FS_SFS: A Novel Feature Selection Method for Support Vector Machines

Yi Liu and Yuan F. Zheng
Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210
Email:{liuyi,zheng}@ee.eng.ohio-state.edu

Abstract— **This paper presents a novel feature selection method which is named Filtered and Supported Sequential Forward Search (FS_SFS) in the context of Support Vector Machines (SVM). In comparison with conventional wrapper methods employing the sequential forward search (SFS) strategy, it has two important properties that reduce the computation time of SVM training during the feature selection process. First, in stead of utilizing all the training samples to obtain the classifier, FS_SFS, by taking advantage of the existence of support vectors in SVM, dynamically maintains an active data set for each SVM to be trained on. In this way the computational demand of a single SVM training decreases. Secondly, a new criterion, in which discriminant ability of individual features and the correlation between them are both taken into consideration, is proposed to effectively filter out non-essential features before every SFS iteration begins. As a result, the total number of training is significantly reduced. The proposed approach is tested on both synthetic and real data to demonstrate its effectiveness and efficiency.**

Index Terms— **feature selection, sequential forward search (SFS), support vector machines (SVM), FS_SFS.**

I. Introduction

Feature dimensionality reduction is of considerable importance for two primary reasons: reduce the computational complexity and improve the classifier's generalization ability. Feature selection addresses the dimensionality reduction problem by determining a subset of those features which is most essential for classification. Based on the criterion for subset evaluation, feature selection approaches can be grouped into two categories: filter methods and wrapper methods [1]. Acquiring no feedback from classifiers, filter methods estimate the classification performance by some indirect assessment such as distance measures. Wrapper methods, on the contrary, are classifier-dependent. They evaluate the "goodness" of the selected feature subset directly based on the classification accuracy, which would intuitively yield better performance. As a matter of fact, experimental results are in general reported in favor of the wrapper methods [1] [2] even though more computational cost is needed.

As a state-of-art classifier, Support Vector Machines (SVM) has been successfully applied in a variety of areas [3]–[5]. However, given the fact that training just a single SVM would impose a lot of computation when the number of training samples is large, the integration of SVM and wrapper methods, which calls for multiple times of SVM training process, might be computationally infeasible. In this paper we present a expedited wrapper method for SVM which is named Filtered and Supported Sequential Forward Search (FS_SFS). As its name suggests, this new wrapper feature selection method employs sequential forward search strategy (SFS), but it has the following advantages over the conventional wrapper/SFS method:

1) FS_SFS combines the advantages of filter and wrapper methods by introducing a filtering process for each SFS iteration;
2) FS_SFS introduces a new criterion that is computationally simple and considers both discriminant ability of individual features and the correlation between them;
3) FS_SFS improves the efficiency of obtaining a single SVM classifier by dynamically maintaining a small active training set.

The rest of the paper is organized as follows. Section II gives a brief introduction of SVM and Section III explains FS_SFS in detail. Experimental results are given in section IV followed by conclusions and discussions in section V.

II. Support Vector Machines

SVM is a state-of-art learning machine based on the *structural risk minimization* induction principle. Here we only give a very brief review while the detailed description can be found in [6]. Consider N training sample pairs

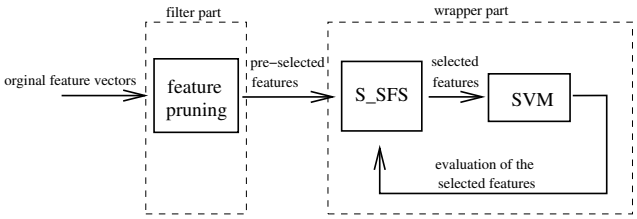
$$\{X(1), Y(1)\}, \{X(2), Y(2)\}, \ldots, \{X(N), Y(N)\},$$

where $X(i)$ is a k-dimensional feature vector representing the i^{th} training sample, and $Y(i) \in \{-1, 1\}$ is the class label of $X(i)$.

A hyperplane in the feature space can be described as the equation $W \cdot X + b = 0$, where $W \in R^k$ and b is a scalar. When the training samples are linearly separable, SVM yields the optimal hyperplane that separates two classes with no training error and maximizes the minimum distance from a point $X(i)$ to the hyperplane by solving the following optimization problem:

$$\begin{aligned} &\text{Minimize :} && f(W) = \tfrac{1}{2}\|W\|^2 \\ &\text{Subject to :} && Y(i)\left(W \cdot X(i) + b\right) \geq 1,\ i = 1, ..., N. \end{aligned} \quad (1)$$

For linearly nonseparable cases, there is no such a hyperplane that is able to classify every training point correctly. However



(a)

Fig. 1. The outline of the proposed method for feature selection for SVM.

the previous idea can be generalized by introducing the concept of *soft margin*. Thus the new optimization problem becomes

$$\begin{aligned} &\text{Minimize :} && f(W,\xi) = \tfrac{1}{2}\|W\|^2 + C\sum_{i=1}^{N}\xi(i) \\ &\text{Subject to :} && Y(i)\big(W\cdot X(i)+b\big) \geq 1-\xi(i), i=1,\ldots N, \end{aligned} \quad (2)$$

where $\xi(i)$, is called a slack variable and related to the soft margin. Both optimization problems (1) and (2) can be solved by introducing the Lagrange multipliers $\alpha(i)$ that reduces them to quadratic programming problems.

In the classification phase, a point $\tilde{X}$ in the feature space is assigned a label $\tilde{Y}$ according to the following equation:

$$\begin{aligned} \tilde{Y} &= \text{sgn}\big[W\cdot\tilde{X}+b\big] \\ &= \text{sgn}[\textstyle\sum_{i=1}^{N}\alpha(i)Y(i)\big(X(i)\cdot\tilde{X}\big)+b]. \end{aligned} \quad (3)$$

III. FS_SFS: Filtered and Supported Sequential Forward Search

A. Algorithm Review of FS_SFS

The outline of the proposed method is shown in Fig. 1. The filtering part in our approach, acting in the generic way similar to a filter method, ranks features without involving the classifier. The features with relatively high ranks are considered as "informative" feature candidates and then re-studied by the wrapper part that further investigates their contributions to a specific classifier. This combinational framework delivers as good performance as the conventional wrapper method but is computationally simpler.

Now with the framework determined, the feature selection problem is reduced to a search problem to find the optimal subset [7]. Many search strategies have been proposed [8]–[10], and we adopt a suboptimal search method called sequential forward search (SFS) [10] algorithm for its simplicity and effectiveness proven in many applications. In the following three subsections, we will explain how FS_SFS works in detail.

B. F_SFS: Filtered_SFS Using a New Criterion

Evidently an effective filtering criterion is needed since it is undesirable if many informative features are discarded by the filtering process. Also the criterion should be simple to avoid excessive computational cost. To address this problem, we propose the following new filtering criterion, which considers both the discriminant ability of individual features as well as the correlation between them. Also it is simple to calculate.

Suppose we have obtained a feature combination $F_s = \{f_{n_1}, f_{n_2}, \ldots, f_{s_d}\}$ and one more feature is to be selected. We evaluate the importance of each individual feature f_i by a score, which is denoted as R_{i,F_s} and is calculated as follows.

1) *discriminant ability of individual features*
The discriminant ability of feature f_i is described by

$$D_i = \frac{|m_1^i - m_2^i|}{std_1^i + std_2^i}, \quad (4)$$

where m_1^i and std_1^i (m_2^i and std_2^i) are the mean and standard deviation of the samples belonging to class 1 (-1) when only feature f_i is considered.

2) *correlation between features*
First we define the correlation coefficient $\rho_{i,j}$ between two features, say f_i and f_j.

$$\rho_{i,j} = \prod_{c=1}^{2}\rho_{i,j}^{(c)} = \prod_{c=1}^{2}\frac{\text{cov}\Big(S_c(f_i), S_c(f_j)\Big)}{\sqrt{\text{var}\Big(S_c(f_i)\Big)\cdot\text{var}\Big(S_c(f_i)\Big)}} \quad (5)$$

where $S_c(f_i) = \{x_{f_i}(l)|Y(l)=c\}$ is the vectors that represented by feature f_i and labeled as class c.
Then based on $\rho_{i,j}$, we define the correlation coefficient between f_i and F_s as

$$\rho_{i,F_s} = \max_{f_j\in F_s}|\rho_{i,j}|. \quad (6)$$

It is desirable to select the features that can individually separate the classes well but has small correlation with the feature set that has been obtained. Thus the final score assigned to f_i is defined as:

$$R_{i,F_s} = \frac{D_i}{\max\{D_l\}} - |\rho_{i,F_s}|, \quad (7)$$

where D_i is normalized such that it is in the same range as $|\rho_{i,F_s}|$.

C. S_SFS: Supported_SFS in the Context of SVM

In SVM there is an special group of training samples named "support vectors", whose corresponding coefficients $\alpha(i)$ in Eq. (3) are non-zeros. In other words, samples other than support vectors have no contribution to determining the decision boundary. Since usually the number of support vectors is relatively small, we could train SVM just by using the support vectors. Following this idea, we propose the supported SFS algorithm, which dynamically maintains an *active training set* as estimated candidates of the support vectors, and trains SVM using this reduced subset rather than the whole original training set. In this way, we are able to find the boundary with less computational cost.

The procedure of S_SFS is described as follows. The first step is to select the best single feature. To do so, we train SVM k times, each of which uses all the training pairs available but only considers the individual feature f_i. Mathematically the initial feature combination set is $F_1^i = f_i, f_i \in F$, and the active training set is $V_1^i = \{1, 2, \ldots, N\}$.

Although in this step every training pair in S is evolved in this initial training task, the computational complexity is not

high because the input vector is just one-dimensional. After the training, each single-feature combination F_1^i is associated with a margin value M_1^i and a group of support vectors v_i. The feature that yields the smallest margin

$$j = \underset{i\in\{1,2,\ldots,N\}}{\arg\min} \quad M_1^i \tag{8}$$

is then chosen as the best single feature. Thus we obtain the initial feature combination $F_1 = \{f_j\}$ and its active training set $V_1 = \{v_j\}$ for the next step.

At step n, we have already obtained the feature combination F_n that contains n features, and the active training set V_n. To choose one more feature into the feature combination set, we add each remaining feature f_i one by one and construct the corresponding active training set for every new feature combination as follows:

$$\begin{cases} F_{n+1}^i = F_n \cup \{f_i\}, \text{ for } f_i \in F_n^{av}, \\ V_{n+1}^i = V_n \cup \{v_i\}. \end{cases} \tag{9}$$

where $F_n^{av} = \{f_r \mid f_r \in F \text{ and } f_r \notin F_n\}$ is the collection of the available features to be selected from.

For each F_{n+1}^i we train SVM using the samples in V_{n+1}^i. The resulting margin and the collection of the support vectors are denoted as M_{n+1}^i and SV_{n+1}^i, respectively. Then the feature f_j that yields the combination with the least margin as

$$j = \underset{f_i\in F_n^{av}}{\arg\min} \; M_{n+1}^i \tag{10}$$

is selected, and accordingly the new feature combination F_{n+1} and new active training set V_{n+1} are obtained as follows:

$$\begin{cases} F_{n+1} = F_{n+1}^j, \\ V_{n+1} = SV_{n+1}^j. \end{cases} \tag{11}$$

The SFS process continues until no significant margin reduction is found or the desired number of features is obtained.

D. FS_SFS: the Integration of F_SFS and S_SFS

The integration of F_SFS and S_SFS is quite straightforward for which the basic idea is discarding the features with low scores that have been computed according to Eq. (7) so as to reduce the number of features S_SFS has to evaluate. Again suppose we are at step n of SFS with F_n and V_n available, and FS_SFS works as follows:

1) calculate the score R_{i,F_n} for each remaining feature f_i;
2) select K_n highest scored features to construct F_n^{av};
3) determine the next feature to be added using Eq. (9) and Eq. (10);
4) update the active training set using Eq. (11).

K_n here is the tuning parameter to balance between the performance and the algorithm complexity. In our experiments, K_n is set to $\lfloor \frac{|F_n|}{2} \rfloor$ such that half of the available features are discarded at every SFS iteration step.

IV. Experimental Results

In the experiments, the proposed feature selection method is applied to both synthetic and real-world data sets. For all the experiments, the SVM optimization is achieved by using SVMTorch [11].

A. Results on Synthetic Data

Three series of experiments are carried out on the synthetic data sets, and for each experiment we sample N vectors $X = (x_1, x_2, \ldots, x_k)$ from two classes (class 1 or class -1) in a k-dimensional data space. The components x_i are *independent* Gaussian variables whose distributions are designed as:

$$p(x_i) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_i} exp(\frac{x_i - 1}{2\sigma_i^2}), & \text{if } X \text{ belongs to class 1;} \\ \frac{1}{\sqrt{2\pi}\sigma_i} exp(\frac{x_i + 1}{2\sigma_i^2}), & \text{if } X \text{ belongs to class -1,} \end{cases} \tag{12}$$

where $\sigma_i = 0.5 \cdot 2^{(i-1)}$ and $i = 1, 2, \ldots, k$.

The first experiment is a 2-D case where $k = 2$ and $N = 100$. Fig. 2 shows how the active training set changes when features are added one by one into the candidate feature set F. FS_SFS is also tested in a 3-D case where $k = 3$ and $N = 100$. In both 2-D and 3-D scenarios, we observe that with our experiment setting FS_SFS and the conventional SVS methods generate exactly the same support vectors .

In the third experiment, we test FS_SFS in a 10-dimensional case where $k = 10$ and $N = 250$. According to Eq. (12), if $i < j$ the variance of feature x_i is larger than that of x_j, and therefore x_i has more discriminant ability. For that reason, we expect x_i to be selected before x_j. For display purpose, we assign a feature x_i a point as $11 - \text{pos}(x_i)$, where $\text{pos}(x_i)$ is the order of x_i selected. For example, if x_i is the number one selected feature component, its point would be 10. Fig. 3(a) gives the ideal point of x_i. Fig. 3(b) and Fig. 3(c) show the actual points of the features, which are averaged over 100 trials, when SFS and FS_SFS are applied, respectively. As one can see, FS_SFS is able to achieve similar results of SFS with lower computational cost.

B. Results on Real-World Data

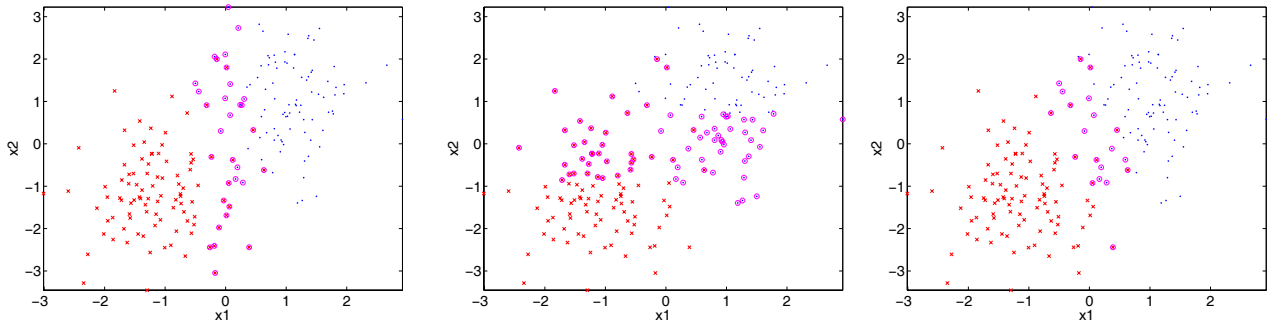
The proposed algorithm is applied to four real-world data sets obtained from the widely-used UCI (University of California, Irvine) repository of machine learning [12]. These data sets are:

1) the BUPA Liver Disorders data set (BUPA Liver) which contains 354 instances with 6 features;
2) the Wisconsin Breast Cancer data set (BCW) which contains 683 instances with 9 feature;
3) the data of letter 'A' and 'B' from Letter Image Recognition data set (A-B-letter) which contains 1555 instances with 16 feature;
4) the Johns Hopkins University Ionosphere data set (Ionosphere) which contains 351 instances with 34 feature.

For each data set we randomly set aside 20% instances as the testing samples, and the rest as the training samples. The results are listed in Table I. As one can see, FS_SFS improves the efficiency of SFS without sacrificing the accuracy of either the selection or the classification.

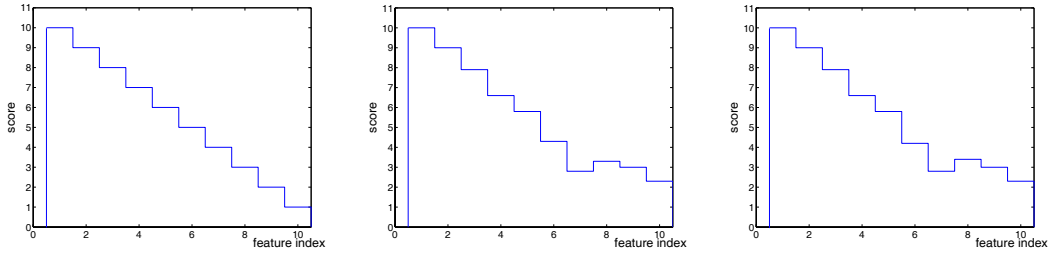
V. Conclusions

In this paper, we present a novel feature selection method for SVM. By introducing a feature pruning process, we filter out "uninformative" features to reduce the required number of



(a) (b) (c)

Fig. 2. The active training set (circled) maintained by S_SFS of a 2-D case. (a) v_1, which is the support vectors obtained by considering only feature x_1. (b) v_2, which is the support vectors obtained by considering only feature x_2. (c) The support vectors obtained by training SVM on $V = v_1 \bigcup v_2$.



(a) (b) (c)

Fig. 3. The points of feature components. (a) The ideal points. (b) The points obtained by using SFS. (b) The points obtained by using FS_SFS.

TABLE I

COMPARISON OF CLASSIFICATION ACCURACY AND RUN TIME BETWEEN FS_SFS AND SFS OVER 10 TRIALS.

	number of features		classification accuracy (%)				Run Time (seconds)		
			training		testing				
	available	selected	FS_SFS	SFS	FS_SFS	SFS	FS_SFS	SFS	FS_SFS/SFS
BUPA Liver	6	4.6	78.7%	78.5%	70.2%	70.7%	4.31	6.08	71%
BCW	9	5.5	97.4%	97.4%	96.3%	95.4%	10.61	13.31	79.7%
A-B Letter	16	6.2	99.95%	100%	99.7%	99.8%	48.8	65.0	72%
Ionosphere	34	10.0	98.9%	99.3%	92.0%	90.6%	81.5	118.9	68.5%

training. We also develop a new feature ranking criterion, in which not only the class separability of individual features but also the correlation between features are taken into account, to make the pruning process more effective. Furthermore, during the SFS searching process, an active training set is maintained as the estimated candidates of the support vectors. Whenever SVM has to be trained, it is done over the reduced training set. In this way, the number of samples participating in a single optimization procedure decreases and therefore the training process is expedited. We test the proposed method on both artificial and real-world data sets, and the experimental results demonstrate its effectiveness and efficiency.

REFERENCES

[1] R. Kohavi, and G.H. John, "Wrappers for Feature Subset Selection", Artificial Intelligence, vol. 97, pp. 273-324. 1997.

[2] H. Watanabe, T. Yamaguchi, and S. Katagiri, "Discriminative Metric Design for Robust Pattern Recognition", *IEEE Trans. on Signal Processing*, vol. 45, no. 11, pp. 2655-2662, Nov. 1997.

[3] M. Pontil. and A. Verri, "Support Vector Machines for 3D Object Recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 6, pp. 637-646, June 1998.

[4] S. Tong and E. Change, "Support Vector Machine Active Learning for Image Retrieval", *Proc. ACM International Conference on Multimedia*, pp. 107-118, Oct. 2001.

[5] T. Joachims, "Transductive Inference for Text Classification Using Support Vector Machines", *Proc. of 16th International Conference on Machine Learning*, pp. 200-209, 1999.

[6] C. Cortes and Vladimir N. Vapnik, "Support Vector Networks", *Machine Learning*, vol. 20, no. 3, pp. 273-297, 1995.

[7] K.Z. Mao, "Fast Orthogonal Forward Selection Algorithm for Feature Subset Selection", *IEEE Trans. on Neural Networks*, vol. 13, no. 5, pp. 1218-1224, Sept. 2002.

[8] P.M. Narendra, and K. Fukunaga, "A Branch and Bound Algorithm for Feature Subset Selection", *IEEE Trans. on Computers*, vol. 26, pp. 917-922, 1977.

[9] M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, and A.K. Jain, "Dimensionality Reduction Using Genetic Algorithms", *IEEE Trans. on Evolutional Computation*, vol. 4, no. 2, pp. 164-171, July 2001.

[10] T. Marill, and D.M. Green, "On the Effectiveness of Recep-tors in Recognition Systems", *IEEE Trans. on Information Theory*, vol. 9, pp. 11-17, 1963.

[11] R. Collobert, and S. Bengio, "SVMTorch: Support Vector Machines for Large-Scale Regression Problems", *Journal of Machine Learning Research*, vol. 1, pp 143-160, 2001.

[12] C.L. Blake, and C.J. Merz, "UCI Repository of Machine Learning Databases", Department of Information and Computer Science, University of California, Irvine, CA, http://www.ics.uci.edu/ mlearn/MLRepository.html, 1998.