# Training Hard Margin Support vector machines Using Greedy Stagewise Algorithm

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

Institute of Intelligent Information Processing, Xidian University, Shaanxi, China

E-mail: blf0218@163.com

Homepage: http://see.xidian.edu.cn/graduate/lfbo/

#### Abstract

Hard margin support vector machines (HM-SVMs) suffer from getting over-fitting 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 over-fitting 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 400000 training samples demonstrate that GS-SVMs are 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. **Keywords:** Classification, support vector machines (SVMs), greedy stagewise

algorithm, VC dimension

#### **1** Introduction

Hard margin support vector machines have a risk of getting over-fitting in the presence of noise (Boser et al., 1992; Keerthi and Lin, 2003). To deal with this problem, soft margin SVMs (Cortes and Vapnik, 1995; Schölkof et al., 2000) introduce the regularization parameter that allows some training error to obtain large margin. This is a highly effective mechanism for avoiding over-fitting, 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 (Tipping, 2001).

In the past few years, a lot of fast iterative algorithms were presented to cope with the problem ①. Chunking algorithm (Osuna et al., 1997) 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 the case the number of support vectors is very large due to high memory requirement. Joachims (1999) identified this shortage and developed an efficient decomposition scheme, named SVM<sup>light</sup>. The key idea of SVM<sup>light</sup> is to find a feasible direction of steepest descend, in which the number of non-zero elements is set to be a small constant. Platt (1999) took the decomposition idea to an extreme where the size of the working set of SMO algorithm is set to be two and hence an analytical solution for sub-problem is obtained. Keerthi et al. (2001) and Shevade et al. (2000) further improved the performance of SMO by introducing the maximal violating pair working set selection. Hastie et al. (2004) derived an algorithm that can fit the entire path of SVM solutionms for every value of the regularization parameter. Some other examples include Kernel-Adatron (Friess et al., 1998), SimpleSVM (Vishwanathan et al., 2003), SVMTorch (Collobert and Bengio, 2001) and so on.

Recently, there have been many attempts to approximately train SVMs. Collobert et al. (2002) proposed a parallel mixture of SVMs. Dong et al. (2005) introduced a parallel optimization step to quickly remove most of the nonsupport vectors for speeding up SVMs. Bakir et al. (2005) selectively removed training samples using probabilistic estimates related to editing algorithms. Bordes et al. (2005) presented an online algorithm to compute an approximation solution of SVMs. Tsang et al. (2005) showed 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. (2006) 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 (Bo et al., 2005), a greedy stagewise algorithm for approximately training SVMs (GS-SVMs) is presented to deal with the over-fitting of HM-SVMs. Instead of employing the regularization term, GS-SVMs attempt to control the complexity of the hypothesis space by itself. It iteratively builds 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 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:

- The computational complexity of GS-SVMs is O(nℓ), where n is the number of support vectors and ℓ is the number of training samples. Even in the worst case that all the training samples are the support vectors, the computational complexity is only O(ℓ²).
- 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 lie in that its early stopping rule can act as an implicit regularization term.

This paper is organized as follows. In section 2, 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 3. GS-SVMs is detailed derived in section 4. Experiments which demonstrate the speed and generalization performance of GS-SVMs are given in section 5. In section 6, we explore the reason for the success of GS-SVMs. In section 7, the contributions of this paper are summarized and the further research direction is indicated.

## 2 Support vector machines

In this section, we briefly introduce support vector machines. For more details, the interested reader can refer to (Vapnik, 1995 and 1998). 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  $\mathbb{R}^d$ ,  $y_i$  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

$$\min \frac{1}{2} \|\mathbf{w}\|^2$$
s.t.  $y_i \left( \langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle \right) \ge 1, \quad i = 1, \cdots, \ell$ 

$$(2.1)$$

Its Wolfe dual is

$$\min\left(\frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_{i}\alpha_{j}y_{i}y_{j}\Phi(\mathbf{x}_{i})^{T}\Phi(\mathbf{x}_{j})-\sum_{i=1}^{\ell}\alpha_{i}\right).$$
s.t.  $0 \le \alpha_{i}$   $i = 1, \cdots, \ell$ 

$$(2.2)$$

According to Mercer's theory (Aronszajn, 1950), 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 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

$$\min\left(\frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_{i}\alpha_{j}y_{i}y_{j}K(\mathbf{x}_{i},\mathbf{x}_{j})-\sum_{i=1}^{\ell}\alpha_{i}\right).$$
s.t.  $0 \le \alpha_{i}$   $i = 1, \cdots, \ell$ 

$$(2.3)$$

To deal with the non separable case, one often uses soft margin SVMs

$$\min\left(\frac{1}{2}\sum_{i,j=1}^{\ell}\alpha_{i}\alpha_{j}y_{i}y_{j}K(\mathbf{x}_{i},\mathbf{x}_{j})-\sum_{i=1}^{\ell}\alpha_{i}\right).$$
s.t.  $0 \le \alpha_{i} \le C$   $i = 1, \cdots, \ell$ 

$$(2.4)$$

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

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)\right), \qquad (2.5)$$

where  $\alpha_i$  is the solution of (2.4).

# **3 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 (1998).

**Theorem 3.1** (Aronszajn, 1950): Let  $X \subset \mathbb{R}^d$ , a real symmetric function  $K(\mathbf{x}, \mathbf{y})$ ,  $\mathbf{x}, \mathbf{y} \in X$  is positive definite symmetric if and only if for every set of real numbers  $\{\alpha_1, \alpha_2, \dots, \alpha_\ell\}$  and every set of vectors  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_\ell\}$ , we have  $\sum_{i,j=1}^\ell \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \ge 0.$ 

**Definition 3.2** (Aronszajn, 1950): A Hilbert space H of function  $f: X \to \mathbb{R}, X \neq \emptyset$  is called a 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  $\langle f(\bullet), K(\bullet, \mathbf{x}) \rangle_H = f(\mathbf{x})$  and spanning H, i. e.  $H = \overline{\text{span}\{K(\bullet, \mathbf{x}), \mathbf{x} \in X\}}$ .

We call  $||f||_{H} = \sqrt{\langle f, f \rangle_{H}}$ , where  $f \in H$ , reproducing norm. Reproducing kernel Hilbert space (RKHS)  $H_{K}$  induced by  $K(\mathbf{x}, \mathbf{y})$  satisfies the following three properties

- (a)  $K(\bullet, \mathbf{x}) \in H_K$  where  $\mathbf{x} \in X$ ;
- (b)  $\sum_{i=1}^{\ell} \alpha_i K(\bullet, \mathbf{x}_i) \in H_K$ , where  $\alpha_i$  and  $\ell$  are finite;
- (c) For  $f(\bullet) \in H_K$ ,  $\mathbf{x} \in X$ ,  $\langle f(\bullet), K(\bullet, \mathbf{x}) \rangle_H = f(\mathbf{x})$ , where  $\langle \bullet, \bullet \rangle_H$  is the inner product of RKHS. In particular,  $\langle K(\bullet, \mathbf{x}_j), K(\bullet, \mathbf{x}_i) \rangle_H = K(\mathbf{x}_i, \mathbf{x}_j)$ .

According to the property (b), 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

$$\frac{1}{2} \left\| f^*(\mathbf{x}) - \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) \right\|_{H}^{2},$$
s.t.  $\alpha_i \ge 0$   $i = 1, 2, \cdots, \ell$ 

$$(3.1)$$

where  $\left\| \bullet \right\|_{H}$  is RKHS norm. (3.1) can be expanded as

$$\frac{1}{2} \left\| f^*(\mathbf{x}) \right\|_{H}^{2} - \sum_{i=1}^{\ell} \alpha_{i} y_{i} \left\langle f^*(\mathbf{x}), K(\mathbf{x}, \mathbf{x}_{i}) \right\rangle_{H} + \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_{i} \alpha_{j} y_{i} y_{j} \left\langle K(\mathbf{x}, \mathbf{x}_{i}), K(\mathbf{x}, \mathbf{x}_{j}) \right\rangle_{H}$$
  
s.t.  $\alpha_{i} \ge 0, \quad i = 1, 2, \cdots, \ell$ 

Using the reproducing property (c) of kernel function, we can transform Eq. (3.2) into

$$\frac{1}{2} \left\| f^*(\mathbf{x}) \right\|_{H}^{2} - \sum_{i=1}^{\ell} \alpha_i y_i f^*(\mathbf{x}_i) + \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_j y_j K(\mathbf{x}_i, \mathbf{x}_j)$$
s.t.  $\alpha_i \ge 0, \quad i = 1, \cdots, \ell$ 

$$(3.3)$$

(3.2)

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

$$\frac{1}{2} \left\| f^*(\mathbf{x}) \right\|_{H}^{2} - \sum_{i=1}^{\ell} \alpha_i + \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j).$$
s.t.  $\alpha_i \ge 0, \quad i = 1, \cdots, \ell$ 

$$(3.4)$$

Dropping the constant term, we can estimate  $\alpha_i, i = 1, \dots, \ell$  by

$$\min\left(\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_{i}\alpha_{j}y_{i}y_{j}K(\mathbf{x}_{i},\mathbf{x}_{j})-\sum_{i=1}^{\ell}\alpha_{i}\right).$$
s.t.  $0 \le \alpha_{i}, \quad i = 1, \cdots, \ell$ 

$$(3.5)$$

It is easily checked that (3.5) completely amounts to (2.5), which enlightens us to take the Wolfe dual of SVMs as the loss function induced by RKHS norm. If we further constrain  $\alpha_i$ ,  $i = 1, \dots, \ell$  smaller than *C*, we can obtain soft margin SVMs.

## 4 Greedy Stagewise Algorithm for SVMs

Though (3.5) is, in principle, solvable by the classical optimization techniques, in practice it suffers from the two serious problems: (1) its computational complexity usually scales cubically with the size of training samples; (2) there often is a risk of getting over-fitting due to no regularization term.

In this section, we will deal with the above two problems by greedy stagewise

algorithm (GS-SVMs) which attempts to fast approximate (3.5) while avoiding the over-fitting. The dictionary  $D = \{k(\mathbf{x}, \mathbf{x}_i) | i = 1, 2, \dots, \ell\}$  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 one 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 (3.5) 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 (Chen et al., 1991), kernel matching pursuit (backfitting and prefitting version) (Vincent and Bengio, 2002), fast sparse approximation for least square support vector machines (Jiao et al., 2006) 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 (Mallat and Zhang, 1993), AdaBoost (Freund, 1995), LogitBoost (Friedman et al., 2000), Doom II (Mason et al., 2000), gradient boosting (Friedman, 2001), leveraged vector machines (Singer, 2000) 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 3. 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 (Friedman, 2001) can be described as the following. For  $m = 1, 2, \dots \ell$ ,

$$(\alpha_m, \beta_m) = \arg\min_{\alpha, \beta} \left( \sum_{i=1}^{\ell} L(y_i, f_{m-1}(\mathbf{x}_i) + \alpha K(\mathbf{x}_i, \mathbf{x}_{\beta})) \right)$$

$$s.t. \quad \beta \neq \beta_j \quad j = 1, 2, \cdots m - 1$$

$$(4.1)$$

and then

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \alpha_m K(\mathbf{x}, \mathbf{x}_{\beta_m}), \qquad (4.2)$$

where  $L(\cdot, \cdot)$  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 over-fitting. On the other hand, it makes that our algorithm only can 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, (4.1) can be formulated as

Eliminating the constant term in (4.3), we have

$$(\alpha_{m}, \beta_{m}) = \arg\min_{\alpha, \beta} \left( \frac{1}{2} \alpha^{2} K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) + \alpha \left( y_{\beta} \sum_{i=1}^{m-1} \alpha_{\beta_{i}} y_{\beta_{i}} K(\mathbf{x}_{\beta_{j}}, \mathbf{x}_{\beta}) - 1 \right) \right)$$

$$s.t. \quad \alpha \ge 0$$

$$\beta \neq \beta_{j} \quad j = 1, 2, \dots m - 1$$

$$(4.4)$$

Define the gradient vector

$$g_{\beta}^{m} = \begin{cases} -1 & \text{if } m = 0\\ y_{\beta} \sum_{k=1}^{m} \alpha_{\beta_{k}} y_{\beta_{k}} K(\mathbf{x}_{\beta_{k}}, \mathbf{x}_{\beta}) - 1 & \text{if } m \ge 1 \end{cases}$$
(4.5)

We can reformulate (4.4) as

$$(\alpha_{m}, \beta_{m}) = \arg\min_{\alpha, \beta} \left( \frac{1}{2} \alpha^{2} K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) + \alpha g_{\beta}^{m-1} \right)$$
  
s.t.  $\alpha \ge 0$   
 $\beta \ne \beta_{j}$   $j = 1, 2, \dots m-1$  (4.6)

(4.6) can be solved in two steps. In the first step, we fix  $\beta$  and compute the minimal value  $h_{\beta}^{m-1}$  of (4.6) 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

$$\min_{\alpha} \left( \frac{1}{2} \alpha^{2} K \left( \mathbf{x}_{\beta}, \mathbf{x}_{\beta} \right) + \alpha g_{\beta}^{m-1} \right).$$
s.t.  $\alpha \ge 0$ 

$$(4.7)$$

Since (4.7) is a single variable quadratic programming, we can give its analytical solution

$$\alpha_{\beta} = \begin{cases} -g_{\beta}^{m-1} / K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}), & \text{if } -g_{\beta}^{m-1} / K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) > 0\\ 0, & \text{if } -g_{\beta}^{m-1} / K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) \le 0 \end{cases}.$$
(4.8)

According to the positive definite property of kernel function, we have  $K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) > 0$ . Thus (4.8) can be further simplified as

$$\alpha_{\beta} = \begin{cases} -g_{\beta}^{m-1} / K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}), & \text{if } g_{\beta}^{m-1} < 0\\ 0, & \text{if } g_{\beta}^{m-1} \ge 0 \end{cases}.$$
(4.9)



Figure 4.1. Visualization of the solution

Combining (4.7) and (4.9), we get

$$h_{\beta}^{m-1} = \min_{\alpha \ge 0} \left( \frac{1}{2} \alpha^{2} K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) + \alpha g_{\beta}^{m-1} \right) = \begin{cases} -\left(g_{\beta}^{m-1}\right)^{2} / \left(2K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta})\right), & \text{if } g_{\beta}^{m-1} < 0\\ 0, & \text{if } g_{\beta}^{m-1} \ge 0 \end{cases}. (4.10)$$

Considering (4.9) and (4.10), we can obtain the parameter pairs  $(\alpha_m, \beta_m)$  by the following equations

$$\beta_m = \arg\min_{\beta \in Q} \left( h_{\beta}^{m-1} \right). \tag{4.11}$$

$$\alpha_m = -g_{\beta_m}^{m-1} / K \left( \mathbf{x}_{\beta_m}, \mathbf{x}_{\beta_m} \right).$$
(4.12)

From (4.10), we can see that if each of unselected training samples satisfies  $g_{\beta}^{m-1} \ge 0$ , the loss function (3.5) stops decreasing. So GS-SVMs should terminate. Accordingly, the greedy stagewise algorithm for SVMs is shown in Figure 4.2.

Algorithm 1: GS-SVMs
1. Set $f_0(\mathbf{x}) = 0$ , $\alpha = 0$ , $\mathbf{g}^0 = -1$ , $\mathbf{h}^0 = -1$ , $Q = \{1, 2, \dots \ell\}$ , $P = \emptyset$ ;
2. For $m = 1$ to $\ell$ , do:
3. If $\min_{\beta \in Q} \left( g_{\beta}^{m-1} \right) \ge 0$ , 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. $g_{\beta}^{m} = g_{\beta}^{m-1} + \alpha_{m} y_{\beta} y_{\beta_{m}} K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta_{m}}),  \beta \in Q;$
7. Update $h_{\beta}^{m-1}$ , $\beta \in Q$ according to Eq. (4.10);
8. $f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \alpha_m K(\mathbf{x}, \mathbf{x}_{\beta_m});$
9. End For
10. End Algorithm

# Figure 4.2. Flowchart of GS-SVMs

A special case is Gaussian kernel that satisfies  $K(\mathbf{x}, \mathbf{x}) \equiv 1$ , which allows us to simplify (4.10) as

$$h_{\beta}^{m-1} = \min_{\alpha \ge 0} \left( \frac{1}{2} \alpha^{2} K(\mathbf{x}_{\beta}, \mathbf{x}_{\beta}) + \alpha g_{\beta}^{m-1} \right) = \begin{cases} -\left(g_{\beta}^{m-1}\right)^{2} / 2 & \text{if } g_{\beta}^{m-1} < 0\\ 0, & \text{if } g_{\beta}^{m-1} \ge 0 \end{cases}.$$
(4.13)

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.

#### 5 **Experiments**

In this section, we investigate the properties of GS-SVMs on the various data sets and compare it 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 (Fan et al., 2005) 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 1GB. 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.

#### 5.1 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 which are available at http://research.microsoft.com/~jplatt/. The characteristics of the data sets and the value of kernel parameter are described in Table 5.1. In the first experiment, we fix C at a suitable value which gives good generalization performance. The results are shown in Table 5.2 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 5.3 as functions of C.

As we can see, the number of kernel evaluations of GS-SVMs is fewer 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.

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

Table 5.1 Characteristics of the data sets and the value of kernel parameter.

From Table 5.2~5.3, 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 fewer than that of SVMs since GS-SVMs do not require selecting the regularization parameter. For example, if we try 10 different values for *C* and  $\theta$  and perform ten-fold 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 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.

**Table 5.2** Comparisons of GS-SVMs and LIBSVM 2.83. SV denotes the number of support vectors; Error denotes the misclassification rate (%); 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 second. Regularization parameter *C* is set to be 2 and 4 for Adult and Web data sets, respectively.

Problems		LIBSVM 2.83					GS-SVMs			
TIOULEIIIS	SV	Error	K1	K2	Time	SV	Error	K1	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 5.3** Comparisons of GS-SVMs and LIBSVM 2.83 for the differentregularization parameters. The definition of K1, K2 and Time is the same as in Table5.2.

Algorithm	C	Adult-7			Web-7		
Algorium	C	K1	K2	Time	K1	K2	Time
GS-SVMs		88	88	28	43	43	12
	$2^{-6}$	128	128	66	47	55	15
	$2^{-4}$	113	116	43	51	60	15
	$2^{-2}$	106	109	54	54	70	16
LIDCUM	$2^{-0}$	103	112	52	53	87	17
$\frac{1103 \text{ v}}{2.83}$	$2^2$	103	177	71	53	135	18
2.65	$2^{4}$	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

## 5.2 Comparison with the existing algorithms on Forest Data Set

To know the behavior of GS-SVMs on the very large data sets, we test the proposed

algorithm on Forest data set (Blackard and Dean, 1999). 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 100000 samples as the training set and 50000 samples as the test set.

**Table 5.4** 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 5.2. For K1 and K2, each unit corresponds to  $10^9$  kernel function evaluations.

Problems	GS-SVMs			LIBSVM 2.83					
TIOUCIIIS	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

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  $\theta = 1/10000$  and C = 10 are obtained for SVMs and the parameter  $\theta = 1/10000$  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 5.4 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 and 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 and Dong et al's results (Collobert et al. 2002; Dong et al. 2005) for reference. Dong et al. randomly divided the full data set into 435756 training samples and 145256 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's experiments was conducted on a PC with single intel P4 1.7GHz processor with 256k L2 (second-level) cache, SDRAM. The total training time was about 6240 seconds. The test error was 10.4% for class 2 and the rest. Collobert et al. considered the same binary classification. Their training and test sets consisted of 100000 and 50000 samples, respectively. The experiments were conducted on the cluster with 50 Athlon 1.2Ghz CPUs. The test error was about 9.3% for the hard mixture of SVMs and the total training time was 2220 seconds. When the size of the training set was increased to 400000 and the local experts were changed to multilayer perceptrons, rather than SVMs, the hard probability mixture of MLPs achieved 5.6% test error on the binary classification, and the training time was 1020 seconds.

**Table 5.5** 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 5.2.

Algorithms	Size	Error	Time	SV
Dong et al's algorithms	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

For comparison, we also run GS-SVMs and core vector machines (Tsang et al., 2005) on 400000 training samples and the results are shown in Table 5.5. 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 should be less than the runtime reported in Table 5.5 if it was run at our computer, but one should note that its generalization performance is very poor.

# 5.3 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 fifteen benchmark data sets from UCI (Blake and Merz, 1998). 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 training samples or with many, with a few classes or with many, with a few features or with many and with low or high noise. The characteristics of benchmark data sets are given in Table 5.6. One-against-one method is used to extend binary classifiers to multi-class classifiers.

For the data sets where the test samples may be available, the error on the test samples is reported in Table 5.7. For the data sets where the test samples may not be available, ten-fold cross validation is run and the average error of ten-fold cross validation is reported in Table 5.7. For each training-test pair, ten-fold 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:

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

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

 Table 5.6. Characteristics of benchmark 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.

**Table 5.7.** 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 bold face. 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

#### 6 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 algorithm not only depends on the empirical risk but also the 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 (2001) 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 over-fitting. 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.

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 controls the capacity of hypothesis space. Thus, GS-SVMs usually do not find a good approximation solution for HM-SVMs. The set of hyperplanes

$$\Gamma_{\Delta} = \left\{ \mathbf{w} \cdot \mathbf{x} - b = 0 : \quad \mathbf{w}^{T} \mathbf{w} \le \Delta^{-2} \right\},$$
(6.1)

is called the set of  $\Delta$ -margin separating hyperplanes if they classify vector **x** as follows

$$y = \begin{cases} 1 & \mathbf{w} \cdot \mathbf{x} - b \ge 1 \\ -1 & \mathbf{w} \cdot \mathbf{x} - b \le -1 \end{cases}$$
(6.2)

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

**Theorem 6.1** (Vapnik, 1999): 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

$$H \le \min\left(\frac{R^2}{\Delta^2}, l\right) + 1. \tag{6.3}$$

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 6.1 shows that (1) the VC dimension of the set of  $\Delta$ -margin separating hyperplanes can be less than d+1; (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

$$\mathbf{w}^{0} = \sum_{i=1}^{l} \alpha_{i}^{0} y_{i} \Phi\left(\mathbf{x}_{i}\right), \tag{6.4}$$

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

$$\frac{1}{\Delta_0^2} = \left(\mathbf{w}^0\right)^T \mathbf{w}^0 = \sum_{i=1}^l \sum_{j=1}^l \alpha_i^0 \alpha_j^0 y_i y_j K\left(\mathbf{x}_i, \mathbf{x}_j\right).$$
(6.5)

This means that the separating hyperplane constructed by GS-SVMs belongs to the set  $\Gamma_{\Delta_0} = \{\mathbf{w} \cdot \mathbf{x} - b = 0: \mathbf{w}^T \mathbf{w} \le \Delta_0^{-2}\}$ . We can look  $\mathbf{w}^T \mathbf{w} \le \Delta_0^{-2}$  as an implicit constraint for GS-SVMs. If we put the constraint to GS-SVM a prior, the solution of GS-SVMs does not change. The smaller  $\Delta_0^{-2}$ , the smaller the capacity of  $\Gamma_{\Delta_0}$ . If  $\Delta_0^{-2}$  obtained by GS-SVMs is suitable for the problems at hand, GS-SVMs can give a good regularization parameter explicitly. However, 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

$$\min \sum_{i=1}^{l} \xi_{i}$$
  
s.t.  $y_{i} \left( \langle \mathbf{w}, \Phi(\mathbf{x}_{i}) \rangle \right) \geq 1 - \xi_{i}, \quad i = 1, \cdots, \ell$ .  
 $\xi_{i} \geq 0, \quad i = 1, \cdots, \ell$   
 $\mathbf{w}^{T} \mathbf{w} \leq \Delta_{0}^{-2}$  (6.6)

(6.6) also is called rigorous support vector machines (RSVM) by Bi and Vapnik (2003). 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 RSVM with  $\Delta = \Delta_0$ , we can explain why GS-SVMs obtain good generalization performance. We will show this by the following experiments.

In Figure 6.1~6.2, 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

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 section 5. Note that the training errors and test errors are the average of ten-fold cross validation.

From Figure 6.1~6.2, 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 over-fitting; however GS-SVMs, RSVMs and soft margin SVMs avoid it.



Figure 6.1. Training errors of GS-SVMs, RSVMs, SVMs and HM-SVMs on seven data sets

From Figure 6.1~6.2, we also can see that RSVM with  $\Delta = \Delta_0$  obtains 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 RSVM. Thus, we can explain the reason for the success of GS-SVMs: (1) GS-SVMs can choose an appropriate value of  $\Delta$ ,  $\Delta_0$  by the early stopping rule; (2) GS-SVMs can find a good approximate solution for rigorous support vector machine with  $\Delta = \Delta_0$ .



Figure 6.2. Test errors of GS-SVMs, RSVM, SVMs and HM-SVMs on seven data sets

#### 7 Conclusion and Discussion

HM-SVMs have a risk of getting over-fitting 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 over-fitting. 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 it is 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 non-positive definite kernel function. Hence, future work also includes exploring the performance of GS-SVMs using the non-positive definite kernel functions.

#### Acknowledgments

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.

## References

N. Aronszajn. Theory of reproducting kernels. Trans. Amer. Math. Soc, vol. 686, pp.

337-404, 1950.

- G. Bakir, L. Bottou, and J. Weston. Breaking SVM complexity with cross training. *In Proceedings of the 17th Neural Information Processing Systems Conference*, 2005.
- J. B. Bi and V. Vapnik. Learning with rigorous Support Vector Machines. In Proceedings of the 16th Annual Conference on Learning Theory, 2003.
- J. A. Blackard and D. J. Dean. Comparative accuracies of rrtificial neural networks and discriminant analysis in predicting rorest cover types from cartographic variables, *Computers and Electronics in Agriculture*, vol. 24, pp. 131-151, 1999.
- C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. Available at: http://www.ics.uci.edu/~mlearn/MLRepository.html.
- L. F. Bo, L. Wang and L. C. Jiao. Training support vector machines using greedy stagewise algorithm, *In Proceedings of the 9th Pacific-Asian Conference on Knowledge Discovery and Data Dinning*, Hanoi, Vietnam, pp. 632-638, 2005.
- A. Bordes, S. Ertekin, J. Weston and L. Bottou: Fast Kernel Classifiers with Online and Active Learning, *Journal of Machine Learning Research*, vol. 6, pp. 1579-1619, 2005.
- B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, pp. 144-152, ACM Press, 1992.
- C. C. Chang and C. J. Lin. Training *v*-support vector classifiers: theory and algorithms. *Neural Computation*, vol. 3, pp. 2119-2147, 2001.
- S. Chen, F. Cowan, and P. Grant. Orthogonal least squares learning algorithm for

radial basis function networks. *IEEE Transactions on Neural Networks*, vol. 2, pp. 302-309, 1991.

- 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.
- R. Collobert, S. Bengio and Y. Bengio. (2003). Scaling large learning problems with hard parallel mixtures, *International Journal of Pattern Recognition and Artificial Intelligence* vol. 17, pp. 349-365, 2003.
- C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, vol. 20, pp. 273-297, 1995.
- J. X. Dong., A. Krzyzak and C. Y. Suen. Fast SVM training algorithm with decomposition on very large data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 27, pp. 603-618, 2005.
- R. E. Fan, P. H. Chen, and C. J. Lin. Working Set Selection Using Second Order Information for Training Support Vector Machines, *Journal of Machine Learning Research*, vol. 6, pp. 1889-1918, 2005.
- Y. Freund. Boosting a weak learning algorithm by majority, *Information and Computation*, vol. 121, pp. 256-285, 1995.
- J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, vol. 28, pp. 337-407, 2000.
- J. H. Friedman. Greedy Function Approximation: A gradient boosting machine. Annals of Statistics, vol. 29, pp.1189-1232, 2001.

- T. T. Friess, N. Cristianini, and C. Campbell. The kernel-adaraton algorithm: A fast simple learning procedure for support vector machine. *In Proceedings of the 15th International Conference on Machine Learning*, pp. 188-196, 1998.
- F. Girosi. An equivalence between sparse approximation and support vector machines, *Neural Computation*. vol. 10, pp. 1455-1480, 1998.
- L. C. Jiao, L. F. Bo, and L. Wang and. Fast Sparse Approximation for Least Square Support vector machines, *IEEE Transactions on Neural Networks*, vol. 18, pp. 685-697, 2007.
- T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. The Entire Regularization Path for the Support Vector Machine. Journal of Machine Learning Research, vol. 5, pp. 1391-1415, 2004.
- T. Joachims. Making large-scale SVM learning practical, *Advances in Kernel Methods-Support Vector learning*, pp. 169-184, Cambridge, MA: MIT Press, 1999.
- S. S. Keerthi, O. Chapelle, and D. Decoste. Building Support Vector Machines with Reduced Classifier Complexity. *Journal of Machine Learning Research*, vol. 7, pp. 1493-1515, 2006.
- S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, vol. 13, pp. 637-649, 2001.
- S. S. Keerthi and C. J. Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, vol. 15, pp. 1667-1689, 2003.
- S. Mallat and Z. Zhang. Matching pursuit with time-frequency dictionaries. IEEE

Transaction on Signal Processing, vol. 41, pp. 3397-3415, 1993.

- L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. Advances in Neural Information Processing Systems, vol. 12, pp. 512-518, MIT Press, 2000.
- E. Osuna, R. Freund and F. Girosi, Training support vector machines: an application to face detection. *In Proceedings of the conference on Computer Vision and Pattern Recognition*, Puerto Rico, 1997.
- J. Platt. Fast training of support vector machines using sequential minimal optimization. Advances in Kernel Methods --- Support Vector Learning, pp. 185-208, Cambridge, MA: MIT Press, 1999.
- B. Schölkpof, A. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, vol. 12, pp. 1207-1245, 2000.
- S. K. Shevade, S. S. Keerthi, C. Bhattacharyya and K. R. K. Murthy. Improvements to the SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks*, vol. 11, pp. 1188-1194, 2000.
- Y. Singer, Leveraged vector machines, In Proceedings of the 12th Neural Information Processing Systems Conference, 2000.
- M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, vol. 1, pp. 211-244, 2001.
- W. Tsang, J. T. Kwok, and P. M. Cheung. Core vector machines: fast SVM training on very large datasets. Journal of Machine Learning Research, vol. 6, pp. 363–392, 2005.

- V. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
- V. Vapnik. *Statistical Learning Theory*. New York: Wiley-Interscience Publication, 1998.
- V. Vapnik. An overview of statisticallearning theory, *IEEE Transactions on Neural Networks*, vol. 10, pp. 988-999, 1999.
- P. Vincent and Y. Bengio. Kernel matching pursuits. *Machine Learning*, vol. 48, pp. 165-187, 2002.
- S. V. N. Vishwanathan, A. J. Smola, and M. N. Murty. SimpleSVM. In Proceedings of the Twentieth International Conference on Machine Learning, 2003.



Liefeng Bo was born in Xi'an, China, on February 18, 1978. He received the B.S. degree in school of science from Xidian University, Xi'an, China, in 2002, and is currently pursuing the Ph.D. degree in circuits and systems from the Institute of

Intelligent Information Processing, Xidian University.

His current research interests include kernel-based learning, manifold learning, neural networks and computer version. He has published several papers in some leading journals such as Neural Computation and IEEE Transactions on Neural Networks.



**Ling Wang** was born in Xi'an, China, on November 10, 1978. She received the B.S. degree in 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 pursuing 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.

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.