# Training Hard-Margin Support Vector Machines Using Greedy Stagewise Algorithm

Liefeng Bo, *Member, IEEE*, Ling Wang, and Licheng Jiao, *Senior Member, IEEE*

***Abstract*—Hard-margin support vector machines (HM-SVMs) suffer from getting overfitting in the presence of noise. Soft-margin SVMs deal with this problem by introducing a regularization term and obtain a state-of-the-art performance. However, this disposal leads to a relatively high computational cost. In this paper, an alternative method, greedy stagewise algorithm for SVMs, named GS-SVMs, is presented to cope with the overfitting of HM-SVMs without employing the regularization term. The most attractive property of GS-SVMs is that its computational complexity in the worst case only scales quadratically with the size of training samples. Experiments on the large data sets with up to 400 000 training samples demonstrate that GS-SVMs can be faster than LIBSVM 2.83 without sacrificing the accuracy. Finally, we employ statistical learning theory to analyze the empirical results, which shows that the success of GS-SVMs lies in that its early stopping rule can act as an implicit regularization term.**

***Index Terms*—Classification, greedy stagewise algorithm, support vector machines (SVMs), Vapnik–Chervonenkis (VC) dimension.**

## I. Introduction

HARD-MARGIN support vector machines (HM-SVMs) have a risk of getting overfitting in the presence of noise [1], [2]. To deal with this problem, soft-margin SVMs [3], [4] introduce the regularization parameter that allows some training error to obtain large margin. This is a highly effective mechanism for avoiding overfitting, which leads to good generalization performance. Though very successful, we can identify shortcomings of soft-margin SVMs.

1) The training procedure of soft-margin SVMs amounts to solving a constrained quadratic programming. Although the training problem is, in principle, solvable, in practice it is intractable by the classical optimization techniques, e.g., interior point method, because their computational complexity usually scales cubically with the size of training samples.
2) The regularization parameter depends on the task at hand; hence, there is no foolproof method for determining it before training. Usually, we have to resort to a cross-validation procedure that is wasteful in computation [5].

In the past few years, many fast iterative algorithms were presented to cope with problem 1). Chunking algorithm [6] splits the variables into inactive and active sets (also named working set). At first, an arbitrary subset of the variables is selected as the working set. After a general optimization algorithm, e.g., interior point method is applied to the subset, the support vectors in the working set are reserved and the rest are replaced with the variables that violate Karush–Kuhn–Tucker (KKT) conditions. However, this algorithm still is inapplicable in case the number of support vectors is very large due to high memory requirement. Joachims [7] identified this shortage and developed an efficient decomposition scheme, named $\text{SVM}^{\text{light}}$. The key idea of $\text{SVM}^{\text{light}}$ is to find a feasible direction of steepest descend, in which the number of nonzero elements is set to be a small constant. Platt [8] took the decomposition idea to an extreme where the size of the working set of sequential minimal optimization (SMO) algorithm is set to be two and hence an analytical solution for subproblem is obtained. Keerthi *et al.* [9] and Shevade *et al.* [10] further improved the performance of SMO by introducing the maximal violating pair working set selection. Hastie *et al.* [11] derived an algorithm that can fit the entire path of SVM solutions for every value of the regularization parameter. Some other examples include Kernel–Adatron [12], SimpleSVM [13], SVMTorch [14], and so on.

Recently, there have been many attempts to approximately train SVMs. Collobert *et al.* [15] proposed a parallel mixture of SVMs. Dong *et al.* [16] introduced a parallel optimization step to quickly remove most of the nonsupport vectors for speeding up SVMs. Bakir *et al.* [17] selectively removed training samples using probabilistic estimates related to editing algorithms. Bordes *et al.* [18] presented an online algorithm to compute an approximation solution of SVMs. Tsang *et al.* [19] showed that many kernel methods can be equivalently formulated as minimum enclosing ball problems in computational geometry and presented core vector machine (CVM) to compute the approximate solution of SVMs. Keerthi *et al.* [20] built sparse SVMs using a matching pursuit-like algorithm. These algorithms proved to be effective and boosted the development of large scale SVMs.

Based on a preliminary work [21], a greedy stagewise algorithm for approximately training SVMs (GS-SVMs) is presented to deal with the overfitting of HM-SVMs. Instead of employing the regularization term, GS-SVMs attempt to control the complexity of the hypothesis space by themselves. They iteratively build the decision function by adding one kernel function at one time. At each iteration, GS-SVMs determine the index and the weight of the new kernel function to be included

Manuscript received March 1, 2007; revised October 18, 2007; accepted February 13, 2008. First published July 15, 2008; last published August 6, 2008 (projected). This work was supported by the National Natural Science Foundation of China under Grant 60372050 and the National Defense Preresearch Foundation of China under Grant A1420060172.

The authors are with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, the Institute of Intelligent Information Processing, Xidian University, Xi'an, Shaanxi, 710071 P. R. China (e-mail: blf0218@ 163.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2008.2000576

by an optimization problem in two variables, whose solution can be obtained in closed form. This procedure is repeated until the loss function stops decreasing. The proposed algorithm possesses the two following attractive properties.

1) The computational complexity of GS-SVMs is $O(n\ell)$, where $n$ is the number of support vectors and $\ell$ is the number of training samples. Even in the worst case of all the training samples being the support vectors, the computational complexity is only $O\left(\ell^2\right)$.
2) No extra regularization parameter is required.

Extensive empirical comparisons validate the efficiency and effectiveness of GS-SVMs. Moreover, we employ statistical learning theory to analyze the empirical results, which shows that the success of GS-SVMs lies in that their early stopping rule can act as an implicit regularization term.

This paper is organized as follows. In Section II, a brief introduction of SVMs is given. The reason that the dual of HM-SVMs can be regarded as a loss function is interpreted in Section III. GS-SVMs is detailed in Section IV. Experiments which demonstrate the speed and generalization performance of GS-SVMs are given in Section V. In Section VI, we explore the reason for the success of GS-SVMs. In Section VII, the contributions of this paper are summarized and the further research direction is indicated.

## II. Support Vector Machines

In this section, we briefly introduce SVMs. For more details, the interested reader can refer to [22] and [23]. In classification, we are given a set of training samples $\{\mathbf{x}_i, y_i\}_{i=1}^{\ell}$, where $\mathbf{x}_i$ is the input sample defined on $R^d$, $y_i$ is the corresponding output, and $\ell$ is the number of training samples. The aim is to determine an approximation function $f(\mathbf{x})$ of the target function $f^*(\mathbf{x})$, which best represents the relationship between the inputs and the outputs. In the feature space, SVMs model takes the form $f(\mathbf{x}) = \mathbf{w}^T\Phi(\mathbf{x})$ where the nonlinear mapping $\Phi(\mathbf{x})$ maps the input data into a higher dimensional feature space whose dimension can be infinite. We have also dropped the threshold $b$ for the sake of simplicity. The generalization performance of SVMs usually is not affected by this drop in most cases (one should be cautious with very unbalanced data sets where the threshold can be helpful). To obtain a classifier, HM-SVMs solve the following optimization problem:

$$\begin{aligned} &\min \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{s.t.} \quad y_i\left(<\mathbf{w}, \Phi(\mathbf{x}_i)>\right) \geq 1, \qquad i = 1, \ldots, \ell. \end{aligned} \tag{1}$$

Its Wolfe dual is

$$\begin{aligned} &\min \left( \frac{1}{2}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j y_i y_j \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) - \sum_{i=1}^{\ell}\alpha_i \right) \\ &\text{s.t.} \quad 0 \leq \alpha_i, \qquad i = 1, \ldots, \ell. \end{aligned} \tag{2}$$

According to Mercer's theory [24], any positive–definite kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ can be expressed as the inner product of two vectors in some feature space, and therefore, it can be used in SVMs. Replacing $\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$ with $K(\mathbf{x}_i, \mathbf{x}_j)$, we get

$$\begin{aligned} &\min \left( \frac{1}{2}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{\ell}\alpha_i \right) \\ &\text{s.t.} \quad 0 \leq \alpha_i, \qquad i = 1, \ldots, \ell. \end{aligned} \tag{3}$$

To deal with the nonseparable case, one often uses soft-margin SVMs

$$\begin{aligned} &\min \left( \frac{1}{2}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{\ell}\alpha_i \right) \\ &\text{s.t.} \quad 0 \leq \alpha_i \leq C, \qquad i = 1, \ldots, \ell. \end{aligned} \tag{4}$$

For a new sample $\mathbf{x}$, we can predict its label by

$$f(\mathbf{x}) = \operatorname{sgn}\left( \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) \right) \tag{5}$$

where $\alpha_i$ is the solution of (4).

## III. RKHS Norm View for SVMs

The key conclusion in this section is that the Wolfe dual of HM-SVMs can be regarded as the loss function induced by a reproducing kernel Hilbert space (RKHS) norm, which is the basis of developing greedy approximation algorithms. Similar conclusion about support vector regression is reported by Girosi [25].

*Theorem 1 [24]:* Let $X \subset \mathbb{R}^d$, a real symmetric function $K(\mathbf{x}, \mathbf{y})$, $\mathbf{x}, \mathbf{y} \in X$ be positive–definite symmetric if and only if for every set of real numbers $\{\alpha_1, \alpha_2, \ldots, \alpha_\ell\}$ and every set of vectors $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\ell\}$, we have $\sum_{i,j=1}^{\ell} \alpha_i\alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$.

*Definition 1 [24]:* A Hilbert space $H$ of function $f: X \to \mathbb{R}$, $X \neq \varnothing$, is called an RKHS with dot product $\langle \bullet, \bullet \rangle_H$ and norm $\|f\|_H = \sqrt{\langle f, f\rangle_H}$ if there exists a function $K: X \times X \to \mathbb{R}$ satisfying $< f(\bullet), K(\bullet, \mathbf{x}) >_H = f(\mathbf{x})$ and spanning $H$, i.e., $H = \overline{\operatorname{span}\{K(\bullet, \mathbf{x}), \mathbf{x} \in X\}}$.

We call $\|f\|_H = \sqrt{\langle f, f\rangle_H}$, where $f \in H$, reproducing norm. RKHS $H_K$ induced by $K(\mathbf{x}, \mathbf{y})$ satisfies the following three properties:

1) $K(\bullet, \mathbf{x}) \in H_K$ where $\mathbf{x} \in X$;
2) $\sum_{i=1}^{\ell} \alpha_i K(\bullet, \mathbf{x}_i) \in H_K$, where $\alpha_i$ and $\ell$ are finite;
3) for $f(\bullet) \in H_K$, $\mathbf{x} \in X$, $< f(\bullet), K(\bullet, \mathbf{x}) >_H = f(\mathbf{x})$, where $\langle \bullet, \bullet \rangle_H$ is the inner product of RKHS. In particular, $< K(\bullet, \mathbf{x}_i), K(\bullet, \mathbf{x}_j) >_H = K(\mathbf{x}_i, \mathbf{x}_j)$.

According to the property 2), we can derive that the decision function of SVMs, $f(\mathbf{x})$, belongs to RKHS $H_K$. We assume that the unknown target function $f^*(\mathbf{x})$ belongs to RKHS $H_K$. Measuring the distance by RKHS norm between the target function $f^*(\mathbf{x})$ and the approximation function $f(\mathbf{x})$, we have the following loss function:

$$\begin{aligned} &\frac{1}{2}\left\| f^*(\mathbf{x}) - \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) \right\|_H^2 \\ &\text{s.t.} \quad \alpha_i \geq 0, \qquad i = 1, 2, \ldots, \ell \end{aligned} \tag{6}$$

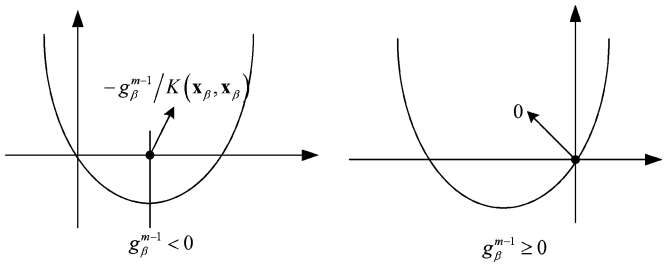


Fig. 1. Visualization of the solution.

where $\|\bullet\|$ is RKHS norm. Equation (6) can be expanded as

$$\frac{1}{2}\|f^*(\mathbf{x})\|_H^2-\sum_{i=1}^{\ell}\alpha_i y_i\left\langle f^*(\mathbf{x}),K(\mathbf{x},\mathbf{x}_i)\right\rangle_H + \frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j y_i y_j\left\langle K(\mathbf{x},\mathbf{x}_i),K(\mathbf{x},\mathbf{x}_j)\right\rangle_H$$
$$\text{s.t.}\quad \alpha_i\geq 0,\qquad i=1,2,\ldots,\ell. \tag{7}$$

Using the reproducing property 3) of kernel function, we can transform (7) into

$$\frac{1}{2}\|f^*(\mathbf{x})\|_H^2-\sum_{i=1}^{\ell}\alpha_i y_i f^*(\mathbf{x}_i)+\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)$$
$$\text{s.t.}\quad \alpha_i\geq 0,\qquad i=1,\ldots,\ell. \tag{8}$$

Since $f^*(\mathbf{x}_i)$ is the output of target function on the point $\mathbf{x}_i$, it is reasonable to estimate it by $y_i$ (for noiseless data, $f^*(\mathbf{x}_i)=y_i$). Thus, we have

$$\frac{1}{2}\|f^*(\mathbf{x})\|_H^2-\sum_{i=1}^{\ell}\alpha_i+\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)$$
$$\text{s.t.}\quad \alpha_i\geq 0,\qquad i=1,\ldots,\ell. \tag{9}$$

Eliminating the constant term, we can estimate $\alpha_i,\ i=1,\ldots,\ell$ by

$$\min\left(\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)-\sum_{i=1}^{\ell}\alpha_i\right)$$
$$\text{s.t.}\quad 0\leq\alpha_i,\qquad i=1,\ldots,\ell. \tag{10}$$

It is easily checked that (10) completely amounts to (5), which enlightens us to take the Wolfe dual of SVMs as the loss function induced by RKHS norm. If we further constrain $\alpha_i,\ i=1,\ldots,\ell$ smaller than $C$, we can obtain soft-margin SVMs.

## IV. Greedy Stagewise Algorithm for SVMs

Though (10) is, in principle, solvable by the classical optimization techniques, in practice, it suffers from two serious problems: 1) its computational complexity usually scales cubically with the size of training samples; and 2) there often is a risk of getting overfitting due to no regularization term.

In this section, we will deal with the aforementioned two problems by GS-SVMs which attempts to fast approximate (10) while avoiding the overfitting. The dictionary $D=\{k(\mathbf{x},\mathbf{x}_i)\,|i=1,2,\ldots,\ell\}$ used by GS-SVMs is a set of the kernel functions centered on the training samples. GS-SVMs iteratively build the decision function by adding one kernel function at a time. At each iteration, GS-SVMs determine the index and the weight of the next kernel function to be included by an optimization problem in two variables. This procedure is repeated until the loss function (10) stops decreasing.

There are many efforts for greedy learning algorithms. In general, the existing methods can be roughly classified into two groups. The first group is called greedy stepwise approach that readjusts the weights of the previously entered basis functions when a new basis function is added. The typical algorithms include orthogonal least squares learning algorithms [26], kernel matching pursuit (backfitting and prefitting version) [27], fast sparse approximation for least square SVMs [28], and so on. The second group is called greedy stagewise approach that fixes the weights of the previously entered basis functions when a new basis function is added. The typical algorithms include matching pursuit [29], AdaBoost [30], LogitBoost [31], Doom II [32], gradient boosting [33], leveraged vector machines [34], and so on.

Our algorithm can be classified into the second group. The most important difference among the algorithms in the second group is the loss function they optimize. Matching pursuit uses a squared loss function; AdaBoost and leveraged vector machines use an exponential loss function; LogitBoost uses a negative binomial log-likelihood; Doom II uses a margin loss function induced by hyperbolic tangent function; however, GS-SVMs use the dual of HM-SVMs as a loss function. The reason that the dual of HM-SVMs can be regarded as a loss function can be found in Section III. Another major difference is caused by the basis functions. In previous boosting algorithms, it is a tradition that the basis functions are trees and hence the weights correspond to features. An exception is leveraged vector machines which share a similar idea with GS-SVMs and build kernel machines by greedy stagewise algorithm. In GS-SVMs, whose basis functions are the kernel functions centered on training samples, the weights correspond to samples.

General greedy stagewise algorithm [33] can be described as the following. For $m=1,2,\ldots\ell$

$$(\alpha_m,\beta_m)=\arg\min_{\alpha,\beta}\left(\sum_{i=1}^{\ell}L(y_i,f_{m-1}(\mathbf{x}_i)+\alpha K(\mathbf{x}_i,\mathbf{x}_\beta))\right)$$
$$\text{s.t.}\quad \beta\neq\beta_j,\qquad j=1,2,\ldots m-1 \tag{11}$$

and then

$$f_m(\mathbf{x})=f_{m-1}(\mathbf{x})+\alpha_m K(\mathbf{x},\mathbf{x}_{\beta_m}) \tag{12}$$

where $L(\bullet,\bullet)$ denotes the loss function, $f_0\equiv 0$ and the occurrence of the constraint terms means that each kernel function is selected once at most. The constraint guarantees that the effect of some kernel function is not excessively magnified, which is an effective mechanism for avoiding overfitting. On the other hand, it causes our algorithm to only obtain an approximation solution. This is not the case for the boosting algorithms, which allow modifying the same parameter several times and actually can converge to the minimum of their loss function.

A key observation is that the solution for this two-variable optimization problem in SVMs can be obtained in closed form. For the loss function in SVMs, (11) can be formulated as

$$\begin{aligned}&(\alpha_m,\beta_m)\\&\quad=\arg\min_{\alpha,\beta}\left(\frac{1}{2}\sum_{i=1}^{m-1}\sum_{j=1}^{m-1}\alpha_{\beta_i}\alpha_{\beta_j}y_{\beta_i}y_{\beta_j}K\left(\mathbf{x}_{\beta_i},\mathbf{x}_{\beta_j}\right)\right.\\&\qquad-\sum_{i=1}^{m-1}\alpha_{\beta_i}+\alpha y_\beta\sum_{i=1}^{m-1}\alpha_{\beta_i}y_{\beta_i}K\left(\mathbf{x}_{\beta_j},\mathbf{x}_\beta\right)\\&\qquad\left.+\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)-\alpha\right)\\&\text{s.t.}\quad\alpha\geq 0.\qquad\beta\neq\beta_j,\qquad j=1,2,\ldots m-1.\end{aligned}\tag{13}$$

Eliminating the constant term in (13), we have

$$\begin{aligned}&(\alpha_m,\beta_m)\\&\quad=\arg\min_{\alpha,\beta}\left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)\right.\\&\qquad\left.+\alpha\left(y_\beta\sum_{i=1}^{m-1}\alpha_{\beta_i}y_{\beta_i}K\left(\mathbf{x}_{\beta_j},\mathbf{x}_\beta\right)-1\right)\right)\\&\text{s.t.}\quad\alpha\geq 0,\qquad\beta\neq\beta_j,\qquad j=1,2,\ldots m-1.\end{aligned}\tag{14}$$

Define the gradient vector

$$g_\beta^m=\begin{cases}-1, & \text{if } m=0\\ y_\beta\sum_{k=1}^{m}\alpha_{\beta_k}y_{\beta_k}K\left(\mathbf{x}_{\beta_k},\mathbf{x}_\beta\right)-1, & \text{if } m\geq 1.\end{cases}\tag{15}$$

We can reformulate (14) as

$$\begin{aligned}&(\alpha_m,\beta_m)=\arg\min_{\alpha,\beta}\left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)+\alpha g_\beta^{m-1}\right)\\&\text{s.t.}\quad\alpha\geq 0,\qquad\beta\neq\beta_j,\qquad j=1,2,\ldots m-1.\end{aligned}\tag{16}$$

Equation (16) can be solved in two steps. In the first step, we fix $\beta$ and compute the minimal value $h_\beta^{m-1}$ of (16) with respect to $\alpha$. In the second step, we compute $\beta_m$ by minimizing $h_\beta^{m-1}$ with respect to $\beta$, and then compute $\alpha_m$ in terms of $\beta_m$. Fixing $\beta$, we have the subproblem

$$\begin{aligned}&\min_{\alpha}\left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)+\alpha g_\beta^{m-1}\right)\\&\text{s.t.}\quad\alpha\geq 0.\end{aligned}\tag{17}$$

Since (17) is a single variable quadratic programming, we can give its analytical solution (see Fig. 1)

$$\alpha_\beta=\begin{cases}-\frac{g_\beta^{m-1}}{K(\mathbf{x}_\beta,\mathbf{x}_\beta)}, & \text{if } -\frac{g_\beta^{m-1}}{K(\mathbf{x}_\beta,\mathbf{x}_\beta)}>0\\ 0, & \text{if} -\frac{g_\beta^{m-1}}{K(\mathbf{x}_\beta,\mathbf{x}_\beta)}\leq 0.\end{cases}\tag{18}$$

TABLE I
CHARACTERISTICS OF THE DATA SETS AND THE VALUE OF KERNEL PARAMETER

| Problems | $\theta$ (SVMs) | $\theta$ (GS-SVMs) | Dim | Size |
|---|---|---|---|---|
| Adult-1 | 0.05 | 0.05 | 123 | 1605 |
| Adult-4 | 0.05 | 0.05 | 123 | 4781 |
| Adult-7 | 0.05 | 0.05 | 123 | 16100 |
| Web-1 | 0.05 | 0.10 | 300 | 2477 |
| Web-4 | 0.05 | 0.10 | 300 | 7366 |
| Web-7 | 0.05 | 0.10 | 300 | 24692 |

According to the positive–definite property of kernel function, we have $K(\mathbf{x}_\beta,\mathbf{x}_\beta)>0$. Thus, (18) can be further simplified as

$$\alpha_\beta=\begin{cases}-\frac{g_\beta^{m-1}}{K(\mathbf{x}_\beta,\mathbf{x}_\beta)}, & \text{if } g_\beta^{m-1}<0\\ 0, & \text{if } \mathbf{g}_\beta^{m-1}\geq 0.\end{cases}\tag{19}$$

Combining (17) and (19), we get

$$\begin{aligned}h_\beta^{m-1}&=\min_{\alpha\geq 0}\left(\frac{1}{2}\alpha^2K\left(\mathbf{x}_\beta,\mathbf{x}_\beta\right)+\alpha g_\beta^{m-1}\right)\\&=\begin{cases}-\frac{\left(g_\beta^{m-1}\right)^2}{(2K(\mathbf{x}_\beta,\mathbf{x}_\beta))}, & \text{if } g_\beta^{m-1}<0\\ 0, & \text{if } g_\beta^{m-1}\geq 0.\end{cases}\end{aligned}\tag{20}$$

Considering (19) and (20), we can obtain the parameter pairs $(\alpha_m,\beta_m)$ by the following:

$$\beta_m=\arg\min_{\beta\in Q}\left(h_\beta^{m-1}\right).\tag{21}$$

$$\alpha_m=-\frac{g_{\beta_m}^{m-1}}{K\left(\mathbf{x}_{\beta_m},\mathbf{x}_{\beta_m}\right)}.\tag{22}$$

From (20), we can see that if each of unselected training samples satisfies $g_\beta^{m-1}\geq 0$, the loss function (10) stops decreasing, so GS-SVMs should terminate. Accordingly, the greedy stagewise algorithm for SVMs is shown in algorithm 1.

**Algorithm 1: GS-SVMs**

1. Set $f_0(\mathbf{x})=0$, $\alpha=\mathbf{0}$, $\mathbf{g}^0=-\mathbf{1}$, $\mathbf{h}^0=-\mathbf{1}$,
$Q=\{1,2,\ldots\ell\}$, $P=\varnothing$;
2. For $m=1$ to $\ell$, do:
3. If $\min_{\beta\in Q}\left(g_\beta^{m-1}\right)\geq 0$, stop;
4. $\beta_m=\arg\min_{\beta\in Q}\left(h_\beta^{m-1}\right)$, $\alpha_m=-g_{\beta_m}^{m-1}/K\left(\mathbf{x}_{\beta_m},\mathbf{x}_{\beta_m}\right)$;
5. $P=P\cup\{\beta_m\}$, $Q=Q-\{\beta_m\}$;
6. $g_\beta^m=g_\beta^{m-1}+\alpha_m\,y_\beta\,y_{\beta_m}K\left(\mathbf{x}_\beta,\mathbf{x}_{\beta_m}\right),\quad\beta\in Q$;
7. Update $h_\beta^{m-1}$, $\beta\in Q$ according to (20);
8. $f_m(\mathbf{x})=f_{m-1}(\mathbf{x})+\alpha_mK\left(\mathbf{x},\mathbf{x}_{\beta_m}\right)$;
9. End For
10. End Algorithm

TABLE II
COMPARISONS OF GS-SVMS AND LIBSVM 2.83. SV DENOTES THE NUMBER OF SUPPORT VECTORS; ERROR DENOTES THE MISCLASSIFICATION RATE (PERCENT); K1 DENOTES THE NUMBER OF KERNEL FUNCTION EVALUATIONS WITH USING THE CACHE WITH EACH UNIT CORRESPONDING TO $10^6$ KERNEL FUNCTION EVALUATIONS; K2 DENOTES THE NUMBER OF KERNEL FUNCTION EVALUATIONS WITHOUT USING THE CACHE WITH EACH UNIT CORRESPONDING TO $10^6$ KERNEL FUNCTION EVALUATIONS; AND TIME DENOTES THE RUNTIME WITH EACH UNIT CORRESPONDING TO 1 s. REGULARIZATION PARAMETER $C$ IS SET TO BE 2 AND 4 FOR ADULT AND WEB DATA SETS, RESPECTIVELY

| Problems | LIBSVM 2.83 | | | | | GS-SVMs | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SV | Error | K1 | K2 | Time | SV | Error | K2 | Time |
| Adult-1 | 679 | 15.7 | 1.1 | 3.2 | 0.4 | 650 | 15.9 | 1.0 | 0.3 |
| Adult-4 | 1872 | 15.5 | 9.6 | 13.9 | 3.7 | 1735 | 15.3 | 8.3 | 2.7 |
| Adult-7 | 5878 | 15.1 | 102.7 | 130.3 | 51.9 | 5461 | 15.0 | 87.9 | 28.1 |
| Web-1 | 441 | 2.0 | 1.1 | 5.5 | 0.4 | 325 | 2.6 | 0.8 | 0.2 |
| Web-4 | 907 | 1.6 | 7.1 | 24.5 | 2.4 | 719 | 1.9 | 5.3 | 1.5 |
| Web-7 | 2017 | 1.2 | 52.9 | 134.9 | 18.0 | 1724 | 1.5 | 42.6 | 11.6 |

TABLE III
COMPARISONS OF GS-SVMS AND LIBSVM 2.83 FOR THE DIFFERENT REGULARIZATION PARAMETERS. THE DEFINITION OF K1, K2, AND TIME IS THE SAME AS IN TABLE II

| Algorithm | $C$ | Adult-7 | | | Web-7 | | |
|---|---|---|---|---|---|---|---|
| | | K1 | K2 | Time | K1 | K2 | Time |
| GS-SVMs | | 88 | 88 | 28 | 43 | 43 | 12 |
| LIBSVM 2.83 | $2^{-6}$ | 128 | 128 | 66 | 47 | 55 | 15 |
| | $2^{-4}$ | 113 | 116 | 43 | 51 | 60 | 15 |
| | $2^{-2}$ | 106 | 109 | 54 | 54 | 70 | 16 |
| | $2^{0}$ | 103 | 112 | 52 | 53 | 87 | 17 |
| | $2^{2}$ | 103 | 177 | 71 | 53 | 135 | 18 |
| | $2^{4}$ | 110 | 531 | 64 | 51 | 186 | 18 |
| | $2^{6}$ | 117 | 1664 | 109 | 48 | 243 | 17 |
| | $2^{8}$ | 117 | 4103 | 159 | 46 | 262 | 17 |
| | $2^{10}$ | 116 | 9156 | 309 | 45 | 260 | 16 |

A special case is Gaussian kernel that satisfies $K(\mathbf{x},\mathbf{x}) \equiv 1$, which allows us to simplify (20) as

$$h_\beta^{m-1} = \min_{\alpha \geq 0} \left( \frac{1}{2}\alpha^2 K(\mathbf{x}_\beta, \mathbf{x}_\beta) + \alpha g_\beta^{m-1} \right) = \begin{cases} -\frac{\left(g_\beta^{m-1}\right)^2}{2} & \text{if } g_\beta^{m-1} < 0 \\ 0, & \text{if } g_\beta^{m-1} \geq 0. \end{cases} \quad (23)$$

According to SVMs, we call the samples corresponding to nonzero weights as support vectors. It is easily checked that the computational complexity of GS-SVMs is only $O(n\ell)$, where $n$ is the number of support vectors.

## V. EXPERIMENTS

In this section, we investigate the properties of GS-SVMs on various data sets and compare them with HM-SVMs and soft-margin SVMs. Gaussian kernel $K(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\theta\|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$ is used to construct classifiers. Soft-margin SVMs are trained using LIBSVM 2.83 [35], which implements the improved SMO algorithm. HM-SVMs are constructed using MOSEK optimization toolbox, since SMO works inefficiently for HM-SVMs. All the experiments are run on a personal computer with 3.2-GHz P4 processors, 2-GB memory, and Windows XP operation system. The size of the cache is set to be 1 GB. The optimization process is terminated when the maximal violation of the KKT conditions is within 0.001. For fair comparison, GS-SVMs also use the sparse representation of training samples as LIBSVM 2.83. The shrinking is used if no further explanation is given.

### A. Comparison With LIBSVM 2.83 on Adult and Web Data Sets

In order to validate the speed of GS-SVMs, we compare it with LIBSVM 2.83 on Adult and Web data sets.[1] The characteristics of the data sets and the value of kernel parameter are described in Table I. In the first experiment, we fix $C$ at a suitable value, which gives good generalization performance. The results are shown in Table II as functions of the number of training samples. In the second experiment, we vary $C$ over a wide range. The results are shown in Table III as functions of $C$

As we can see, the number of kernel evaluations of GS-SVMs is smaller than that of LIBSVM 2.83 on the two data sets. LIBSVM 2.83 benefits from the large cache size. Many expensive kernel evaluations are avoided since the entities of the kernel matrix can be accessed from the cache when needed again. However, for the large scale data sets, it is hopeless to fit the larger part of the kernel matrix to the cache, because the space requirement for the kernel matrix grows quadratically with $\ell$. We will illustrate this point in the next section.

From Tables II and III, GS-SVMs are consistently faster than LIBSVM 2.83 on the two data sets, especially for the large $C$ values where the runtime of LIBSVM 2.83 has a sharp increasing. If grid search is used for the selection of free parameters, the number of the trainings of GS-SVMs is significantly

[1]Available at http://research.microsoft.com~jplatt/

TABLE IV
COMPARISONS OF GS-SVMs AND LIBSVM 2.83 ON FOREST DATA SET. THE DEFINITION OF SV, ERROR, K1, K2, AND TIME IS THE SAME AS IN TABLE II. FOR K1 AND K2, EACH UNIT CORRESPONDS TO $10^9$ KERNEL FUNCTION EVALUATIONS

| Problems | GS-SVMs | | | | LIBSVM 2.83 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | SV | Error | K2 | Time | SV | Error | K1 | K2 | Time |
| 20000 | 17923 | 12.4 | 0.4 | 109 | 18561 | 13.1 | 1.7 | 0.9 | 331 |
| 40000 | 30894 | 8.9 | 1.2 | 384 | 33197 | 8.5 | 7.2 | 6.7 | 2277 |
| 60000 | 40722 | 7.0 | 2.4 | 731 | 44401 | 6.5 | 16.5 | 15.5 | 5343 |
| 80000 | 48346 | 6.0 | 3.9 | 1184 | 53546 | 5.4 | 29.3 | 27.3 | 8938 |
| 100000 | 54831 | 5.3 | 5.6 | 1879 | 60921 | 4.7 | 45.4 | 41.8 | 13675 |

TABLE V
COMPARISONS OF GS-SVMs AND OTHER EXISTING ALGORITHMS. SIZE DENOTES THE NUMBER OF TRAINING SAMPLES. THE DEFINITION OF SV, ERROR, AND TIME IS THE SAME AS IN TABLE II

| Algorithms | Size | Error | Time | SV |
|---|---|---|---|---|
| Dong et al's algorithm | 435756 | 10.4 | 6240 | N/A |
| Collobert et al's algorithm | 400000 | 5.61 | 1020 | N/A |
| Core vector machines | 400000 | 2.35 | 24369 | 42182 |
| GS-SVMs | 400000 | 2.60 | 15449 | 103658 |

fewer than that of SVMs since GS-SVMs do not require selecting the regularization parameter. For example, if we try ten different values for $C$ and $\theta$ and perform tenfold cross validation, then GS-SVMs only require retraining 100 times; however, SVMs do that 1000 times. Hence, the training speed of GS-SVMs is significantly more times faster than that of SVMs. Also, we can see that the test errors of GS-SVMs and LIBSVM 2.83 are very close. Thus, we have the conclusion that GS-SVMs are significantly faster in speed than LIBSVM 2.83 and comparable in generalization performance with LIBSVM 2.83.

### *B. Comparison With the Existing Algorithms on Forest Data Set*

To know the behavior of GS-SVMs on very large data sets, we test the proposed algorithm on Forest data set [36]. The data set contains 581 012 samples with seven classes. The dimension of samples is 54. We look only at the binary classification problem of differentiating class 2 from the rest. We randomly select 100 000 samples as the training set and 50 000 samples as the test set.

To get good free parameters, we first choose two small subsets: one for training and the other for validation. The parameters are tuned on the validation set. Then, the parameters $\theta = 1/10\,000$ and $C = 10$ are obtained for SVMs and the parameter $\theta = 1/10\,000$ is obtained for GS-SVMs. This group of data sets covers a wide range of kernel matrix size, which fits into the cache by nearly 30% to only 1%; hence, in most cases, we have to reevaluate the kernel function when some entity of the kernel matrix is needed. Table IV shows the results of GS-SVMs and LIBSVM 2.83 as functions of the number of training samples. We can see that the generalization performance of GS-SVMs and LIBSVM 2.83 is close, however the training time of GS-SVMs is much less than that of LIBSVM 2.83. Note that the shrinking does not apply to Forest data set since it increases the training time of LIBSVM 2.83.

Since different divisions of training and test sets are used in the benchmark test, it is not easy to compare the performance of the different algorithms fairly. Here, we give Collobert *et al.*'s [14] and Dong *et al.*'s [16] results for reference. Dong *et al.* randomly divided the full data set into 435 756 training samples and 145 256 test samples. Then, they trained SVMs on some subsets using the parallel techniques and uniformly combined the outputs of these SVMs to make a final decision. Dong *et al.*'s experiments were conducted on a PC with single intel P4 1.7-GHz processor with 256-k L2 (second-level) cache, SDRAM. The total training time was about 6240 s. The test error was 10.4% for class 2 and the rest. Collobert considered the same binary classification. Their training and test sets consisted of 100 000 and 50 000 samples, respectively. The experiments were conducted on the cluster with 50 Athlon 1.2-Ghz central processing units (CPUs). The test error was about 9.3% for the hard mixture of SVMs and the total training time was 2220 s. When the size of the training set was increased to 400 000 and the local experts were changed to multilayer perceptrons (MLPs), rather than SVMs, the hard probability mixture of MLPs achieved 5.6% test error on the binary classification, and the training time was 1020 s.

TABLE VI
CHARACTERISTICS OF BENCHMARK DATA SETS

| Problems | Size | Class | Dimension |
|---|---|---|---|
| Australian Credit | 690 | 2 | 15 |
| German | 1000 | 2 | 20 |
| Glass | 214 | 6 | 9 |
| Heart | 270 | 2 | 13 |
| Ionosphere | 351 | 2 | 34 |
| Iris | 150 | 3 | 4 |
| Liver disorders | 345 | 2 | 6 |
| Pima Indians Diabetes | 768 | 2 | 8 |
| Segment | 2310 | 7 | 18 |
| Vowel | 528 | 11 | 10 |
| Wdbc | 569 | 2 | 30 |
| Wine | 178 | 3 | 13 |
| Zoo | 101 | 7 | 10 |
| Page | 5473 | 4 | 10 |
| Splice | 3175 | 3 | 240 |
| Dna | 2000/1186 | 3 | 180 |
| Letter | 15000/5000 | 26 | 16 |
| Satimage | 4435/2000 | 6 | 36 |
| Shuttle | 43500/14500 | 9 | 7 |

For comparison, we also run GS-SVMs and core vector machines [19] on 400 000 training samples and the results are shown in Table V. It is observed that GS-SVMs are very competitive with the existing approximation algorithms. GS-SVMs are comparable with CVM and superior to the other two algorithms in terms of the generalization performance. The runtime of Collobert *et al.*'s algorithm is less comparable with the other three algorithms because it exploits the significantly faster machine. The runtime of Collobert *et al.*'s algorithm

TABLE VII
ERRORS OF GS-SVMS, HM-SVMS, AND SVMS ON BENCHMARK DATA SETS. THE RESULTS OF THE BEST METHOD AND OF ALL OTHER METHODS WITH NO SIGNIFICANT DIFFERENCE (THE SIGNIFICANT LEVEL$> 0.05$) ARE SET IN BOLDFACE. NOTE THAT PAIRWISE TWO-TAILED $t$-TESTS ARE NOT APPLIED OVER DNA, LETTER, SATIMAGE, AND SHUTTLE

| Problems | GS-SVMs | HM-SVMs | SVMs |
|---|---|---|---|
| Australian Credit | **0.1507** | 0.2145 | **0.1551** |
| German | **0.2580** | 0.3070 | **0.2460** |
| Glass | **0.2846** | 0.3134 | 0.3319 |
| Heart | **0.1630** | 0.2333 | **0.1677** |
| Ionosphere | **0.0600** | **0.0600** | **0.0598** |
| Iris | **0.0467** | 0.0800 | **0.0400** |
| Liver disorders | 0.3397 | 0.3831 | **0.2871** |
| Pima Indians Diabetes | **0.2279** | 0.2945 | **0.2292** |
| Segment | **0.0268** | **0.0316** | **0.0299** |
| Vowel | **0.0171** | **0.0095** | **0.0095** |
| Wdbc | **0.0228** | **0.0351** | **0.0246** |
| Wine | **0.0111** | 0.0337 | **0.0111** |
| Zoo | **0.0291** | **0.0391** | **0.0391** |
| Mean | 0.1260 | 0.1565 | **0.1255** |
| Page | **0.0355** | / | **0.0307** |
| Splice | **0.0328** | / | **0.0375** |
| Dna | 0.0447 | / | 0.0455 |
| Letter | 0.0270 | / | 0.0202 |
| Satimage | 0.0830 | / | 0.0870 |
| Shuttle | 0.0015 | / | 0.0008 |
| **Mean** | **0.0374** | / | **0.0369** |

should be less than the runtime reported in Table V if it was run on our computer, but one should note that its generalization performance is very poor.

### C. Comparison With SVMs on More Benchmark Data Sets

In order to validate the generalization performance of GS-SVMs, we compare to HM-SVMs, soft-margin SVMs on 15 benchmark data sets from University of California at Irvine (UCI) [37]. These data sets have been extensively used in testing the performance of diversified kinds of learning algorithms. This collection is a well-balanced mixture of the learning tasks with different characteristics, which contains problems with a few or with many training samples, with a few or with many classes, with a few or with many features, and with low or high noise. The characteristics of benchmark data sets are given in Table VI. One-against-one method is used to extend binary classifiers to multiclass classifiers.

For the data sets where the test samples may be available, the error on the test samples is reported in Table VII. For the data sets where the test samples may not be available, tenfold cross validation is run and the average error of tenfold cross validation is reported in Table VII. For each training–test pair, tenfold cross validation is performed on the training set for tuning-free parameters. Before training, we scale all the training samples into the interval $[-1 1]$, and then adjust the test samples using the same linear transformation. The detailed experimental setup is the following.

1) For soft-margin SVMs, kernel parameter and regularization parameter are chosen from intervals $\log_2(\theta) = [-8,-7,-6,-5,\ldots,5,6,7,8]$ and $\log_2(C) = [-1,0,1,2,\ldots 7,8,9,10]$. This range is enough for our problems. The number of trainings needed on each training–test pair is $10\times 17\times 12 = 2040$.
2) For GS-SVMs and HM-SVMs, kernel parameter is chosen from interval $\log_2(\theta) = [-8,-7,-6,-5,\ldots,5,6,7,8]$. The number of trainings needed on each training–test pair is $10\times 17 = 170$. This range is enough for these data sets.

Pairwise two-tailed $t$-tests indicate that GS-SVMs are much better than HM-SVMs on eight data sets, i.e., Australian, German, Glass, Heart, Iris, Liver, Wine, and Diabetes. As for the remaining data sets, GS-SVMs and HM-SVMs obtain the similar performance. Pairwise two-tailed $t$-tests also indicate that GS-SVMs are much better than SVMs on Glass, and worse than SVMs on Liver. As for the remaining data sets, GS-SVMs and SVMs obtain the similar performance.

## VI. WHY DOES GREEDY STAGEWISE ALGORITHM FOR SVMS WORK?

Empirical study has shown that GS-SVMs work well on various data sets. In this section, we will further explore the reason for the success of GS-SVMs. According to statistical learning theory, the generalization performance of learning algorithms not only depends on the empirical risk but also the Vapnik–Chervonenkis (VC) dimension of the hypothesis space. If the VC dimension of the hypothesis space is too large, the empirical risk minimization is possibly not consistent, i.e., the learning algorithms with a small empirical risk may bring a large actual risk.

Chang and Lin [38] have shown that if a kernel function is strictly positive definite, HM-SVMs have unique solution. In other words, HM-SVMs with positive–definite kernel can completely separate the training samples with the presence of noise or not. This means that the hypothesis space is too large and HM-SVMs can suffer from overfitting. In order to obtain good generalization performance, it is necessary to find a right balance between the empirical risk and the VC dimension of the hypothesis space. By introducing a regularization term, soft-margin SVMs can balance the empirical risk and the VC dimension of the hypothesis space and thus obtain the good generalization performance.

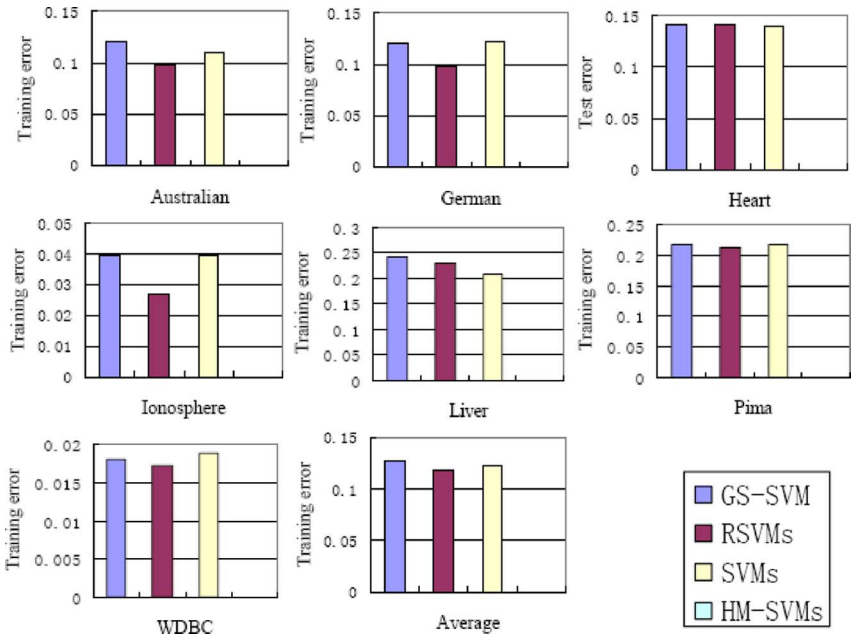


Fig. 2. Training errors of GS-SVMs, RSVMs, SVMs, and HM-SVMs on seven data sets.

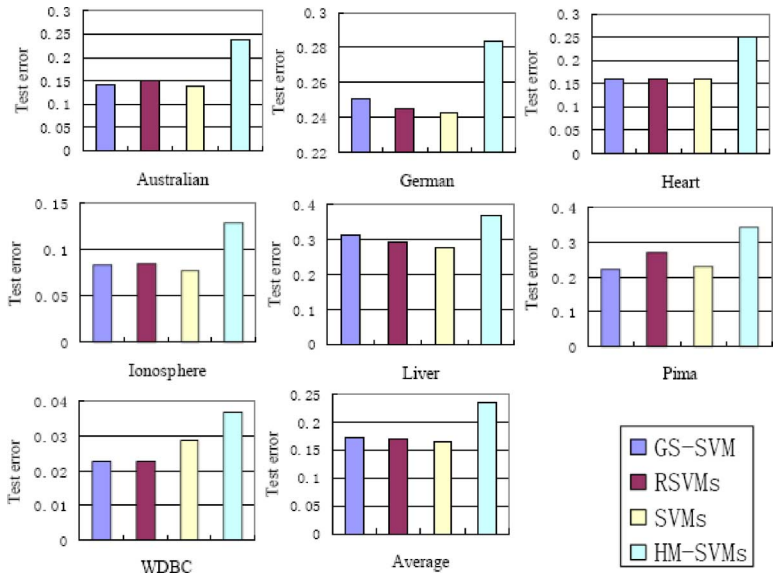


Fig. 3. Test errors of GS-SVMs, RSVMs, SVMs, and HM-SVMs on seven data sets.

GS-SVMs adjust the weights of the kernel functions one by one. The weight of each kernel function centered on the training samples is adjusted once at most, so GS-SVMs run $\ell$ iterations at most. In fact, the early stopping rule can act as an implicit regularization term, and thus, it controls the capacity of hypothesis space. Note that GS-SVMs usually do not give a good approximation solution for HM-SVMs.

The set of hyperplanes

$$\Gamma_\Delta = \left\{ \mathbf{w} \cdot \mathbf{x} - b = 0 : \quad \mathbf{w}^T \mathbf{w} \le \Delta^{-2} \right\} \tag{24}$$

is called the set of $\Delta$-margin separating hyperplanes if they classify vector $\mathbf{x}$ as follows:

$$y = \begin{cases} 1, & \mathbf{w} \cdot \mathbf{x} - b \ge 1 \\ -1, & \mathbf{w} \cdot \mathbf{x} - b \le -1. \end{cases} \tag{25}$$

Note that classifications of vectors $\mathbf{x}$ that fall into the margin $[-1, 1]$ are undefined. For the set of $\Delta$-margin separating hyperplanes, the following theorem holds true.

*Theorem 2 [39]:* Let vectors $\mathbf{x} \in X$ belong to a sphere of radius $R$. Then, the set of $\Delta$-margin separating hyperplanes has the VC dimension $H$ bounded by the inequality

$$H \le \min\left( \frac{R^2}{\Delta^2}, l \right) + 1. \tag{26}$$

It is well known that the VC dimension of the set of hyperplanes is equal to $d+1$, where $d$ is dimensionality of input space. However, Theorem 2 shows the following: 1) the VC dimension of the set of $\Delta$-margin separating hyperplanes can be less than $d+1$; and 2) we can control the VC dimension of the set

of $\Delta$-margin separating hyperplanes by controlling $\Delta$, i.e., the length of the weight vector $\mathbf{w}$.

The weight vector obtained by GS-SVMs is

$$\mathbf{w}^0 = \sum_{i=1}^{l} \alpha_i^0 y_i \Phi(\mathbf{x}_i) \tag{27}$$

where $\alpha_i^0, i = 1, \ldots \ell$ is the solution of GS-SVMs. Consequently, the length of the weight vector is

$$\frac{1}{\Delta_0^2} = \left(\mathbf{w}^0\right)^T \mathbf{w}^0 = \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i^0 \alpha_j^0 y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \tag{28}$$

This means that the separating hyperplane constructed by GS-SVMs belongs to the set $\Gamma_{\Delta_0} = \left\{\mathbf{w} \cdot \mathbf{x} - b = 0 : \quad \mathbf{w}^T\mathbf{w} \leq \Delta_0^{-2}\right\}$. We can look at $\mathbf{w}^T\mathbf{w} \leq \Delta_0^{-2}$ as an implicit constraint for GS-SVMs. If we put the constraint to a prior GS-SVM, the solution of GS-SVMs does not change. The smaller $\Delta_0^{-2}$ is, the smaller the capacity of $\Gamma_{\Delta_0}$ becomes. If $\Delta_0^{-2}$ obtained by GS-SVMs is suitable for the problems at hand, GS-SVMs can give a good regularization parameter implicitly. However, one should remember that the separating hyperplane constructed by GS-SVMs usually is not the hyperplane that minimizes the empirical risk. According to statistical learning theory, the hyperplane minimizing the empirical risk is preferred for the given capacity of hypothesis space. One can find such hyperplane in $\Gamma_{\Delta_0}$ by the following optimization problem:

$$\begin{aligned} &\min \sum_{i=1}^{l} \xi_i \\ &\text{s.t.} \quad y_i\left(<\mathbf{w}, \Phi(\mathbf{x}_i)>\right) \geq 1 - \xi_i, \qquad i = 1, \ldots, \ell, \\ &\xi_i \geq 0, \qquad i = 1, \ldots, \ell, \qquad \mathbf{w}^T\mathbf{w} \leq \Delta_0^{-2}. \end{aligned} \tag{29}$$

Equation (29) also is called rigorous support vector machines (RSVMs) by Bi and Vapnik [40]. The solutions of RSVMs and SVMs coincide if the appropriate $C$ and $\Delta_0^{-2}$ are given. Thus, if GS-SVMs can find a good approximate solution for RSVMs with $\Delta = \Delta_0$, we can explain why GS-SVMs obtain good generalization performance. We will show this by the following experiments.

In Figs. 2 and 3, we give the training errors and test errors of GS-SVMs, RSVMs, SVMs, and HM-SVMs. The kernel parameter of RSVMs is set to the same as for GS-SVMs, and $\Delta_0^{-2}$ in RSVMs is computed by the weight vector obtained by GS-SVMs. Detailed experimental setup of GS-SVMs, SVMs, and HM-SVMs is the same as in Section V. Note that the training errors and test errors are the average of a tenfold cross validation.

From Figs. 2 and 3, we can see that the test error of HM-SVMs is significantly larger than its training error on each data set; however, the test errors of GS-SVMs, RSVMs, and soft-margin SVMs are close to their training errors on each data set. This indicates that HM-SVMs suffer from overfitting; however, GS-SVMs, RSVMs, and soft-margin SVMs avoid it.

From Figs. 2 and 3, we also can see that RSVMs with $\Delta = \Delta_0$ obtain good generalization performance. This indicates that the early stopping rule in GS-SVM can choose an appropriate regularization parameter implicitly. On the other hand, the training error of GS-SVMs is close to that of RSVMs on seven data sets. This shows that GS-SVMs can find a good approximate solution for RSVMs. Thus, we can explain the reason for the success of GS-SVMs: 1) GS-SVMs can choose an appropriate value of $\Delta$ and $\Delta_0$ by the early stopping rule; and 2) GS-SVMs can find a good approximate solution for RSVM with $\Delta = \Delta_0$.

## VII. Conclusion and Discussion

HM-SVMs have a risk of getting overfitting in the presence of noise. To deal with this problem, this paper presents a greedy stagewise algorithm for SVMs, named GS-SVMs, to train HM-SVMs, which attempts to approximately train HM-SVMs while avoiding overfitting. Extensive empirical comparisons show that GS-SVMs are superior to HM-SVMs and comparable with soft-margin SVMs in generalization performance. On the other hand, GS-SVMs also obtain an impressive speedup relative to soft- and hard-margin SVMs; hence, they are very suitable for large scale problems. To explore the reason for the success of GS-SVMs, statistical learning theory is utilized to analyze the empirical results. It seems that the success of GS-SVMs lies in that the early stopping rule in GS-SVMs can act as an implicit regularization term.

Note that although our algorithm is derived under the condition that the kernel function is positive definite, GS-SVMs can also be extended to the nonpositive–definite kernel function. Hence, future work also includes exploring the performance of GS-SVMs using the nonpositive–definite kernel functions.

## References


[1] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 4th Annu. ACM Workshop Comput. Learn. Theory*, Pittsburgh, PA, 1992, pp. 144–152.

[2] S. S. Keerthi and C. J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Comput.*, vol. 15, pp. 1667–1689, 2003.

[3] C. Cortes and V. Vapnik, "Support vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, 1995.

[4] B. Schölkpof, A. Smola, R. C. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural Comput.*, vol. 12, pp. 1207–1245, 2000.

[5] M. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.

[6] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: An application to face detection," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 1997, pp. 130–136.

[7] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods-Support Vector learning*. Cambridge, MA: MIT Press, 1999, pp. 169–184.

[8] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods — Support Vector Learning*. Cambridge, MA: MIT Press, 1999, pp. 185–208.

[9] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," *Neural Comput.*, vol. 13, pp. 637–649, 2001.

[10] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to the SMO algorithm for SVM regression," *IEEE Trans. Neural Netw.*, vol. 11, no. 5, pp. 1188–1194, Sep. 2000.

[11] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "The entire regularization path for the support vector machine," *J. Mach. Learn. Res.*, vol. 5, pp. 1391–1415, 2004.

[12] T. T. Friess, N. Cristianini, and C. Campbell, "The kernel-adatron algorithm: A fast simple learning procedure for support vector machine," in *Proc. 15th Int. Conf. Mach. Learn.*, 1998, pp. 188–196.

[13] S. V. N. Vishwanathan, A. J. Smola, and M. N. Murty, "SimpleSVM," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 760–767.
[14] R. Collobert and S. Bengio, "SVMTorch: Support vector machines for large-scale regression problems," *J. Mach. Learn. Res.*, vol. 1, pp. 143–160, 2001.
[15] R. Collobert, S. Bengio, and Y. Bengio, "Scaling large learning problems with hard parallel mixtures," in *Int. J. Pattern Recognit. Artif. Intell.*, 2003, vol. 17, pp. 349–365.
[16] J. X. Dong, A. Krzyzak, and C. Y. Suen, "Fast SVM training algorithm with decomposition on very large data sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 603–618, Apr. 2005.
[17] G. Bakir, L. Bottou, and J. Weston, "Breaking SVM complexity with cross training," in *Proc. 17th Neural Inf. Process. Syst. Conf.*, 2005, pp. 81–88.
[18] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *J. Mach. Learn. Res.*, vol. 6, pp. 1579–1619, 2005.
[19] I. W. Tsang, J. T. Kwok, and P. M. Cheung, "Core vector machines: Fast SVM training on very large datasets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, 2005.
[20] S. S. Keerthi, O. Chapelle, and D. Decoste, "Building support vector machines with reduced classifier complexity," *J. Mach. Learn. Res.*, vol. 7, pp. 1493–1515, 2006.
[21] L. F. Bo, L. Wang, and L. C. Jiao, "Training support vector machines using greedy stagewise algorithm," in *Proc. 9th Pacific-Asian Conf. Knowl. Disc. Data Dinning*, Hanoi, Vietnam, 2005, pp. 632–638.
[22] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[23] V. Vapnik, *Statistical Learning Theory*. New York: Wiley-Interscience, 1998.
[24] N. Aronszajn, "Theory of reproducting kernels," *Trans. Amer. Math. Soc*, vol. 686, pp. 337–404, 1950.
[25] F. Girosi, "An equivalence between sparse approximation and support vector machines," *Neural Comput.*, vol. 10, pp. 1455–1480, 1998.
[26] S. Chen, F. Cowan, and P. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.
[27] P. Vincent and Y. Bengio, "Kernel matching pursuits," *Mach. Learn.*, vol. 48, pp. 165–187, 2002.
[28] L. C. Jiao, L. F. Bo, and L. Wang, "Fast sparse approximation for least square support vector machines," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 685–697, May 2007.
[29] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[30] Y. Freund, "Boosting a weak learning algorithm by majority," *Inf. Comput.*, vol. 121, pp. 256–285, 1995.
[31] J. H. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Statist.*, vol. 28, pp. 337–407, 2000.
[32] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Boosting algorithms as gradient descent," in *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press, 2000, vol. 12, pp. 512–518.
[33] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, pp. 1189–1232, 2001.
[34] Y. Singer, "Leveraged vector machines," in *Proc. 12th Neural Inf. Process. Syst. Conf.*, 2000, pp. 610–616.
[35] R. E. Fan, P. H. Chen, and C. J. Lin, "Working set selection using second order information for training support vector machines," *J. Mach. Learn. Res.*, vol. 6, pp. 1889–1918, 2005.
[36] J. A. Blackard and D. J. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Comput. Electron. Agriculture*, vol. 24, pp. 131–151, 1999.
[37] C. L. Blake and C. J. Merz, UCI Repository of Machine Learning Databases, 1998 [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html
[38] C. C. Chang and C. J. Lin, "Training $v$-support vector classifiers: Theory and algorithms," *Neural Comput.*, vol. 3, pp. 2119–2147, 2001.
[39] V. Vapnik, "An overview of statistical learning theory," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999.
[40] J. B. Bi and V. Vapnik, "Learning with rigorous support vector machines," in *Proc. 16th Annu. Conf. Learn. Theory*, 2003, pp. 243–257.

**Liefeng Bo** (M'08) was born in Xi'an, China, on February 18, 1978. He received the B.S. degree from Xidian University, Xi'an, China, in 2002 and the Ph.D. degree in circuits and systems from Institute of Intelligent Information Processing, Xidian University, in 2007, .

Currently, he is a Postdoctoral Scholar, collaborating with Prof. Dr. C. Sminchisescuin, at Toyota Technological Institute at Chicago (TTI-C). He is with the Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Institute of Intelligent Information Processing, Xidian University. He has published several papers in some leading journals such as *Neural Computation* and the IEEE TRANSACTIONS ON NEURAL NETWORKS. His current research interests include large scale optimization algorithms and probabilistic models, kernel-based machine, human modeling, and recognition.

**Ling Wang** was born in Xi'an, China, on November 10, 1978. She received the B.S. degree from the School of Science and the M.S. degree in computer science from Xidian University, Xi'an, China, in 2001 and 2005, respectively. She is currently working towards the Ph.D. degree in circuits and systems from the Institute of Intelligent Information Processing, Xidian University.

Her current research interests include pattern recognition, statistical machine learning, and image processing.

**Licheng Jiao** (SM'89) was born in Shaanxi, China, on October 15, 1959. He received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, in 1982, and the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1984 and 1990, respectively, all in electronical engineering.

He is the author or coauthor of more than 150 scientific papers. His current research interests include signal and image processing, nonlinear circuit and systems theory, learning theory and algorithms, optimization problems, wavelet theory, and data mining.