In [17] these internal metrics were compared with CV on 110 benchmark datasets, and it was shown that CV performs the best in terms of the expected error on unseen data. The work in [28] proposed a granularity selection criterion to reduce the computational cost of CV procedure. In addition, the work in [41] suggested a guideline to implement cross-validation more effectively. However, few guidelines were provided to tune parameter of SVM-RFE.

The model selection step of linear SVM-RFE aims to find an appropriate value of regularization parameter C such that SVM achieves the best generalization performance on the selected gene subsets. SVM-RFE requires training d linear SVMs with a decreasing number of features, where d is the number of features in the dataset. A natural way to tune parameter of SVM-RFE is to perform model selection before every SVM training process. The work in [29] used an adaptive kernel width criterion to find the optimal parameters of SVM recursively. The work in [47] studied a two-step cross-validation to find the optimal parameters of SVM models, and then conducted the SVM-RFE algorithm. Work in [35] directly adopted previous SVM model selection strategies in the SVM-RFE process. However, these works did not consider the generalization performance of SVM for the optimum gene subset.

In this paper, we propose an approximation strategy to evaluate the generalization performance of a linear SVM-RFE, and design a new criterion to tune the penalty parameter C. Because the computational complexity of the proposed algorithm is high, we suggest several alpha seeding strategies to reduce the computational cost of the proposed algorithm. The effectiveness and efficiency of the proposed algorithms together with nine state-of-the-art algorithms are validated by a series of experiments using bioinformatics benchmark datasets. The remainder of the paper is organized as follows. In Section 2, we briefly review the related work. In Section 3, we present a model selection algorithm for linear SVM-RFE. In Section 4, we suggest several alpha seeding strategies to reduce the computational cost of the proposed algorithm. In Section 5, we further discuss our experimental results and finally, we conclude the paper in Section 6.

2. Related work

This section introduces a linear SVM-RFE algorithm. Given a binary classification dataset $X = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$ is the target value of $\mathbf{x}_i$, and d is the number of features.

2.1. Linear support vector machine

The goal of linear SVM is to find an optimal hyperplane that best separates the two classes. Linear SVM is regarded as a margin maximization problem, which leads to the following optimization problem

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^N \xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}\cdot\mathbf{x}_i + b) + \xi_i \geqslant 1 \\ & \xi_i \geqslant 0, \quad i = 1, \ldots, N, \end{aligned} \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ denote the weight vector and the bias term, respectively, C is a penalty parameter, and ξ_i are slack variables that allow some training samples to fall out of the margin.

Using Lagrange for solving problem (1) and introducing a set of Lagrange multipliers α_i, this yields the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \mathbf{x}_i \mathbf{x}_j - \sum_{i=1}^N \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^N y_i \alpha_i = 0, \ 0 \leqslant \alpha_i \leqslant C. \end{aligned} \tag{2}$$

Let $\boldsymbol{\alpha}$ solve the dual problem (2), then the weight vector $\mathbf{w}$ is solved as follows

$$\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i. \tag{3}$$

2.2. Linear SVM-RFE

A linear SVM-RFE starts with all feature variables, ranks them based on the criteria of the weight vector $\mathbf{w}$ of a linear SVM, and eliminates the feature with the lowest ranking score. This process is repeated until the optimality condition is met. At each loop of the SVM-RFE, the coefficients of $\mathbf{w}$ in Eq. (3) are used to compute the feature ranking score. This means that SVM-RFE requires training SVM d times.

The linear SVM-RFE has a parameter C that needs to be tuned. Model selection of a linear SVM-RFE refers to the process of tuning regularization parameter C such that a linear SVM achieves the best feature selection performance. The best feature selection performance is usually evaluated by the classification accuracy of a SVM classifier for the optimum number of features. However, SVM-RFE does not generate the optimal number of features, which makes the best feature selection performance very hard to estimate.

3. Model selection algorithm of linear SVM-RFE

3.1. Approximation performance for the optimum number of features

SVM-RFE is an iterative algorithm that works by fitting a linear SVM on an initial set of features. Before training such a linear SVM, model selection should be conducted. The goal of model selection is to tune the parameter C to achieve the lowest generalization error. Generalization error is the expected prediction error of SVM over a test dataset for all features. However, we cannot perform model selection through the generalization error on a high dimensional microarray data with many irrelevant or redundant features. This is because a linear SVM is hard to distinguish all sample categories when the number of samples is far smaller than the number of features. Among these features, only very few related features are important to well distinguish sample categories.

Although SVM-RFE generates a lot of nested subsets of features, it does not specify the optimum subset of features. Many literatures studied how to evaluate the generalization performance of SVM-RFE. The work in [42] evaluated the performance of partial least squares based recursive feature elimination (PLS-RFE) for feature subsets varied from 1 to 50. Work in [46] evaluated the performance of SVM-RFE for feature subsets varied from 1 to 400. The work in [19] presented the classification accuracies for the number of selected features equals to 50, 100, 200, 1,000 and all features. Work in [32] presented testing results of the classification for feature subsets in the range of [10:10:140].

To approximate the best performance of a linear SVM-RFE, we evaluate the performance of linear SVM-RFE for feature subsets varied over $[ra : rb : rc]$ when combined with a linear SVM classifier, where ra, rb, and rc are user specified parameters. This means that we can compute the generalization error of a SVM for $(rc - ra)/rb + 1$ nested subsets of features generated by the linear SVM-RFE.

3.2. Model selection criterion

In this section, we aim to answer two questions: "How do we measure the generalization error of SVM for $(rc-ra)/rb+1$ nested subsets of features?" and "How should we integrate these results to perform model selection?"

To answer the first question, we study a K-fold cross-validation resampling strategy. All training samples are divided into K disjoint sets of approximately equal size. In each of the k iterations, a linear SVM is trained on $K-1$ sets and tested on the other set, and the mean classification error rates over k test sets are reported. For each nested subset of features, we obtain a mean classification error rate.

To select the best value of parameter C, we give a candidate value set of parameter C. For each candidate value of C, we run SVM-RFE one time, and finally obtain $(rc-ra)/rb+1$ mean classification error rates. Suppose that the number of mean classification error rates is M. Let $\boldsymbol{\zeta}_i$be the error rate vector of the ith linear SVM-RFE, and ζ_{ij} be the error rate value associated with the jth nested subset of features.

We define the following model selection criterion of a linear SVM-RFE

$$c_i = \frac{\sigma_{\zeta_i}}{\overline{\boldsymbol{\zeta}_i}}, \tag{4}$$

where $\overline{\boldsymbol{\zeta}_i}$ and σ_{ζ_i} are mean and standard deviation of vector $\boldsymbol{\zeta}_i$,

$$\overline{\boldsymbol{\zeta}_i} = \frac{1}{M}\sum_{j=1}^{M}\zeta_{ij}, \tag{5}$$

and

$$\sigma_{\zeta_i} = \sqrt{\frac{\sum_{j=1}^{M}(\zeta_{ij}-\overline{\boldsymbol{\zeta}_i})^2}{M-1}}. \tag{6}$$

Eq. (4) demonstrates the ratio of mean to its standard deviation, which measures the relative variability of mean classification error rates. Smaller c_i denotes that the mean classification error rates have small relative variability among nested subsets of features. Thus, the best value of parameter C can be selected by the minimum value of c_i. In summary, Algorithm 1 describes a model selection algorithm of linear SVM-RFE, which we abbreviate as "model selection SVM-RFE"(MS-SVM-RFE). The algorithm takes five parameters: 1) candidate value set S of parameter C; 2) several nested subsets of features $[ra : rb : rc]$; 3) the partition factor K in K-fold.

Algorithm 1. MS-SVM-RFE algorithm

Algorithm 1: MS-SVM-RFE algorithm

Input: $X = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^d$, candidate value set S of parameter C, ra, rb, rc, K

```
1  Initialize err ← 0 ;
2  for i = 1 to length of S do
3      C ← S(i) ;
4      Implement a linear svm-rfe r ← linear SVM-RFE(X, C) ;
5      for j = 1 to (rc − ra)/rb + 1 do
6          Generate a nested subset of features r_j by r ;
7          for k = 1 to K do
8              Generate a training set Tr and a test set Te ;
9              Training a SVM classifier SVM-train(Tr, r_j, C) ;
10             Test a SVM classifier errnew ← SVM-test(Te, r_j, C) ;
11             err ← err + errnew ;
12         end
13         ζ_ij ← err/K ;
14     end
15 end
16 Compute c_i by Equation (4) ;
17 Find the feature with the lowest value: f ←argminc_i ;
18 Return the optimum value S(f) ;
```

The main idea of Algorithm 1 is to obtain mean classification error rates for $(rc-ra)/rb+1$ nested subsets of features, which are computed by the K-fold cross-validation resampling strategy, and we use these mean classification error rates as the approximation of the generalization error of a linear SVM-RFE. Line 4 implements a linear SVM-RFE algorithm on a dataset and obtains a feature ranking list r. According to the ranking list r, line 6 generates the nested subsets of features in the sequence of $r_1, r_2, \ldots, r_{((rc-ra)/rb+1)}$. In the nested subset r_1, there are ra top-ranked features in the ranking list r. In the nested subset r_2, there are $ra+rb$ top-ranked features in the ranking list r. Similarly, the nested subset r_i has $ra+rb*(i-1)$ top-ranked features in the ranking list r, and so on. Lines 11–13 calculate the error rate value ζ_{ij} associated with the ith linear SVM-RFE and the jth nested subset of features.

4. Implementation details

4.1. Time complexity analysis

In this section, we analyze the time complexity of Algorithm 1. Suppose N is the number of instances, d is the number of features in each instance, m is the number of elements in the set S, n is the number of nested subsets of features. The main computational cost of Algorithm 1 involves the computation of multiple linear SVM-RFEs and the

Table 1
Notation of SVM models for each value of parameter C. In each row, all SVMs are trained on the same samples with different features. In each column, all SVMs are trained on the different samples with same features.

	Number of features			
Fold	ra	$ra+rb$	…	rc
1	11th SVM	12th SVM	1jth SVM	1nth SVM
2	21th SVM	22th SVM	2jth SVM	2nth SVM
…	k1th SVM	k2th SVM	kjth SVM	knth SVM
K	K1th SVM	K2th SVM	kJth SVM	Knth SVM

measurement process of the generalization error of the linear SVM-RFE.

A linear SVM-RFE has a time complexity of $O(Nd^2/2)$ if one feature is eliminated from the current feature set. Then the time complexity related to linear SVM-RFE in Algorithm 1 is $O(mNd^2/2)$. In addition, measurement process requires training linear SVM Kmn times. In each learning component, linear SVM has a time complexity of $O(N(K-1)(ra+rb)/2K)$. Therefore, Algorithm 1 has a total time complexity of $O(mNd^2/2+Nmn(K-1)(ra+rb)/2)$.

4.2. Alpha seeding methods

Training a SVM requires solvinga quadratic programming (QP) problem (2). Generally, problem (2) can be solved in three ways: active set method [16], interior point method [11], and sequential minimal optimization (SMO) method [33]. In this paper, we implement the idea of alpha seeding in the context of training SVM using an active set method. Algorithm 1 involves solving successive QP problems of SVM, then alpha seeding can be employed to reduce the computational cost. Alpha seeding refers to the strategy of seeding the next SVM training using the solution of previous SVM training [9]. It has been proved to be a powerful strategy to measure generalization error by leave one out (LOO) [25], GrowC [8] and K-fold cross validation [43]. In this section we aim to analyze how this method is incorporated into solving successive QP problems in the Algorithm 1.

In the first part of Algorithm 1, linear SVM-RFE is implemented m times. For each SVM-RFE, d linear SVMs are trained with a decreasing number of features. The work in [14] suggested a direct alpha seeding method to reuse the solution of the ith optimized extreme learning machine (OELM) for training the $(i+1)$th OELM. As the optimization problem of OELM is similar to SVM, this alpha seeding method can be used to accelerate the training process of linear SVM-RFE.

Suppose $\boldsymbol{\alpha}$ is the solution of the ith SVM, then the initial point $\widetilde{\boldsymbol{\alpha}}$ of the $(i+1)$th SVM can be set as

$$\widetilde{\boldsymbol{\alpha}} \equiv \boldsymbol{\alpha} \tag{7}$$

This alpha seeding method is work for linear SVM-RFE because two successive SVMs share the same training samples.

In the second part of Algorithm 1, linear SVM is trained $m*\ n*\ K$ times, which means a linear SVM is trained $n*\ K$ times for each candidate value of parameter C. The details of these $n*\ K$ SVM models are listed in Table 1. In order to apply alpha seeding strategy to train all SVMs in Table 1, we first train the 11th SVM with a feasible initial point, and obtain a solution $\boldsymbol{\alpha}^{11}$. Then we use two alpha seeding methods to train other SVMs.

The first alpha seeding method is used to reusing the solution of 11th SVM to train later k1th SVMs. For example, use $\boldsymbol{\alpha}^{11}$ to train 21th SVM and obtain a solution $\boldsymbol{\alpha}^{21}$, and use $\boldsymbol{\alpha}^{21}$ to train 31th SVM, and so on. This alpha seeding method is discussed in the work [43], which refers to multiple instance replacement (MIR).

The K-fold cross-validation divides all training samples into K subsets of approximately equal size. Take 11th SVM and 21th SVM for example, we train the 11th SVM on 2st to Kth subsets, and train the 21th SVM on 1st subset and 3st to Kth subsets. Then, 11th SVM and 21th SVM share $(K-2)$ subsets, denote by T. The unshared training set in the 11th SVM is denoted by U, and the unshared training set in the 21th SVM is denoted by V. We also denote $I_U=\{i|\mathbf{x}_i\in U\}, I_V=\{i|\mathbf{x}_i\in V\}$, and $I_T=\{i|\mathbf{x}_i\in T\}$.

When we train the 11th SVM, the solution of alpha values must satisfy the equality constraint of Eq. (2)

$$\sum_{u\in I_U} y_u\boldsymbol{\alpha}_u^{11}+\sum_{t\in I_T} y_t\boldsymbol{\alpha}_t^{11}=0. \tag{8}$$

The initial alpha values of the 21th SVM also must satisfy the same constraint

$$\sum_{v\in I_V} y_v\boldsymbol{\alpha}_v^{11}+\sum_{t\in I_T} y_t\boldsymbol{\alpha}_t^{21}=0. \tag{9}$$

Following the intuition of support vectors, the alpha values of the shared subsets between the two SVMs are set the same value (i.e. $\boldsymbol{\alpha}_t^{21}=\boldsymbol{\alpha}_t^{11}$). Then, we must satisfy the following constraint

$$\sum_{u\in I_U} y_u\boldsymbol{\alpha}_u^{11}=\sum_{v\in I_V} y_v\boldsymbol{\alpha}_v^{21}. \tag{10}$$

Suppose the decision function of a linear SVM is

$$f(\mathbf{x})=\sum_{i=1}^{N}\alpha_i y_i\mathbf{x}_i\cdot\mathbf{x}+b, \tag{11}$$

the output of the 11th SVM can be expressed as

$$f_i^{11}=\sum_{u\in I_U} y_u\alpha_u^{11}\mathbf{x}_i\cdot\mathbf{x}_u+\sum_{t\in I_T} y_t\alpha_t^{11}\mathbf{x}_i\cdot\mathbf{x}_t+b, \tag{12}$$

where f_i^{11} denotes the output of the 11th SVM for the ith sample. The output of the 21th SVM can be expressed as

$$f_i^{21}=\sum_{v\in I_V} y_v\alpha_v^{21}\mathbf{x}_i\cdot\mathbf{x}_v+\sum_{t\in I_T} y_t\alpha_t^{21}\mathbf{x}_i\cdot\mathbf{x}_t+b. \tag{13}$$

Subtracting Eq. (13) from Eq. (12) gives that

$$\triangle f_i=\sum_{v\in I_V} y_v\alpha_v^{21}\mathbf{x}_i\cdot\mathbf{x}_v-\sum_{u\in I_U} y_u\alpha_u^{11}\mathbf{x}_i\cdot\mathbf{x}_u. \tag{14}$$

To satisfy the KKT conditions of (2)

$$\begin{cases}\alpha_i=0\Leftrightarrow y_i f_i\geqslant 1\\ 0<\alpha_i<C\Leftrightarrow y_i f_i=1,\\ \alpha_i=C\Leftrightarrow y_i f_i\leqslant 1\end{cases} \tag{15}$$

$\triangle f_i$ should be expressed as

$$\begin{cases}\triangle f_i=1-f_i, \text{when } \alpha_i=0 \text{ or } \alpha_i=C\\ \triangle f_i=0, \text{when } 0<\alpha_i<C.\end{cases} \tag{16}$$

Table 2
Benchmark datasets.

Dataset	#Total	#Training	#Test	#Positive	#Negative	#Features
Colon Tumor	62	30	32	22	40	2000
Leukemia	72	38	34	47	25	7129
Lung Cancer	181	32	149	31	150	12533
Prostate Cancer	136	102	59	77	34	12600

Table 3
Best parameter C of compared algorithms.

Algorithm	Colon Tumor	Leukemia	Lung Cancer	Prostate Cancer
Kf3	2^{-9}	2^{-7}	2^{0}	2^{0}
Kf5	2^{-8}	2^{-8}	2^{1}	2^{0}
Kf10	2^{4}	2^{-8}	2^{0}	2^{1}
5xho	2^{-9}	2^{-6}	2^{-8}	2^{-3}
10xho	2^{-8}	2^{3}	2^{-8}	2^{-1}
20xho	2^{-9}	2^{-7}	2^{-7}	2^{-3}
50/50	2^{5}	2^{0}	2^{-3}	2^{-3}
20/80	2^{10}	2^{1}	2^{-4}	2^{-4}
80/20	2^{7}	2^{2}	2^{-2}	2^{-4}
MSR	2^{-6}	2^{-3}	2^{2}	2^{5}

By Eqs. (10) and (14), $\boldsymbol{\alpha}_V^{21}$ can be computed by the following system

$$\begin{bmatrix} \mathbf{y}\circ\triangle\mathbf{f} + \boldsymbol{\alpha}_U^{11}\mathbf{x}_X\cdot\mathbf{x}_U \\ \mathbf{y}_U^T\cdot\boldsymbol{\alpha}_U^{11} \end{bmatrix} = \begin{bmatrix} x_X\cdot\mathbf{x}_V \\ \mathbf{y}_V^T \end{bmatrix}\boldsymbol{\alpha}_V^{21}, \tag{17}$$

where $\triangle\mathbf{f} = [\triangle f_1, \ldots, \triangle f_N] \in \mathbb{R}^d$. The second alpha seeding method is used to reuse the solution of $k1$th SVM to train later kjth SVMs. For example, use $\boldsymbol{\alpha}^{k1}$ to train the $k2$th SVM and obtain a solution $\boldsymbol{\alpha}^{k2}$, and use $\boldsymbol{\alpha}^{k2}$ to train the $k3$th SVM, and so on. Take 11th SVM and 21th SVM for example, they share the same subsets with different features. Based on the intuition of support vectors, the solution of the kjth SVM can be directly used as the initial point of the $k(j+1)$th SVM by Eq. (7).

5. Experimental study

5.1. Parameter setting

In this section, we compare our MS-SVM-RFE (MSR) algorithm[1] with three state-of-the-art resampling procedures described in the work [41]: K-fold cross-validation, Hold-out, K-times repeated hold-out. More specifically, 9 resampling procedures are employed:

1. 3-fold cross-validation (kf3).
2. 5-fold cross-validation (kf5).
3. 10-fold cross-validation (kf10).

Table 4
Mean accuracy (%) of 10 compared algorithms. In each column, the best results are shown in boldface.

Dataset	algorithm	Number of features									
		10	20	30	40	50	60	70	80	90	100
Colon	Kf3	63.75	62.50	63.59	63.91	63.59	63.75	65.63	66.41	66.41	68.91
	Kf5	62.50	63.75	63.75	64.84	66.25	69.53	**69.69**	69.69	71.09	71.09
	Kf10	**70.00**	61.88	63.44	61.72	66.56	67.34	65.94	70.00	70.63	72.81
	5xho	63.75	62.50	63.59	63.91	63.59	63.75	65.64	66.41	66.41	68.91
	10xho	62.50	63.75	63.75	64.84	66.25	69.53	69.69	69.69	71.09	71.09
	20xho	63.75	62.50	63.59	63.91	63.59	63.75	65.63	66.41	66.41	68.91
	50/50	65.16	64.22	63.44	61.72	66.56	67.34	65.94	70.00	70.63	72.81
	20/80	66.09	64.53	63.44	61.72	66.56	67.34	65.94	70.00	70.63	72.81
	80/20	66.09	64.69	63.44	61.72	66.56	67.34	65.94	70.00	70.63	72.81
	MSR	64.39	**71.56**	**75.47**	**72.03**	**72.03**	**71.09**	68.75	**70.31**	**71.56**	**75.47**
Leukemia	Kf3	68.68	72.35	76.32	77.21	78.97	82.35	83.82	89.41	89.41	**91.91**
	Kf5	67.21	64.71	67.06	66.32	69.12	70.00	73.68	76.03	76.76	79.85
	Kf10	67.21	64.71	67.06	66.32	69.12	70.00	73.68	76.03	76.76	79.85
	5xho	73.09	77.65	81.76	86.03	**85.59**	83.53	**87.50**	**88.68**	**87.94**	88.24
	10xho	78.82	72.06	80.59	80.88	80.00	**83.68**	85.29	85.44	87.21	89.41
	20xho	68.68	72.35	76.32	77.21	78.97	82.35	83.82	89.41	89.41	**91.91**
	50/50	79.41	73.97	80.59	80.88	80.00	**83.68**	85.29	85.44	87.21	89.41
	20/80	**80.71**	75.00	80.59	80.88	80.00	**83.68**	85.29	85.44	87.21	89.41
	80/20	77.06	75.74	80.59	80.88	80.00	**83.68**	85.29	85.44	87.21	89.41
	MSR	80.15	**86.03**	**90.29**	**88.24**	83.53	**83.68**	85.15	85.44	87.21	89.41
Lung	Kf3	92.89	93.29	**96.34**	95.94	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	Kf5	92.95	93.15	**96.34**	95.94	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	Kf10	92.89	93.29	**96.34**	95.94	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	5xho	82.48	84.33	89.06	91.17	92.32	93.19	93.66	94.23	95.77	96.41
	10xho	82.48	84.33	89.06	91.17	92.32	93.19	93.66	94.23	95.77	96.41
	20xho	82.68	87.38	91.31	92.72	94.73	95.57	96.38	96.41	96.71	97.08
	50/50	93.12	**93.79**	94.09	**96.51**	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	20/80	90.84	93.66	95.20	95.37	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	80/20	94.30	93.52	**96.34**	95.94	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
	MSR	**96.11**	93.15	**96.34**	95.94	**96.85**	**97.18**	**97.68**	**97.75**	**97.68**	**97.92**
Prostate	Kf3	64.12	72.35	76.18	77.06	77.35	81.03	85.00	84.85	87.06	88.38
	Kf5	64.12	72.35	76.18	77.06	77.35	81.03	85.00	84.85	87.06	88.38
	Kf10	**77.21**	76.03	79.71	79.12	80.59	82.79	87.35	**89.41**	**90.44**	**90.59**
	5xho	50.15	65.88	66.03	66.32	66.18	66.47	67.21	70.59	70.88	70.44
	10xho	58.82	66.32	71.91	73.09	73.97	75.74	77.21	81.62	81.62	82.21
	20xho	50.15	65.88	66.03	66.32	66.18	66.47	67.21	70.59	70.88	70.44
	50/50	50.15	65.88	66.03	66.32	66.18	66.47	67.21	70.59	70.88	70.44
	20/80	50.44	50.88	50.88	51.32	54.26	65.88	66.03	65.44	65.15	65.44
	80/20	50.44	50.88	50.88	51.32	54.26	65.88	66.03	65.44	65.15	65.44
	MSR	70.59	**77.79**	**89.71**	**87.06**	**85.44**	**86.62**	**87.50**	86.18	88.53	87.50

[1] https://github.com/SVMrelated/mssvmrfe

4. 5 times repeated hold out (5xho).
5. 10 times repeated hold out (10xho).
6. 20 times repeated hold out (20xho).
7. 50/50 hold out (50/50) — training and test sets of 50%.
8. 20/80 hold out (20/80) — training set of 20% and test set of 80%.
9. 80/20 hold out (80/20) — training set of 80% and test set of 20%.

We implement all algorithms in MATLAB, and conduct the experiments on a laptop with a *i*7-7700HQ CPU @ 2.80 GHz and 8 GB RAM. We present empirical results on four public datasets: 1) Colon Tumor [2]; 2) Leukemia [20]; 3) Lung Cancer [21]; and 4) Prostate Cancer [36]. These benchmark datasets are downloaded from the website https://leo.ugr.es/elvira/DBCRepository/. Colon Tumor includes 62 samples gathered from colon-cancer patients. Among them, 40 tumor biopsies are from tumors (labelled as "negative") and 22 normal (labelled as "positive") biopsies are from healthy parts of the colons of the same patients. Two thousand out of around 6500 genes were selected based on the confidence in the measured expression levels. Leukemia includes 72 leukemia samples, over 7129 probes from 6817 human genes. Among 72 leukemia samples, 47 in class Acute Lymphoblastic Leukemia (ALL) and 25 in class Acute Myeloid Leukemia (AML). Lung Cancer includes 181 tissue samples of each sample is described by 12533 genes. The 181 tissue samples include 31 samples of malignant pleural mesothelioma (labelled as MPM) and 150 samples of adenocarcinoma (labelled as ADCA). Prostate Cancer includes 136 prostate samples with around 12600 genes. The 136 prostate samples contain 77 prostate tumor samples (labelled as "negative") and 59 non-tumor samples (labelled as "positive"). Table 2 shows the details of the datasets.

For all compared algorithms combined with a linear SVM, we vary C in the range of $\{2^{-9}, 2^{-8}, \ldots, 2^{0}, \ldots, 2^{14}, 2^{15}\}$. For MSR algorithm, parameter K is set as 10, ra, rb and rc are set as 10, 10 and 100, respectively.

5.2. Accuracy

In the MSR algorithm, we evaluate feature ranking lists generated by multiple linear SVM-RFE algorithms and select a feature ranking list of the best performance when combined with a linear SVM. Once a best parameter C value is selected, we implement a linear SVM-RFE with this value, and obtain a feature ranking list. The best parameter values of these compared algorithms are reported in Table 3.

In this section, we compare the generalization performance of these feature ranking lists using a linear SVM classifier. The 20-times repeated hold-out procedure and the range of the number of selected features in

Table 5
Standard deviation of accuracy (%) of 10 compared algorithms.

Dataset	algorithm	Number of features									
		10	20	30	40	50	60	70	80	90	100
Colon	Kf3	6.83	5.07	6.90	7.41	6.82	6.98	8.36	8.17	8.48	9.85
	Kf5	5.07	6.91	7.20	9.23	8.27	9.45	9.57	9.41	9.88	10.1
	Kf10	9.90	9.81	9.02	8.04	10.2	7.13	6.87	6.52	7.61	8.49
	5xho	6.83	5.07	6.90	7.41	6.82	6.98	8.36	8.17	8.48	9.85
	10xho	5.07	6.91	7.20	9.23	8.27	9.45	9.57	9.41	9.88	10.1
	20xho	6.83	5.07	6.90	7.41	6.82	6.98	8.36	8.17	8.48	9.85
	50/50	7.94	10.7	9.02	8.04	10.2	7.13	6.87	6.52	7.61	8.49
	20/80	11.07	9.03	9.02	8.04	10.2	7.13	6.87	6.52	7.61	8.49
	80/20	11.07	9.07	9.02	8.00	10.2	7.13	6.87	6.52	7.61	8.49
	MSR	7.68	8.72	9.42	8.92	7.75	9.50	7.85	5.87	6.79	6.82
Leukemia	Kf3	11.14	12.05	13.65	14.05	14.39	14.84	13.91	8.97	9.51	6.18
	Kf5	10.25	6.18	9.28	8.77	11.27	12.24	13.65	14.13	14.09	14.79
	Kf10	10.25	6.18	9.28	8.77	11.27	12.24	13.65	14.13	14.09	14.79
	5xho	10.64	12.49	11.31	8.74	8.42	6.57	7.13	5.35	5.80	6.18
	10xho	5.61	7.78	5.91	7.90	6.08	6.22	4.37	4.82	4.50	4.20
	20xho	11.14	12.05	13.65	14.05	14.39	14.84	13.91	8.97	9.51	6.18
	50/50	6.18	7.83	5.91	7.90	6.08	6.22	4.37	4.82	4.50	4.20
	20/80	5.60	6.91	5.91	7.90	6.08	6.22	4.37	4.82	4.50	4.20
	80/20	8.57	8.32	5.91	7.90	6.08	6.22	4.37	4.82	4.50	4.20
	MSR	8.69	5.47	4.49	4.87	6.57	5.18	4.62	5.68	4.50	4.20
Lung	Kf3	3.15	2.79	5.14	5.64	5.23	4.94	4.85	4.65	4.59	4.14
	Kf5	2.97	3.97	5.14	5.64	5.23	4.94	4.85	4.65	4.59	4.14
	Kf10	3.15	2.79	5.14	5.64	5.23	4.94	4.85	4.65	4.59	4.14
	5xho	1.27	3.99	8.08	8.50	8.63	8.64	8.66	8.19	7.15	6.66
	10xho	1.27	3.99	8.08	8.50	8.63	8.64	8.66	8.19	7.15	6.66
	20xho	1.58	6.60	7.89	7.22	6.70	6.86	6.56	6.60	6.39	6.11
	50/50	5.51	4.27	4.82	5.71	5.23	4.94	4.85	4.65	4.59	4.14
	20/80	6.67	4.92	5.12	5.08	5.23	4.94	4.85	4.65	4.59	4.14
	80/20	4.06	4.01	5.14	5.64	5.23	4.94	4.85	4.65	4.59	4.14
	MSR	1.50	3.97	5.14	5.64	5.23	4.94	4.85	4.65	4.59	4.14
Prostate	Kf3	8.02	7.83	6.81	5.93	6.48	8.34	7.45	6.56	6.57	5.35
	Kf5	8.02	7.83	6.81	5.93	6.48	8.34	7.45	6.56	6.57	5.35
	Kf10	5.55	5.42	6.39	6.60	7.42	7.59	6.34	5.43	4.46	3.39
	5xho	8.92	9.51	9.18	9.42	9.13	8.56	7.71	8.26	8.74	8.67
	10xho	9.35	8.45	8.12	8.55	8.06	7.63	7.13	7.80	7.20	7.48
	20xho	8.92	9.51	9.18	9.42	9.13	8.56	7.71	8.26	8.74	8.67
	50/50	8.92	9.51	9.18	9.42	9.13	8.56	7.71	8.26	8.74	8.67
	20/80	9.31	9.31	9.40	9.42	6.36	9.37	9.13	9.15	9.22	9.25
	80/20	9.31	9.31	9.40	9.42	6.36	9.37	9.13	9.15	9.22	9.25
	MSR	6.04	5.76	5.18	5.35	5.68	6.15	6.18	5.06	6.06	6.10

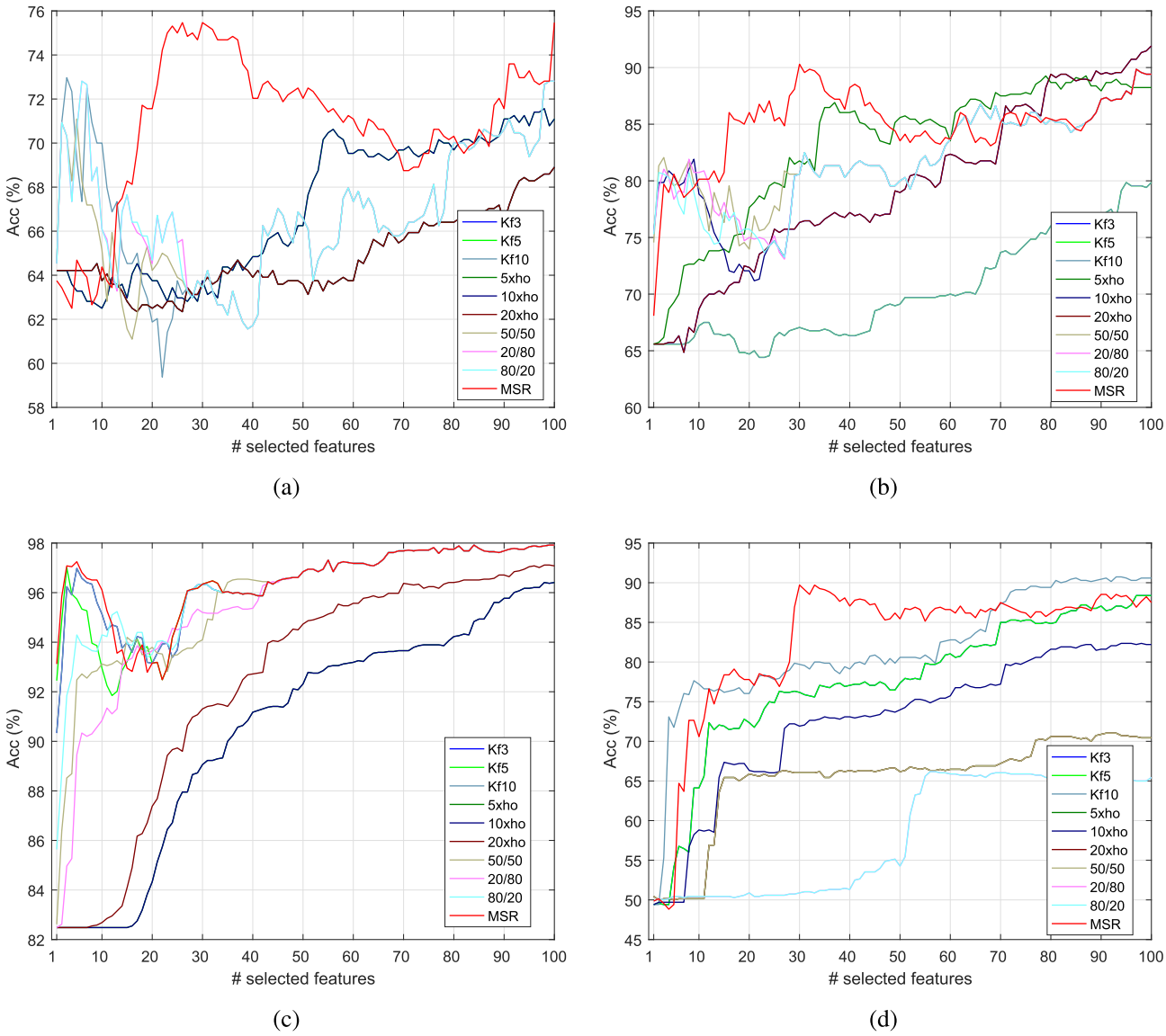

Fig. 1. Mean accuracy as well as the number of selected features of 10 compared algorithms on (a) Colon Tumor, (b) Leukemia, (c) Lung Cancer and (d) Prostate Cancer datasets.

Table 6
Computational time (s) comparison between with alpha seeding and without alpha seeding.

Dataset	RFE-O	MEA-O	RFE-AS	MEA-AS
Colon Tumor	402	85.15	58.6	58.6
Leukemia	2237	462	711	450
Lung Cancer	8125	1504	5017	1470
Prostate Cancer	11776	1156	4884	1085

[10, 20, 30,…,100] is used in this experiment for performance evaluation. The mean and standard deviation of accuracy of 20-times repeated hold-out procedure are reported in Tables 4 and 5.

For Colon dataset, MSR wins the best performance on 8 nested subsets, which contain 20, 30, 40, 50, 60, 80, 90 and 100 features. Although Kf10 wins the best performance on a nested subset of 10 features, it performs badly on other nested subsets. Among 10 nested subsets, MSR achieves the best accuracy (75.47%) on subsets of 30 features and 100 features. For the Leukemia dataset, MSR wins the best performance on 4 nested subsets, which contain 20, 30, 40, and 60 features. Although 20/80 wins the best performance on a nested subset of 10 features, it also performs badly on other nested subsets. Although 20xho wins the best accuracy (91.9%), this accuracy is achieved on the subset of 100 features, which is much more than the number of features that MSR achieved the best accuracy (90.29%). Generally speaking, the optimal subset of features gives the best generalization performance with the number of features as small as possible. In this case, 20xho gives the best accuracy (91.91%) on a subset with 100 features, and MSR gives the best accuracy (90.29%) on a subset with 30 features. If we take all factors into consideration, the best algorithm is MSR. The similar conclusions can be concluded on other datasets.

To further investigate the relationship between the number of features and the accuracy, we perform an extensive experiment of these algorithms with features numbering in the range of [1]. The results of 100 points are shown in Fig. 1.

To meet the requirement of the optimal subset of features, we observe the accuracy with features numbering in the range of [1]. In Fig. 1, MSR wins the highest accuracy with features in the range of [1] on all datasets. For example, MSR achieves the accuracy of 75.47% with 26 features on Colon dataset. These observations demonstrate that the MSR is the best algorithm in terms of the generalization performance.

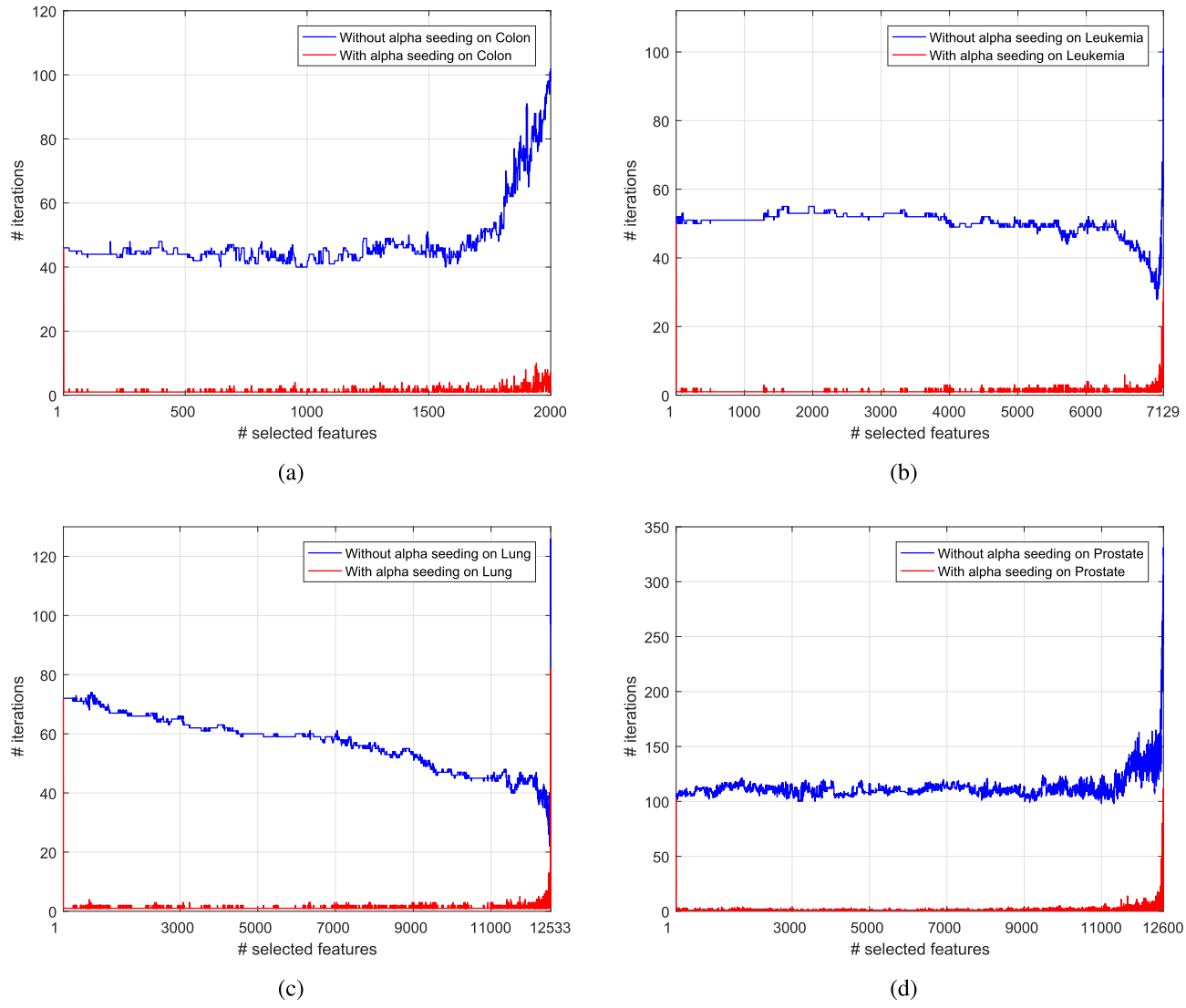

Fig. 2. Iteration times comparison between with alpha seeding and without alpha seeding in the linear SVM-RFE. (a) Colon Tumor; (b) Leukemia; (c) Lung Cancer; (d) Prostate Cancer.

5.3. Computational cost

In this section, we compare the computational cost of MSR with alpha seeding and without alpha seeding. The computational cost of MSR mainly involves computational time of the linear SVM-RFE and feature ranking list measurement. The computational cost of compared algorithms are measured through internal clock statements of MATLAB.

For notational convenience, we denote the computational time of a linear SVM-RFE without alpha seeding and with alpha seeding by RFE-O and RFE-AS, respectively. We also denote the computational time of feature ranking list measurement without alpha seeding and with alpha seeding by MEA-O and MEA-AS, respectively. The experimental results of compared algorithms are shown in Table 6.

As can be seen in Table 6, RFE-AS has clear computational advantage over RFE-O. RFE-AS is about 7 times faster than RFE-O on Colon, and 3 times faster than on Leukemia. However, MEA-AS does not offer much improvement in terms of the computational cost. This is because the computational cost of feature ranking list measurement when combined with a linear SVM classifier involves linear kernel calculation time and QP solving time. Alpha seeding only brings QP solving time saving, rather than kernel calculation time saving. If the kernel calculation time is much larger than the QP solving time, alpha seeding can offer little effort on reducing the computational cost of training a set of successive SVM classifiers.

To further study the computational cost saving of alpha seeding on the QP solving time, we perform an experiment to compare the computational cost of SVM with alpha seeding and without alpha seeding. Take linear SVM-RFE for example, we compare the QP solving time and the number of iterations of successive SVM classifiers in the linear SVM-RFE. Results are shown in Figs. 2 and 3.

As can be seen in Figs. 2 and 3, both the number of iterations and the QP solving time of SVM are greatly reduced by alpha seeding on all datasets. On Colon, the total number of iterations of 2000 SVMs without alpha seeding is 95798, and the total number of iterations of these SVMs with alpha seeding is 2695, which provides 35 times saving. In addition, the total computational time without alpha seeding is 17.30s, and the total computational time with alpha seeding is 0.64s, which provides 27 times saving. These results confirm the superiority of the alpha seeding in the training of successive SVM classifiers.

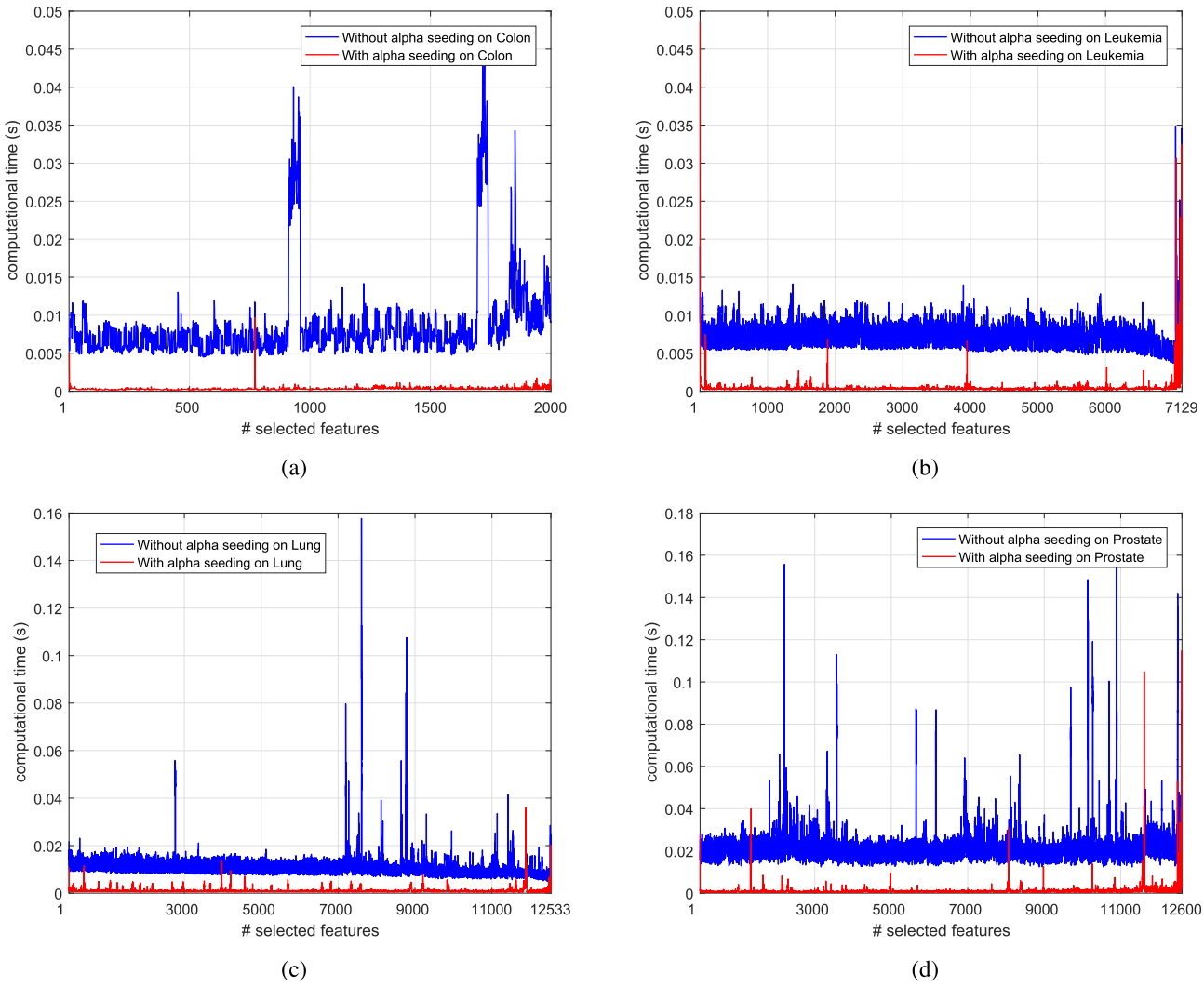

Fig. 3. Computational time comparison between with alpha seeding and without alpha seeding in the linear SVM-RFE. (a) Colon Tumor; (b) Leukemia; (c) Lung Cancer; (d) Prostate Cancer.

6. Conclusions

This study presents a new model selection algorithm for a linear SVM-RFE. The algorithm uses an approximation method to evaluate the generalization performance of a linear SVM-RFE, and a criterion of tuning the penalty parameter C. The computational complexity of the proposed algorithm is discussed. Several alpha seeding strategies are proposed to reduce the computational complexity of the proposed algorithm. The performance of the proposed algorithm is tested against several state-of-the-art algorithms on benchmark bioinformatics datasets. SVM-RFE with the proposed algorithm is effective for early tumor detection and cancer discovery as it leads to a more reliable cancer diagnosis or prognosis and better clinical treatment. The suggested model selection algorithm can be extended to other RFE algorithms, such as least square SVM-RFE, random forest RFE (RF-RFE).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (62002156,61973151), and the Natural Science Foundation of Jiangsu (BK20191406, BE2020001-1), and the Key Project of Philosophy and Social Science Research in Colleges and Universities in Jiangsu Province (2021SJA0265).

References

[1] M. Ali, T. Aittokallio, Machine learning and feature selection for drug response prediction in precision oncology applications, Biophys. Rev. 11 (2019) 31–39.
[2] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, A.J. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proc. Nat. Acad. Sci. 96 (1999) 6745–6750.
[3] J.C. Ang, A. Mirzal, H. Haron, H.N.A. Hamed, Supervised, unsupervised, and semi-supervised feature selection: A review on gene selection, IEEE/ACM Trans. Comput. Biol. Bioinf. 13 (2016) 971–989, https://doi.org/10.1109/TCBB.2015.2478454.
[4] N.E. Ayat, M. Cheriet, C.Y. Suen, Automatic model selection for the optimization of svm kernels, Pattern Recogn. 38 (2005) 1733–1745.
[5] Y. Bao, Z. Hu, T. Xiong, A pso and pattern search based memetic algorithm for svms parameters optimization, Neurocomputing 117 (2013) 98–106.
[6] fish 0,punct]"> A.L. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artif. Intell. 97 (1997) 245–271.
[7] C. Campbell, N. Cristianini, J. Shawe-Taylor, Dynamically adapting kernels in support vector machines, Adv. Neural Inform. Process. Syst. 11 (1999) 204–210.

[8] B.Y. Chu, C.H. Ho, C.H. Tsai, C.Y. Lin, C.J. Lin, Warm start for parameter selection of linear classifiers, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 149–158.

[9] D. DeCoste, K. Wagstaff, Alpha seeding for support vector machines, in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000, pp. 345–349.

[10] N.D. Dees, Q. Zhang, C. Kandoth, M.C. Wendl, W. Schierding, D.C. Koboldt, T. B. Mooney, M.B. Callaway, D. Dooling, E.R. Mardis, et al., Music: identifying mutational significance in cancer genomes, Genome Res. 22 (2012) 1589–1598.

[11] X. Ding, S. Jin, M. Lei, F. Yang, A predictor-corrector affine scaling method to train optimized extreme learning machine, J. Franklin Inst. 359 (2022) 1713–1731.

[12] X. Ding, J. Liu, F. Yang, J. Cao, Random compact gaussian kernel: Application to elm classification and regression, Knowl.-Based Syst. 217 (2021) 106848.

[13] X. Ding, J. Liu, F. Yang, J. Cao, Random radial basis function kernel-based support vector machine, J. Franklin Inst. 358 (2021) 10121–10140.

[14] X. Ding, F. Yang, S. Jin, J. Cao, An efficient alpha seeding method for optimized extreme learning machine-based feature selection algorithm, Comput. Biol. Med. (2021) 104505.

[15] X. Ding, F. Yang, Y. Zhong, J. Cao, A novel recursive gene selection method based on least square kernel extreme learning machine, IEEE/ACM Trans. Comput. Biol. Bioinf. (2021).

[16] X.J. Ding, B.F. Chang, Active set strategy of optimized extreme learning machine, Chin. Sci. Bull. (2014) 59, https://doi.org/10.1007/s11434-014-0512-2.

[17] E. Duarte, J. Wainer, Empirical comparison of cross-validation and internal metrics for tuning svm hyperparameters, Pattern Recogn. Lett. 88 (2017) 6–11, https://doi.org/10.1016/j.patrec.2017.01.007. URL: http://www.sciencedirect.com/science/article/pii/S0167865517300077.

[18] A. Emdadi, C. Eslahchi, Auto-hmm-lmf: feature selection based method for prediction of drug response via autoencoder and hidden markov model, BMC Bioinformat. 22 (2021) 1–22.

[19] M.J. Gangeh, H. Zarkoob, A. Ghodsi, Fast and scalable feature selection for gene expression data using hilbert-schmidt independence criterion, IEEE/ACM Trans. Comput. Biol. Bioinformat. 14 (2017) 167–181.

[20] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, et al., Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.

[21] G.J. Gordon, R.V. Jensen, L.L. Hsiao, S.R. Gullans, J.E. Blumenstock, S. Ramaswamy, W.G. Richards, D.J. Sugarbaker, R. Bueno, Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma, Cancer Res. 62 (2002) 4963–4967.

[22] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Mach. Learn. 46 (2002) 389–422.

[23] T. Joachims, The maximum-margin approach to learning text classifiers: methods theory, and algorithms, Ausgezeichnete Informatikdissertationen 2001 (2003).

[24] J.H. Lee, X.M. Zhao, I. Yoon, J.Y. Lee, N.H. Kwon, Y.Y. Wang, K.M. Lee, M.J. Lee, J. Kim, H.G. Moon, et al., Integrative analysis of mutational and transcriptional profiles reveals driver mutations of metastatic breast cancers, Cell Discov. 2 (2016) 1–14.

[25] M.M. Lee, S.S. Keerthi, C.J. Ong, D. DeCoste, An efficient method for computing leave-one-out error in support vector machines with gaussian kernels, IEEE Trans. Neural Networks 15 (2004) 750–757.

[26] C. Li, X. An, R. Li, A chaos embedded gsa-svm hybrid system for classification, Neural Comput. Appl. 26 (2015) 713–721.

[27] S. Liu, S. Xue, J. Wu, C. Zhou, J. Yang, Z. Li, J. Cao, Online active learning for drifting data streams, IEEE Trans. Neural Netw. Learn. Syst. 1–15 (2021) 1–15, https://doi.org/10.1109/TNNLS.2021.3091681.

[28] Y. Liu, S. Liao, Granularity selection for cross-validation of svm, Inf. Sci. 378 (2017) 475–483, https://doi.org/10.1016/j.ins.2016.06.051. URL: http://www.sciencedirect.com/science/article/pii/S0020025516304807.

[29] Y. Mao, X. Zhou, Z. Yin, D. Pi, Y. Sun, S.T. Wong, Gene selection using gaussian kernel support vector machine based recursive feature elimination with adaptive kernel width strategy, in: International Conference on Rough Sets and Knowledge Technology, Springer, 2006, pp. 799–806.

[30] F. Meric-Bernstam, A. Johnson, V. Holla, A.M. Bailey, L. Brusco, K. Chen, M. Routbort, K.P. Patel, J. Zeng, S. Kopetz, et al., A decision support framework for genomically informed investigational cancer therapy, JNCI: J. Natl. Cancer Inst. 107 (2015) 107.

[31] G. Moore, C. Bergeron, K.P. Bennett, Model selection for primal svm, Mach. Learn. 85 (2011) 175–208.

[32] T.K.B. Mudiyanselage, X. Xiao, Y. Zhang, Y. Pan, Deep fuzzy neural networks for biomarker selection for accurate cancer detection, IEEE Trans. Fuzzy Syst. 28 (2019) 3219–3228.

[33] J. Platt, Sequential minimal optimization: A fast algorithm for training support vector machines, 1998.

[34] Y. Saeys, I. Inza, P. Larranaga, A review of feature selection techniques in bioinformatics, Bioinformatics 23 (2007) 2507–2517.

[35] M.D. Shieh, C.C. Yang, Multiclass svm-rfe for product form feature selection, Expert Syst. Appl. 35 (2008) 531–541.

[36] D. Singh, P.G. Febbo, K. Ross, D.G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A.V. D'Amico, J.P. Richie, E.S. Lander, M. Loda, P.W. Kantoff, T. R. Golub, W.R. Sellers, Gene expression correlates of clinical prostate cancer behavior, Cancer Cell 1 (2002) 203–209, https://doi.org/10.1016/S1535-6108(02)00030-2. URL: http://www.sciencedirect.com/science/article/pii/S1535610802000302.

[37] Y. Sugai, N. Kadoya, S. Tanaka, S. Tanabe, M. Umeda, T. Yamamoto, K. Takeda, S. Dobashi, H. Ohashi, K. Takeda, et al., Impact of feature selection methods and subgroup factors on prognostic analysis with ct-based radiomics in non-small cell lung cancer patients, Radiat. Oncol. 16 (2021) 1–12.

[38] J. Sun, C. Zheng, X. Li, Y. Zhou, Analysis of the distance between two classes for tuning svm hyperparameters, IEEE Trans. Neural Networks 21 (2010) 305–318, https://doi.org/10.1109/TNN.2009.2036999.

[39] V. Vapnik, O. Chapelle, Bounds on error expectation for support vector machines, Neural Comput. 12 (2000) 2013–2036.

[40] S. Varma, R. Simon, Bias in error estimation when using cross-validation for model selection, BMC Bioinformat. 7 (2006) 1–8.

[41] J. Wainer, G. Cawley, Empirical evaluation of resampling procedures for optimising svm hyperparameters, J. Mach. Learn. Res. 18 (2017) 475–509.

[42] A. Wang, N. An, G. Chen, L. Li, G. Alterovitz, Improving pls–rfe based gene selection for microarray data classification, Comput. Biol. Med. 62 (2015) 14–24.

[43] Z. Wen, B. Li, R. Kotagiri, J. Chen, Y. Chen, R. Zhang, Improving efficiency of svm k-fold cross-validation by alpha seeding, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[44] C.H. Wu, G.H. Tzeng, R.H. Lin, A novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression, Expert Syst. Appl. 36 (2009) 4725–4735.

[45] Z. Wu, J. Cao, Y. Wang, Y. Wang, L. Zhang, J. Wu, hpsd: A hybrid pu-learning-based spammer detection model for product reviews, IEEE Trans. Cybernet. 50 (2020) 1595–1606, https://doi.org/10.1109/TCYB.2018.2877161.

[46] F. Yang, K.Z. Mao, Robust feature selection for microarray data based on multicriterion fusion, IEEE/ACM Trans. Comput. Biol. Bioinf. 8 (2011) 1080–1092, https://doi.org/10.1109/TCBB.2010.103.

[47] L. Zhang, X. Zheng, Q. Pang, W. Zhou, Fast gaussian kernel support vector machine recursive feature elimination algorithm, Appl. Intell. (2021) 1–14.