Kobe University Repository : Kernel

PDF issue: 2024-04-27

Decomposition techniques for training linear programming support vector machines

Torii, Yusuke

Abe, Shigeo

(Citation)
Neurocomputing,72(4-6):973-984

(Issue Date)
2009-01

(Resource Type)
journal article

(Version)
Accepted Manuscript

(URL)
https://hdl.handle.net/20.500.14094/90000923

KOBE UNIVERSITY

Decomposition Techniques for Training Linear Programming Support Vector Machines

Yusuke Torii and Shigeo Abe
Graduate School of Engineering, Kobe University, Kobe, Japan

April 14, 2008

Abstract

In this paper, we propose three decomposition techniques for linear programming (LP) problems: (1) Method 1, in which we decompose the variables into the working set and the fixed set, but we do not decompose the constraints, (2) Method 2, in which we decompose only the constraints, and (3) Method 3, in which we decompose both the variables and the constraints into two. By Method 1, the value of the objective function is proved to be non-decreasing (non-increasing) for the maximization (minimization) problem and by Method 2, the value is non-increasing (non-decreasing) for the maximization (minimization) problem. Thus, by Method 3, which is a combination of Methods 1 and 2, the value of the objective function is not guaranteed to be monotonic and there is a possibility of infinite loops. We prove that infinite loops are resolved if the variables in an infinite loop are not released from the working set and Method 3 converges in finite steps. We apply Methods 1 and 3 to LP support vector machines (SVMs) and discuss a more efficient method of accelerating training by detecting the increase in the number of violations and restoring variables in the working set that are released at the previous iteration step.

By computer experiments for microarray data with huge input variables and a small number of constraints, we demonstrate the effectiveness of Method 1 for training the primal LP SVM with linear kernels. We also demonstrate the effectiveness of Method 3 over Method 1 for the nonlinear LP SVMs.

1 Introduction

Support vector machines (SVMs) [1, 2] are widely used for pattern classification. But in training an SVM we need to solve a quadratic programming problem with the number of variables equal to the number of training data. Thus, to speed up training for a large problem, we usually use a decomposition technique, in which the original variables are divided into working variables and fixed variables and a small problem with the working variables is iteratively solved [3, 4]. A special case of the decomposition technique is the sequential minimal optimization (SMO) with the working set size of two [5]. The convergence of the decomposition technique for SVMs is theoretically proved [3, 4, 6, 7, 8] and there are many discussions on working set selection to speed up convergence of

SMO [7, 9, 10, 11] and general decomposition techniques with working set sizes larger than two [12, 13, 14, 15]. By the decomposition techniques training of SVMs for large-scale problems is considerably speeded up.

As a variant of SVMs, linear programming SVMs (LP SVMs), in which the quadratic objective functions are replaced with linear objective functions, have been proposed [16, 17, 18]. In training LP SVMs, we need to solve linear programming problems with the number of variables more than three times the number of training data. But until now, there are not so many discussions on the decomposition techniques for LP SVMs. In [19], a decomposition technique is proposed, in which only a part of linear constraints are used for linear support vector machines. This method confirms monotonic convergence of the objective function and is useful for the problems with a large number of constraints but a small number of variables. In [20], decomposition techniques for SVMs are extended to LP SVMs. Because direct implementation of the decomposition techniques leads to infinite loops, training speedup is done by modifying working set selection when the number of violations of complementarity conditions increases.

In this paper we propose three decomposition techniques for LP programs: Method 1, in which variables are divided into working variables and fixed variables but constraints are all used; Method 2, in which constraints are divided into working constraints and fixed constraints but variables are all used; and Method 3, in which variables and constraints are divided into working and fixed variables and constraints, respectively. We prove that in Method 1, the values of the objective function are non-increasing for a minimization problem during training. While in Method 2 the values of the objective function are non-decreasing for a minimization problem. Therefore, for the combined method, Method 3, the values of the objective function are not monotonic and there is a possibility of infinite loops. We prove that if the variables in an infinite loop are kept in the working set during training, Method 3 converges in finite steps. We apply Methods 1 and 3 to LP SVMs and for Method 3, we discuss more efficient method for training. In computer experiments, we show that Method 1 can accelerate training of linear LP SVMs for microarray data, and Method 3 for training LP SVMs with a large number of training data.

The structure of the paper is as follows. In Sections 2, we propose three decomposition techniques and clarify relations of the proposed decomposition techniques with that for SVMs. Then in Section 3, we apply these methods to LP SVMs and in Section 4, we demonstrate the effectiveness of the proposed methods using some benchmark data sets. Finally in Section 5 we conclude our work.

2 Decomposition Techniques

If the size of a problem is very large, it is natural to consider dividing the problem into small sub-problems and solving the sub-problems iteratively. For an optimization problem, one way is to divide the problem into a working sub-problem and a fixed sub-problem, solve the working sub-problem, re-divide the problem into a working sub-problem and a fixed sub-problem, and iterate the procedure until the solution is obtained. However, to obtain the solution by this method, the objective function needs to be monotonic during the training

process. If not, convergence in finite steps is required.

In the following, we discuss three decomposition techniques for a linear programming problem.

2.1 Formulation

We consider the following problem, which is a generalized version of an LP SVM:

$$\begin{aligned} &\text{minimize} && \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\boldsymbol{\xi} && (1)\\ &\text{subject to} && A\mathbf{x} \geq \mathbf{b} - \boldsymbol{\xi}, \quad \mathbf{x} \geq \mathbf{0}, \quad \boldsymbol{\xi} \geq \mathbf{0}, && (2) \end{aligned}$$

where $\mathbf{c}$ is an m-dimensional constant vector, $\mathbf{d}$ is an M-dimensional vector and $\mathbf{d} > 0$, A is an $M \times m$ constant matrix, $\mathbf{b}$ is an M-dimensional positive constant vector, and $\boldsymbol{\xi}$ is a slack variable vector to make $\mathbf{x} = \mathbf{0}$ and $\boldsymbol{\xi} = \mathbf{b}$ be a feasible solution. Therefore, the optimal solution always exists.

Introducing an M-dimensional slack variable vector $\mathbf{u}$, (2) becomes

$$A\mathbf{x} = \mathbf{b} + \mathbf{u} - \boldsymbol{\xi}, \quad \mathbf{x} \geq \mathbf{0}, \quad \mathbf{u} \geq \mathbf{0}, \quad \boldsymbol{\xi} \geq \mathbf{0}. \tag{3}$$

The dual problem of (1) and (3) is as follows:

$$\begin{aligned} &\text{maximize} && \mathbf{b}^T\,\mathbf{z} && (4)\\ &\text{subject to} && A^T\,\mathbf{z} + \mathbf{v} = \mathbf{c}, \quad \mathbf{z} + \mathbf{w} = \mathbf{d}, \\ & && \mathbf{v} \geq \mathbf{0}, \quad \mathbf{z} \geq \mathbf{0}, \quad \mathbf{w} \geq \mathbf{0}, && (5) \end{aligned}$$

where $\mathbf{z}$ is an M-dimensional vector, $\mathbf{v}$ is an m-dimensional slack variable vector, and $\mathbf{w}$ is an M-dimensional slack variable vector.

The optimal solution $(\mathbf{x}^*, \boldsymbol{\xi}^*, \mathbf{u}^*, \mathbf{z}^*, \mathbf{v}^*, \mathbf{w}^*)$ must satisfy the following complementarity conditions:

$$x_i^*\, v_i^* = 0 \quad \text{for} \quad i = 1, \ldots, m, \tag{6}$$

$$\xi_i^*\, w_i^* = 0, \quad z_i^*\, u_i^* = 0 \quad \text{for} \quad i = 1, \ldots, M. \tag{7}$$

Now solving the primal or dual problem is equivalent to solving

$$\begin{aligned} &A\mathbf{x} = \mathbf{b} + \mathbf{u} - \boldsymbol{\xi}, \quad \mathbf{x} \geq \mathbf{0}, \quad \mathbf{u} \geq \mathbf{0}, \quad \boldsymbol{\xi} \geq \mathbf{0},\\ &A^T\,\mathbf{z} + \mathbf{v} = \mathbf{c}, \quad \mathbf{z} + \mathbf{w} = \mathbf{d},\\ &\mathbf{z} \geq \mathbf{0}, \quad \mathbf{w} \geq \mathbf{0}, \quad \mathbf{v} \geq \mathbf{0},\\ &x_i\, v_i = 0 \quad \text{for} \quad i = 1, \ldots, m,\\ &\xi_i\, w_i = 0, \quad z_i\, u_i = 0 \quad \text{for} \quad i = 1, \ldots, M. \end{aligned}$$

Here, we call x_i active if $x_i > 0$ and inactive if $x_i = 0$. Likewise, the ith constraint is active if $u_i = 0$ and inactive if $u_i > 0$. Notice that even if we delete inactive variables and constraints, we can obtain the same solution as that of the original problem.

By the primal-dual interior-point method, the above set of equations is solved. By the simplex method, if we solve the primal or dual problem, the primal and dual solutions are obtained simultaneously [21]. Therefore, either by the primal-dual interior-point method or the simplex method, we obtain the primal and dual solutions.

2.2 Three Decomposition Techniques

Now we consider the following three decomposition methods to solve (1) and (3).

Method 1, in which, a subset of the variables in $\mathbf{x}$ is optimized using all the constraints, while fixing the remaining variables. Let the set of indices of the subset be $W_{\rm v}$ and the remaining subset be $F_{\rm v}$, where $W_{\rm v} \cap F_{\rm v} = \emptyset$ and $W_{\rm v} \cup F_{\rm v} = \{1, \ldots, m\}$. Assuming $x_i = 0\,(i \in F_{\rm v})$, the original problem given by (1) and (3) reduces as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{i \in W_{\rm v}} c_i\, x_i + \mathbf{d}^T \boldsymbol{\xi} && (8) \\ \text{subject to} \quad & \sum_{j \in W_{\rm v}} A_{ij} x_j = b_i + u_i - \xi_i \quad \text{for} \quad i = 1, \ldots, M, \\ & x_i \geq 0 \quad \text{for} \quad i \in W_{\rm v}, \quad \mathbf{u} \geq \mathbf{0}, \quad \boldsymbol{\xi} \geq \mathbf{0}. && (9) \end{aligned}$$

The dual problem of (8) and (9) is as follows:

$$\begin{aligned} \text{maximize} \quad & \mathbf{b}^T \mathbf{z} && (10) \\ \text{subject to} \quad & \sum_{j=1}^{M} A_{ji}\, z_j + v_i = c_i, \quad v_i \geq 0 \quad \text{for} \quad i \in W_{\rm v}, \\ & \mathbf{z} + \mathbf{w} = \mathbf{d}, \quad \mathbf{z} \geq \mathbf{0}, \quad \mathbf{w} \geq \mathbf{0}. && (11) \end{aligned}$$

Therefore from (10) and (11), if we solve (8) and (9), in addition to the solution of the primal problem, we obtain the solution of the dual problem except for $v_i\,(i \in F_{\rm v})$. Namely, except for $x_i v_i = 0\,(i \in F_{\rm v})$, the complementarity conditions given by (6) and (7) are satisfied. Using the first equation in (5) for $i \in F_{\rm v}$, we can calculate $v_i\,(i \in F_{\rm v})$. Because we assume that $x_i = 0\,(i \in F_{\rm v})$, if $v_i \geq 0\,(i \in F_{\rm v})$, v_i satisfy the constraint and the obtained primal solution is optimal. But if some of v_i are negative, the obtained solution is not optimal.

If the obtained solution is not optimal, we move the indices associated with inactive variables from $W_{\rm v}$ to $F_{\rm v}$, move, from $F_{\rm v}$ to $W_{\rm v}$, the indices associated with the violating variables, and iterate the previous procedure.

By this method, the optimal solution at each iteration step is obtained by restricting the original space

$$\{\mathbf{x} \,|\, A\,\mathbf{x} \geq \mathbf{b} - \boldsymbol{\xi},\ \mathbf{x} \geq \mathbf{0},\ \boldsymbol{\xi} \geq \mathbf{0}\} \tag{12}$$

to

$$\{\mathbf{x} \,|\, A\,\mathbf{x} \geq \mathbf{b} - \boldsymbol{\xi},\ \ \boldsymbol{\xi} \geq \mathbf{0},\ x_i \geq 0 \quad \text{for } i \in W_{\rm v}, \quad x_i = 0 \quad \text{for } i \in F_{\rm v}\}. \tag{13}$$

If the solution is not optimal, we repeat solving the subproblem with the non-zero $x_i\,(i \in W_{\rm v})$ and with the violating variables $x_i\,(i \in F_{\rm v})$. Because of the added violating variables, the objective function of the minimization problem for the newly obtained solution does not increase at least. Namely, the values of the objective function are monotonically non-increasing during the iteration process. Thus, the following theorem holds.

Theorem 1 For Method 1 the sequence of the objective function values is non-increasing and is bounded below by the global minimum of (8).

Method 2, in which we optimize $\mathbf{x}$ using a subset of the constraints. Let the set of indices for the subset be $W_{\rm c}$ and the set of the remaining indices be $F_{\rm c}$. Then we consider the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T \mathbf{x} + \sum_{i \in W_{\rm c}} d_i \, \xi_i && (14) \\ \text{subject to} \quad & A_i \, \mathbf{x} = b_i + u_i - \xi_i, \quad u_i \geq 0, \quad \xi_i \geq 0 \\ & \text{for} \quad i \in W_{\rm c}, \quad \mathbf{x} \geq \mathbf{0}, && (15) \end{aligned}$$

where A_i is the ith row vector of A. The dual problem is given as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in W_{\rm c}} b_i \, z_i && (16) \\ \text{subject to} \quad & \sum_{j \in W_{\rm c}} A_{ji} \, z_j + v_i = c_i \quad \text{for} \quad i = 1, \ldots, M, \quad \mathbf{v} \geq \mathbf{0}, \\ & z_i + w_i = d_i, \quad z_i \geq 0, \quad w_i \geq 0, \quad \text{for} \quad i \in W_{\rm c}. && (17) \end{aligned}$$

For the solution and the dual solution of (14) and (15), we can generate the solution of (1) and (3) as follows: From (15), for $i \in F_{\rm c}$

1. if $A_i \, \mathbf{x} - b_i > 0$, $\xi_i = 0$ and $u_i = A_i \, \mathbf{x} - b_i$,
2. otherwise, $\xi_i = b_i - A_i \, \mathbf{x}$ and $u_i = 0$.

From (17), the first equation of (5) is satisfied if $z_i = 0$ for $i \in F_{\rm c}$. Thus from the second equation of (5), $w_i = d_i \, (i \in F_{\rm c})$. Now the optimal solution $\mathbf{x}$ obtained from (14) and (15) is also the optimal solution of (1) and (3) if

$$\xi_i \, w_i = 0 \quad \text{for} \quad i \in F_{\rm c}. \tag{18}$$

If (18) is not satisfied for some i, we move the indices for inactive constraints from $W_{\rm c}$ to $F_{\rm c}$, move some indices for violating constraints from $F_{\rm c}$ to $W_{\rm c}$, and iterate the preceding procedure.

The optimal solution at each iteration step is obtained by restricting the original space

$$\{\mathbf{x} \,|\, A \, \mathbf{x} \geq \mathbf{b} - \boldsymbol{\xi}, \ \mathbf{x} \geq \mathbf{0}, \ \boldsymbol{\xi} \geq \mathbf{0}\} \tag{19}$$

to

$$\{\mathbf{x} \,|\, A_i \, \mathbf{x} \geq b_i - \xi_i, \quad \xi_i \geq 0 \quad \text{for } i \in W_{\rm c}, \quad \mathbf{x} \geq \mathbf{0}\}. \tag{20}$$

For the non-optimal solution, we repeat solving the subproblem with the active constraints in the working set and with the violating constraints in the fixed set. Therefore, because new constraints are added, the value of the objective function for the newly obtained solution does not decrease at least [22]. Namely, the objective function is monotonically non-decreasing during the iteration process. Thus the following theorem holds.

Theorem 2 For Method 2 the sequence of the objective function values is non-decreasing and is bounded above by the global minimum of (14).

Unlike Theorem 3.2 in [19], we do not claim the finite convergence of Method 2, since according to the implementation of LP infinite loops may occur even if decomposition techniques are not used [21].

Method 3, in which we optimize a subset of variables using a subset of the constraints:

$$\text{minimize} \quad \sum_{i \in W_{\rm v}} c_i\, x_i + \sum_{i \in W_{\rm c}} d_i\, \xi_i \tag{21}$$

$$\text{subject to} \quad \sum_{j \in W_{\rm v}} A_{ij}\, x_j = b_i + u_i - \xi_i, \quad u_i \geq 0, \quad \xi_i \geq 0$$
$$\text{for} \quad i \in W_{\rm c}, \quad x_j \geq 0 \quad \text{for} \quad j \in W_{\rm v}. \tag{22}$$

The dual problem of (21) and (22) is as follows:

$$\text{maximize} \quad \sum_{i \in W_{\rm c}} b_i\, z_i \tag{23}$$

$$\text{subject to} \quad \sum_{j \in W_{\rm c}} A_{ji}\, z_j + v_i = c_i, \quad v_i \geq 0 \quad \text{for} \quad i \in W_{\rm v},$$
$$z_j + w_j = d_j, \quad z_j \geq 0, \quad w_j \geq 0 \quad \text{for} \quad j \in W_{\rm c}. \tag{24}$$

Now we construct, from the solution of (21) and (22), the solution of $\mathbf{x}$ in (1) and (3). Assuming $x_i = 0\,(i \in F_{\rm v})$, $\mathbf{x}$ satisfies

$$A_i\, \mathbf{x} = b_i + u_i - \xi_i \quad \text{for} \quad i \in W_c. \tag{25}$$

We generate ξ_i and u_i $(i \in F_{\rm c})$ as follows:

1. if $A_i\, \mathbf{x} - b_i > 0$, $\xi_i = 0$ and $u_i = A_i\, \mathbf{x} - b_i$,
2. otherwise, $\xi_i = b_i - A_i\, \mathbf{x}$ and $u_i = 0$.

Now assuming $z_i = 0$ for $i \in F_{\rm c}$,

$$(A^T)_i\, \mathbf{z} + v_i = c_i \quad \text{for} \quad i \in W_{\rm v}, \tag{26}$$

and $w_i = d_i$ for $i \in W_{\rm c}$. Further,

$$v_i = c_i - \sum_{j \in W_{\rm c}} A_{ji} z_i \quad \text{for} \quad i \in F_{\rm v}. \tag{27}$$

Now, if $v_i \geq 0\,(i \in F_{\rm v})$ and $\xi_i\, w_i = 0$ $(i \in F_{\rm c})$, the generated solution is optimal. If the solution is not optimal, we move the indices for inactive variables from $W_{\rm v}$ to $F_{\rm v}$, the indices for violating variables from $F_{\rm v}$ to $W_{\rm v}$, the indices for inactive constraints from $W_{\rm c}$ to $F_{\rm c}$, and some indices for violating constraints from $F_{\rm c}$ to $W_{\rm c}$, and iterate the preceding procedure.

Since Method 3 is a combination of Methods 1 and 2, whose objective functions are non-increasing and non-decreasing, respectively, monotonicity of the objective function of Method 3 is not guaranteed. Namely, the following corollary holds:

Corollary 1 For Method 3 the sequence of the objective function values is not guaranteed to be monotonic.

The problem with a non-monotonic objective function is that the solution may not be obtained because of an infinite loop. Since the combinations of the working sets are finite, in an infinite loop, the same working set selection occurs infinitely. Let the working set sequence be

$$\cdots,\ W_k,\ W_{k+1}, \cdots,\ W_{k+t},\ W_{k+t+1},\ W_{k+t+2}, \cdots, W_{k+2t+1} \cdots.$$

where W_k is the working set indices at the kth iteration and $W_k = W_{\mathrm{v},k} \cup W_{\mathrm{c},k}$. If

$$W_k = W_{k+t+1}, \quad W_{k+1} = W_{k+t+2}, \quad \cdots \quad , W_{k+t} = W_{k+2t+1} \tag{28}$$

are satisfied, the same sequence of working set selection occurs infinitely. Equation (28) means that all the variables and constraints that are moved out of the working set are moved back afterward. Thus the infinite loop can be avoided if we keep all the variables and constraints that are fed into the working set even after they become inactive. Namely, for the initial W_{c} and W_{v}, we solve (21) and (22) and delete the indices for the inactive variables and constraints from W_{v} and W_{c}, respectively. Then we repeat solving (21) and (22) adding some violating indices to W_{v} and W_{c}. But we do not delete the indices for inactive variables and constraints from W_{v} and W_{c}, respectively. By this method, the working set size monotonically increases and the method terminates when there is no violating variables and constraints. Evidently the method terminates in finite steps, but the memory usage is inefficient.

To improve memory efficiency, we consider detecting and resolving infinite loops. If an infinite loop given by (28) is detected at the $(k + 2t + 1)$st step, we set

$$W_{k+2t+2} = W_k \cup W_{k+1} \cup \cdots \cup W_{k+t}. \tag{29}$$

We do not remove the indices included in W_{k+2t+2} for the subsequent iterations. We call this procedure infinite loop resolution. This guarantees the convergence of the method in finite steps as the following theorem shows.

Theorem 3 If infinite loop resolution is adopted, Method 3 terminates in finite steps.

Proof If an infinite loop is detected and infinite lop resolution is done, the same infinite loop does not occur in the subsequent iterations. Since the numbers of variables and constraints are finite, the number of infinite loops that will occur is also finite. Thus, the infinite loops are eventually resolved in finite steps. Thus Method 3 with infinite loop resolution terminates in finite steps.

2.3 Comparison of the Three Methods

Method 1 is useful for problems with a large number of variables but with a small number of constraints. For instance, microarray data sets have usually a large or sometimes huge number of variables but a small number of training data, namely constraints. In addition, they are usually linearly separable. Thus, we can use a linear LP SVM applying Method 1 to (31) and (32) discussed later.

Method 2 is suited for problems with a small number of variables but a large number of constraints. But similar to Method 1, Method 2 is only applicable to linear LP SVM expressed by (31) and (32). Method 3 is useful for problems with large numbers of variables and constraints.

2.4 Working Set Selection and Stopping Conditions

Bradley and Mangasarian [19] discussed a decomposition technique for linear LP SVMs, which is similar to Method 2. They divide the constraints into several sets of constraints and solve the problem with the first set of constraints.

Then they solve the problem with the active constraints in the first set and the constraints in the second set. In this way they solve the problem with the active constraints and the next set of constraints and terminate calculations if the solution does not change after several additions of the full set of constraints. (They state that four times of addition are enough.) In this method, we need not use complementarity conditions either for the addition of constraints or stopping calculations but with the expense of an additional computation.

We discuss selection of q variables for Method 1 extending the above method. To simplify discussions, we do not discuss deletion of variables in the working set. Let p be the pointer to the set of indices $\{1, \ldots, m\}$. Initially, we use the first q variables as working variables. Thus, $W_{\rm v} = \{1, \ldots, q\}$ and $p = q + 1$. After solving the subproblem, we check if x_p satisfies the complementarity conditions. If not, we add index p to $W_{\rm v}$. And incrementing p we iterate the above procedure until q indices are added to $W_{\rm v}$. If p exceeds m we set $p = 1$ and repeat the above procedure. Or if p returns back to the point where the search started, we terminate working set selection. We can use similar methods for Method 2. But by Method 3 the objective function values are not monotonic during iteration. Thus if we use the method for accelerating training discussed in [20], we need to count the number of violating variables. Therefore, we cannot use the above method. In the computer experiments in Section 5, we randomly selected q indices.

Taking the similar selection strategies as in SVMs [12, 13, 14, 15], we may be able to improve convergence of the decomposition technique further, but we leave this to the future study.

We can stop training using the decomposition techniques when the complementarity conditions are satisfied. But in some cases the conditions are too strict and it may increase iterations. One way to alleviate the conditions is to slacken the conditions by introducing a threshold and assume that the conditions are satisfied if the conditions are within the threshold. Or we can stop training if the change of the objective function values is within a threshold.

3 Decomposition Techniques for Linear Programming Support Vector Machines

In this section, first we define LP SVMs and discuss LP SVMs with Method 1 and with Method 3 [20]. Then we clarify the relation of the proposed method with the decomposition techniques for SVMs.

3.1 Formulation of Linear Programming Support Vector Machines

Let M m-dimensional input vector $\mathbf{x}_i$ $(i = 1, \ldots, M)$ belong to Class 1 or 2, and the class label be $y_i = 1$ for Class 1 and $y_i = -1$ for Class 2. We map the input space into the high dimensional feature space by the mapping function $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_l(\mathbf{x}))^T$, where l is the dimension of the feature space, and determine the following linear decision function:

$$D(\mathbf{x}) = \mathbf{w}^T \mathbf{g}(\mathbf{x}) + b \tag{30}$$

so that the margin is maximized, where $\mathbf{w}$ is an l-dimensional vector and b is a bias term.

The LP SVM [16, 17, 18] is given by

$$\text{minimize} \qquad Q(\mathbf{w}, b, \boldsymbol{\xi}) = \sum_{i=1}^{l} |w_i| + C \sum_{i=1}^{M} \xi_i \tag{31}$$

$$\text{subject to} \qquad y_i(\mathbf{w}^T \mathbf{g}(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for} \quad i = 1, \ldots, M, \tag{32}$$

where ξ_i is a slack variable and C is a margin parameter to control the trade-off between the classification error of the training data and the generalization ability.

For linear kernels, where $\mathbf{g}(\mathbf{x}) = \mathbf{x}$, we can solve (31) and (32) by linear programming, but for nonlinear kernels we need to treat feature space variables explicitly. To avoid this, we redefine the decision function by [23]

$$D(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i H(\mathbf{x}, \mathbf{x}_i) + b, \tag{33}$$

where α_i and b take real values, and $H(\mathbf{x}, \mathbf{x}')$ is a kernel function:

$$H(\mathbf{x}, \mathbf{x}') = \mathbf{g}^T(\mathbf{x})\, \mathbf{g}(\mathbf{x}'). \tag{34}$$

In this study in addition to linear kernels, we use polynomial kernels with degree d:

$$H(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + 1)^d, \tag{35}$$

and RBF kernels with positive parameter γ:

$$H(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2). \tag{36}$$

.

We define the LP SVM by

$$\text{minimize} \qquad Q(\boldsymbol{\alpha}, b, \boldsymbol{\xi}) = \sum_{i=1}^{M} (|\alpha_i| + C\xi_i) \tag{37}$$

$$\text{subject to} \qquad y_j \left(\sum_{i=1}^{M} \alpha_i H(\mathbf{x}_j, \mathbf{x}_i) + b \right) \geq 1 - \xi_j, \quad \xi_j \geq 0$$
$$\text{for} \quad j = 1, \ldots, M. \tag{38}$$

To solve a problem by the simplex method [21] or the primal-dual interior-point method [24], we need to change variables into nonnegative variables. Then using $\alpha_i^+ \geq 0,\ \alpha_i^- \geq 0,\ b^+ \geq 0,\ b^- \geq 0$, we define $\alpha_i = \alpha_i^+ - \alpha_i^-$, $b = b^+ - b^-$ and convert (37) and (38) into the following linear programming problem:

$$\text{minimize} \qquad Q(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, b^+, b^-, \boldsymbol{\xi}) = \sum_{i=1}^{M} (\alpha_i^+ + \alpha_i^- + C\xi_i) \tag{39}$$

$$\text{subject to} \qquad y_j \left(\sum_{i=1}^{M} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_j, \mathbf{x}_i) + b^+ - b^- \right) + \xi_j \geq 1$$
$$\text{for} \quad j = 1, \ldots, M \tag{40}$$

which has $(3M + 2)$ variables and M constraints.

By introducing slack variables u_i $(i = 1, \ldots, M)$ into (40), (39) and (40) become

$$\text{minimize} \qquad Q(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, b^+, b^-, \boldsymbol{\xi}, \mathbf{u}) = \sum_{i=1}^{M} (\alpha_i^+ + \alpha_i^- + C\xi_i) \tag{41}$$

$$\text{subject to} \qquad y_j \left(\sum_{i=1}^{M} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_j, \mathbf{x}_i) + b^+ - b^- \right) + \xi_j = 1 + u_j$$
$$\text{for} \quad j = 1, \ldots, M, \tag{42}$$

respectively, which have $(4M + 2)$ variables and M constraints.

Assuming the problem given by (39) and (40) primal, the dual problem is as follows:

$$\text{maximize} \qquad Q(\mathbf{z}) = \sum_{i=1}^{M} z_i \tag{43}$$

$$\text{subject to} \qquad \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \le 1 \qquad \text{for} \quad j = 1, \ldots, M, \tag{44}$$

$$\sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \ge -1 \quad \text{for} \quad j = 1, \ldots, M, \tag{45}$$

$$z_j \le C \qquad \text{for} \quad j = 1, \ldots, M, \tag{46}$$

$$\sum_{i=1}^{M} y_i z_i = 0, \tag{47}$$

where z_i $(i = 1, \ldots, M)$ are dual variables and the number of constraints is $(3M + 1)$. Introducing non-negative slack variables v_i^+, v_i^-, w_i $(i = 1, \ldots, M)$, (43)–(47) become as follows:

$$\text{maximize} \qquad Q(\mathbf{z}, \mathbf{v}^+, \mathbf{v}^-, \mathbf{w}) = \sum_{i=1}^{M} z_i \tag{48}$$

$$\text{subject to} \qquad \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i + v_j^+ = 1 \quad \text{for } j = 1, \ldots, M, \tag{49}$$

$$\sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i = v_i^- - 1 \quad \text{for } j = 1, \ldots, M, \tag{50}$$

$$z_j + w_i = C \qquad \text{for } j = 1, \ldots, M, \tag{51}$$

$$\sum_{i=1}^{M} y_i z_i = 0. \tag{52}$$

The linear programming problem given by (48)–(52) has $4M$ variables and $(3M + 1)$ constraints.

Let the optimal solution of the primal problem given by (41) and (42) be $(\boldsymbol{\alpha}^{+*}, \boldsymbol{\alpha}^{-*}, b^{+*}, b^{-*}, \boldsymbol{\xi}^*, \mathbf{u}^*)$ and that of the dual problem given by (48)–(52) be

$(\mathbf{z}^*, \mathbf{v}^{+*}, \mathbf{v}^{-*}, \mathbf{w}^*)$. Then the following complementarity conditions are satisfied:

$$\begin{aligned} \alpha_i^{+*} v_i^{+*} &= 0 \quad \text{for} \quad i = 1, \ldots, M, && (53)\\ \alpha_i^{-*} v_i^{-*} &= 0 \quad \text{for} \quad i = 1, \ldots, M, && (54)\\ \xi_i^* w_i^* &= 0 \quad \text{for} \quad i = 1, \ldots, M, && (55)\\ u_i^* z_i^* &= 0 \quad \text{for} \quad i = 1, \ldots, M. && (56) \end{aligned}$$

Even if we delete $(\mathbf{x}_i, y_i)$ that satisfies

$$\begin{aligned} \alpha_i^{+*} &= 0, && (57)\\ \alpha_i^{-*} &= 0, && (58)\\ \xi_i^* &= 0, && (59)\\ z_i^* &= 0, && (60) \end{aligned}$$

the optimal solution does not change. Namely, training data that do not satisfy either of (57)–(60) are support vectors. Therefore, unlike SVMs, $\mathbf{x}_i$ is a support vector even if $\alpha_i = 0$ so long as either of ξ_i and z_i is nonzero. In classification, however, only nonzero α_i are necessary and the small number of nonzero α_i is important to speed up classification.

3.2 Linear Programming Support Vector Machines Using Method 1

For linear LP SVMs defined by (31) and (32), the number of variable is m and the number of constraints is M. Since the definition of linear SVMs is similar to nonlinear LP SVMs defined by (37) and (38). We only discuss the latter, in which α_i is associated with the ith training datum and for each training datum a constraint is defined. Therefore, $m = M$.

We divide the index set of training data, $T = \{1, \ldots, M\}$, into W and F. Since there is no confusion we do not append the subscript v to the working sets. Then we divide variables in the primal problem given by (41) and (42) and those by the dual problem given by (48)–(52): namely, $\boldsymbol{\alpha}^+ = \{\alpha_i^+ | i = 1, \ldots, M\}$ into $\boldsymbol{\alpha}_W^+ = \{\alpha_i^+ | i \in W\}$ and $\boldsymbol{\alpha}_F^+ = \{\alpha_i^+ | i \in F\}$; $\boldsymbol{\alpha}^-$ into $\boldsymbol{\alpha}_W^-$ and $\boldsymbol{\alpha}_F^-$; $\mathbf{v}^+$ into $\mathbf{v}_W^+$ and $\mathbf{v}_F^+$; $\mathbf{v}^-$ into $\mathbf{v}_W^-$ and $\mathbf{v}_F^-$; $\mathbf{w}$ into $\mathbf{w}_W$ and $\mathbf{w}_F$; $\mathbf{z}$ into $\mathbf{z}_W$ and $\mathbf{z}_F$. Here, we do not divide $\boldsymbol{\xi}$ and $\mathbf{u}$ because they are slack variables.

Fixing $\boldsymbol{\alpha}_F^+$ and $\boldsymbol{\alpha}_F^-$ we optimize the following subproblem:

$$\text{maximize} \quad Q(\boldsymbol{\alpha}_W^+, \boldsymbol{\alpha}_W^-, \boldsymbol{\xi}, b^+, b^-, \mathbf{u}) = \sum_{i \in W} (\alpha_i^+ + \alpha_i^-) + \sum_{i=1}^{M} C\xi_i \quad (61)$$

$$\begin{aligned} \text{subject to} \quad & y_j \left(\sum_{i \in W} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_i, \mathbf{x}_j) + b^+ - b^- \right.\\ & \left. + \sum_{i \in F} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_i, \mathbf{x}_j) \right) + \xi_j = 1 + u_j\\ & \qquad \text{for} \quad j = 1, \ldots, M. \end{aligned} \quad (62)$$

After solving the subproblem fixing $\alpha_i = 0\,(i \in F)$, we check whether the solution is the optimal solution of the entire problem using the complementarity conditions.

Because $\boldsymbol{\alpha}_F^+$ and $\boldsymbol{\alpha}_F^-$ are fixed to zero and $\boldsymbol{\xi}_F$ and $\mathbf{u}_F$ are determined when (61) and (62) are solved, all the primal variables are determined. Since $\mathbf{w}_F^+$ and $\mathbf{z}_\mathbf{F}^+$ are dual variables associated with $\boldsymbol{\xi}_F$ and $\mathbf{u}_F$, respectively, the values of their variables are determined when solved by the simplex method or the primal-dual interior-point method.

But dual variables $\mathbf{v}_F^+$ and $\mathbf{v}_F^-$ are not determined yet. They can be determined so that the following constraints are satisfied:

$$v_j^+ = 1 - \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \quad \text{for} \quad j \in F, \tag{63}$$

$$v_j^- = 1 + \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \quad \text{for} \quad j \in F. \tag{64}$$

If some of them are negative, the entire solution does not satisfy the complementarity conditions. Thus, we need to solve the sub-problem again adding the indices associated with violating variables into W and deleting the indices associated with zero variables in W.

The algorithm of training an LP SVM by Method 1 is as follows:

Step 1 Set $\alpha_i^+ = 0$ and $\alpha_i^- = 0$ for $i = 1, \ldots, M$. And initialize the iteration count: $k = 1$. Go to Step 2.

Step 2 The initial working set be W_1. Select q elements from T and set the remaining elements to F_1. Go to Step 3.

Step 3 Setting $W = W_k$ and $F = F_k$, optimize (61) and (62).

Step 4 Calculate $\mathbf{v}_F^+$ and $\mathbf{v}_F^-$. Go to Step 5.

Step 5 For data corresponding to F_k, check if $v_i^+ \geq 0$ or $v_i^- \geq 0$ is satisfied. If there are violating variables, go to Step 6. Otherwise, stop training. For $k \geq 2$ if $Q_k - Q_{k-1} < \epsilon$ is satisfied, stop training, where Q_k is the value of the objective function for W_k and F_k and ϵ is a small positive value.

Step 6 Move the indices associated with zero variables in W_k to the fixed set and add at most q indices associated with the violating variables to the working set. Let the working set and the fixed set determined be W_{k+1} and F_{k+1} and add k to 1 and go to Step 3.

3.3 Linear Programming Support Vector Machines Using Method 3

In LP SVMs, α_i is associated with the ith training datum and for each training datum a constraint is defined, namely, $M = m$. It is possible to treat the variables and the constraints separately, but to make the definition of the LP SVM simpler, we set $W_\mathrm{v} = W_\mathrm{c}$ and $F_\mathrm{v} = F_\mathrm{c}$. Therefore, for simplify notations, we denote the working and fixed sets by W and F, respectively.

In Section 3.3.1, we define a subproblem for Method 3 and in Section 3.3.2 we discuss infinite loop resolution of Method 3 based on Theorem 3. And in Section 3.3.3 we discuss working set selection for training speedup based on [20].

3.3.1 Definition of Subproblems

We divide the index set $T = \{1, \ldots, M\}$ into the working set W and the fixed set F. Then in (41) and (42) we fix $\boldsymbol{\alpha}_F^+$, $\boldsymbol{\alpha}_F^-$, $\boldsymbol{\xi}_F$, and $\mathbf{u}_F$, delete constraints associated with the fixed set F, and obtain the following sub-problem:

$$\text{minimize} \qquad Q(\boldsymbol{\alpha}_W^+, \boldsymbol{\alpha}_W^-, \boldsymbol{\xi}_W, \mathbf{u}_W, b^+, b^-) = \sum_{i \in W} (\alpha_i^+ + \alpha_i^- + C\xi_i) \quad (65)$$

$$\text{subject to} \qquad y_j \left(\sum_{i \in W} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_i, \mathbf{x}_j) + b^+ - b^- \right.$$

$$\left. + \sum_{i \in F} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_i, \mathbf{x}_j) \right) + \xi_j = 1 + u_j \quad \text{for} \quad j \in W. \quad (66)$$

We solve (65) and (66) for $\boldsymbol{\alpha}_W^+$, $\boldsymbol{\alpha}_W^-$, $\boldsymbol{\xi}_W$, $\mathbf{u}_W$, $\mathbf{v}_W^+$, $\mathbf{v}_W^-$, $\mathbf{w}_W$, and $\mathbf{z}_W$. To check if the obtained solution satisfy the entire solution of (39) and (40), we generate solutions for the fixed set and check the complementarity conditions and constraints for the data associated with the fixed set.

For the primal variables, assuming $\alpha_i^+ = 0$ and $\alpha_i^- = 0$ $(i \in F)$ we generate $\boldsymbol{\xi}_F$ and $\mathbf{u}_F$ using

$$y_j \left(\sum_{i \in W, F} (\alpha_i^+ - \alpha_i^-) H(\mathbf{x}_i, \mathbf{x}_j) + b^+ - b^- \right) + \xi_j = 1 + u_j \quad \text{for} \quad j \in F. \quad (67)$$

Here, (67) is included in (42). Using (33), (67) reduces to

$$y_j D(\mathbf{x}_j) + \xi_j = 1 + u_j \quad \text{for} \quad j \in F. \quad (68)$$

Thus, we generate $\boldsymbol{\xi}_F$ and $\mathbf{u}_F$ as follows:

1. If $y_j D(\mathbf{x}_j) > 1$, $\xi_j = 0$. Thus, from (68), $u_j = y_j D(\mathbf{x}_j) - 1$.
2. If $y_j D(\mathbf{x}_j) \leq 1$, $\xi_j = 1 - y_j D(\mathbf{x}_j)$. Thus, from (68), $u_j = 0$.

Then we generate the dual variables $\mathbf{v}_F^+$, $\mathbf{v}_F^-$, $\mathbf{w}_F$, and $\mathbf{z}_F$. Fixing $z_i = 0$ $(i \in F)$, we generate $\mathbf{v}_F^+$, $\mathbf{v}_F^-$, and $\mathbf{w}_F$ by

$$v_j^+ = 1 - \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \quad \text{for} \quad j \in F, \quad (69)$$

$$v_j^- = 1 + \sum_{i=1}^{M} y_i H(\mathbf{x}_i, \mathbf{x}_j) z_i \quad \text{for} \quad j \in F, \quad (70)$$

$$w_j = C \quad \text{for} \quad j \in F. \quad (71)$$

Equations (69)–(71) are included in (49)–(52). The values of v_j^+, v_j^- $(j \in F)$ obtained from (69) and (70) may be negative, which violate the constraints.

Instead of (66), if we solve the subproblem using (42), the subproblem is optimized by Method 1. Unlike SVMs, deleting the constraints associated with the fixed set in (42), the optimal solution changes. This is because the constraints associated with the fixed set include α_i^+ and α_i^- $(i \in W)$. And since Method 3 is the combination of Methods 1 and 2, which have opposite convergence characteristics, the values of the objective function for Method 3 are not monotonic during training. However, from Theorem 3, if we apply infinite loop resolution to Method 3 the optimal solution is obtained in finite steps.

3.3.2 Infinite Loop Resolution (Method 3-1)

We call Method 3 with the infinite loop resolution discussed in Section 2.2 Method 3-1. This method guarantees the finite steps of convergence. In the following we show the algorithm.

Step 1 Set $\alpha_i^+ = 0,\ \alpha_i^- = 0\ z_i = 0$ for $i = 1, \ldots, M$ and $k = 1$. Go to Step 2.

Step 2 Let the initial working set be W_1 and set q elements from T to W_1 and the remaining elements to F_1. Go to Step 3.

Step 3 Setting $W = W_k,\ F = F_k$, solve (65) and (66), and go to Step 4.

Step 4 Calculate the values of variables in F_k and go to Step 5.

Step 5 Check if the training data associated with F_k satisfy complementarity conditions (53)–(56) and $v_i^+ \geq 0$, $v_i^- \geq 0$. If not go to Step 6, otherwise stop training.

Step 6 If an infinite loop occurs, set W_{k+1} so that it include all the indices of the working set in the loop and set the remaining indices to F_{k+1}. The indices that are newly added to W_{k+1} are kept until the algorithm terminates. Increment k by 1 and go to Step 3. If there is no infinite loop go to Step 7.

Step 7 Select at most q indices in F_k that correspond to violating variables and/or constraints and add them to the working set. Delete the indices in W_k that are not support vectors and move them to the fixed set. Let the working and fixed set thus generated be W_{k+1} and F_{k+1}, respectively. Increment k by 1, and go to Step 3.

3.3.3 Working Set Selection Checking Violating Variables (Method 3-2)

By Method 3-1 we avoid infinite loops, but since the values of the objective function fluctuate during training, the number of iterations increases tremendously in some cases. To accelerate training, we further improve the working set selection strategy.

Let the number of variables that violate the complementarity conditions or constraints at step k be V_k. In general, initially the value of V_k is large but as training proceeds, it decreases and at the final stage $V_k = 0$ or near zero and training is terminated. However, according to our experiments in training of an LP SVM by Method 3-1, it frequently occurred that V_k increased at some iteration step, i.e., $V_k > V_{k-1}$. This slowed down training.

Then if $V_k \geq V_{k-1}$, we consider that important data are deleted from W_{k-1} and moved into the fixed set. Thus among the data that were moved away from W_{k-1} we return back the data that violate the complementarity conditions or the constraints. In addition we keep the data in W_k from deleting even if they satisfy the complementarity conditions.

We call the above method including infinite loop resolution Method 3-2. The detailed flow is as follows.

Step 1 Set $\alpha_i^+ = 0,\ \alpha_i^- = 0$ and $z_i = 0$ for $i = 1, \ldots, M$ and $k = 1$. Go to Step 2.

Step 2 Set the initial working set W_1 by q elements from the index set T and F_1 by the remaining elements. Go to Step 3.

Step 3 Setting $W = W_k,\ F = F_k$, solve (65) and (66) and go to Step 4.

Step 4 Generate values of the variables associated with F_k and go to Step 5.

Step 5 If some training data associated with F_k violate (53)–(56) or $v_i^+ \geq 0$, $v_i^- \geq 0$, go to Step 6. Otherwise terminate training.

Step 6 If an infinite loop exists, select all the indices associated with the working sets that form infinite loops as W_{k+1}. The indices that are newly added to W_{k+1} are kept until the algorithm terminates. Increment k by 1 and go to Step 3. Otherwise, go to Step 7.

Step 7 If $V_k < V_{k-1}$, go to Step 8. Otherwise, add the indices of the data whose indices moved away from W_{k-1} and which violate the complementarity conditions or the constraints at the kth step to the working set. Let the obtained working set be W_{k+1} and the set of the indices of the remaining data be F_{k+1}. Increment k by 1 and go to Step 3. If there are not such data go to Step 8.

Step 8 Check if the data associated with F_k violate the complementarity conditions or constraints. Add at most q indices of the violating variables to the working set. Delete the indices from the working set whose associated data are not support vectors and move them to the fixed set. Let the working set and the fixed set thus determined be W_{k+1} and F_{k+1}, respectively. Increment k by 1 and go to Step 3.

3.4 Relation of the Proposed Methods with the Decomposition technique for Support Vector Machines

Now we discuss the relation of the proposed methods with the decomposition technique for the SVM. The dual problem of the SVM is given by

$$\text{maximize} \qquad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} y_i y_j \alpha_i \alpha_j H(\mathbf{x}_i, \mathbf{x}_j) \tag{72}$$

$$\text{subject to} \qquad \sum_{i=1}^{M} y_i \alpha_i = 0, \tag{73}$$

$$0 \leq \alpha_i \leq C \qquad \text{for} \quad i = 1, \ldots, M, \tag{74}$$

where α_i are dual variables associated with $\mathbf{x}_i$.

To solve (72)–(74) a decomposition technique [3, 4, 25] are proposed as follows. We divide the training data indices $T = \{1, \ldots, M\}$ into W (working set) and F (fixed set), where $W \cup F = \{1, \ldots, M\}$ and $W \cap F = \emptyset$. Dividing $\boldsymbol{\alpha} = \{\alpha_i | i = 1, \ldots, M\}$ into $\boldsymbol{\alpha}_W = \{\alpha_i | i \in W\}$ and $\boldsymbol{\alpha}_F = \{\alpha_i | i \in F\}$ we

optimize, instead of (72)–(74), the following problem:

$$\text{maximize} \quad Q(\boldsymbol{\alpha}_W) = \sum_{i \in W} \alpha_i - \frac{1}{2} \sum_{i,j \in W} \alpha_i \alpha_j y_i y_j H(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i \in W, j \in F} \alpha_i \alpha_j y_i y_j H(\mathbf{x}_i, \mathbf{x}_j) \tag{75}$$

$$\text{subject to} \quad \sum_{i \in W} y_i \alpha_i = -\sum_{i \in F} y_i \alpha_i \tag{76}$$

$$0 \le \alpha_i \le C \qquad \text{for} \quad i \in W, \tag{77}$$

where $\boldsymbol{\alpha}_W = \{\alpha_i | i \in W\}$ is a variable vector but $\boldsymbol{\alpha}_F = \{\alpha_i | i \in F\}$ is a fixed vector.

The number of constraints for the original problem given by (72)–(74) is $M + 1$, but that for (75)–(77) is $|W| + 1$.

The major characteristics of (75)–(77) are that there is only one equality constraint (76) and that the inequality constraints (77) impose the upper and lower bounds. Therefore so long as the variables in the fixed set satisfy (77) before optimization, after optimization the variables do not violate constraints. Thus the objective function value is non-decreasing in optimizing the subprogram given by (75)–(77). Therefore, we can adopt either fixed size or variable size working set [25]. On the other hand, for Methods 3-1 and 3.2 since there is a possibility that the variables and/or constraints in the fixed set are violated after optimization, we used a variable working set size to ensure stable convergence.

4 Computer Experiments

In this section we investigate performance of linear LP SVMs using Method 1 and nonlinear LP SVMs using Methods 1, 3-1, and 3-2 for three types of data sets: microarray data sets, multi-class data sets, and two-class data sets. To speed up training, in Methods 3-1 and 3-2 we freed the non-support vectors after infinite loop resolution. But for all the cases that we tested the optimal solutions were obtained in finite steps.

4.1 Benchmark Data Sets and Evaluation Conditions

Table 1 lists the numbers of training data, test data, inputs, and classes of the microarray problems.[1] For each problem there is one training data set and one test data set. As seen from the table, the microarray data sets are characterized by a large number of input variables but a small number of training/test data. Thus the classification problems are linearly separable and overfitting occurs quite easily. Therefore, usually, feature selection or extraction is performed to improve generalization ability. But since we were interested in the speedup by the decomposition technique, we used the original data sets for classification. We applied Method 1 to the linear LP SVM given by (31) and (32) and evaluated the linear LP SVM for microarray data sets.

Table 2 lists the specification of multi-class problems. Each problem consists of one training data set and one test data set.

[1] http://homes.esat.kuleuven.be/~npochet/Bioinformatics/

Table 1: Benchmark data sets for microarray problems

Data	Training	Test	Inputs	Classes
C. cancer [26]	40	20	2000	2
Leukemia [27]	38	34	7129	2
B. cancer (1) [28]	14	8	3226	2
B. cancer (2) [28]	14	8	3226	2
B. cancer (s) [28]	14	8	3226	2
H. carcinoma [29]	33	27	7129	2
Glioma [30]	21	29	12625	2
P. cancer [31]	102	34	12600	2
B. cancer (3) [32]	78	19	24188	2

Table 2: Benchmark data sets for multiclass and two-class problems

Data	Training	Test	Inputs	Classes
Numeral [33][2]	810	820	12	10
Blood cell [34]	3097	3100	13	13
Thyroid (M) [35]	3772	3428	21	3
Hiragana-50 [36, 37]	4610	4610	50	39
Hiragana-13 [36, 37]	8375	8356	13	38
Hiragana-105 [36, 37]	8375	8356	105	38
German	700	300	20	2

Table 3 lists the number of inputs, training data, test data, and data sets for 13 two-class classification problems.[3] Each problem has 100 or 20 training data sets and their corresponding test data sets.

For evaluating the speedup of the proposed decomposition techniques we used microarray and multiclass data sets and for performance comparison of LP SVMs with SVMs we used all the data sets.

For multiclass problems we used one-against all classification [38], in which one class is separated from the remaining classes. Thus, all the training data were used in training each decision function. We used the revised simplex method [21] to solve linear programming problems. In measuring training time we used a workstation (3.6GHz, 2GB memory, Linux operating system).

4.2 Convergence Characteristics of the Decomposition Techniques

We investigated the convergence characteristics of Methods 1, 3-1, and 3-2 using the german data set, which is the first training data set of the german problem listed in Table 3. We used polynomial kernels with degree 3 and $C = 10$.

Figure 1 shows the objective function values by Method 1 against the number of iterations. And Fig. 2 magnifies the values after the second iterations. As the figures show, the objective function values are non-increasing.

Figure 3 shows the objective function values for Methods 3-1 and 3-2 and

[3] http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm

Table 3: Benchmark data sets for two-class problems.

Data	Train.	Test	Inputs	Sets
Banana	400	4900	2	100
B. cancer	200	77	9	100
Diabetes	468	300	8	100
German	700	300	20	100
Heart	170	100	13	100
Image	1300	1010	18	20
Ringnorm	400	7000	20	100
F. solar	666	400	9	100
Splice	1000	2175	60	20
Thyroid	140	75	5	100
Titanic	150	2051	3	100
Twonorm	400	7000	20	100
Waveform	400	4600	5	100

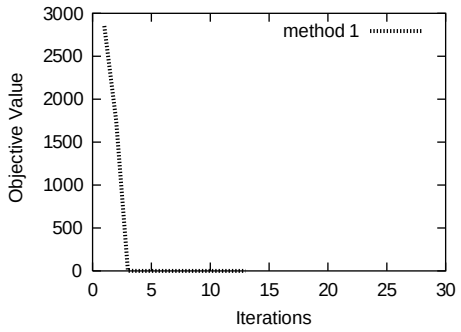

Figure 1: Objective function values for Method 1

Fig. 4 magnifies the values after the eighth iterations. From the figures, the objective function values are not monotonic by Methods 3-1 and 3-2.

Figure 5 shows the numbers of violations by Methods 1, 3-1, and 3-2 as the training proceeds. The number by Method 1 decreased most smoothly. And the number by Method 3-2 decreased to zero a little faster than that by Method 3-1.

Figure 6 shows the working set sizes for the decomposition techniques. For this case, the number of support vectors was 430 and for any of the three decomposition techniques after the working set sizes increased to around the number of support vectors, the sizes did not increased tremendously larger than the number.

We examined the effect of q for the linear LP SVM given by (31) and (32) combined with Method 1 for the breast cancer (3) training data listed in Table 1. Table 4 lists the results. The first row of the results shows the result without decomposition. Since the number of training data is 78, the training time was short without decomposition. By decomposing the problem, the training time was shortened and for $q = 200$, the training time was the shortest. But for $q = 50$

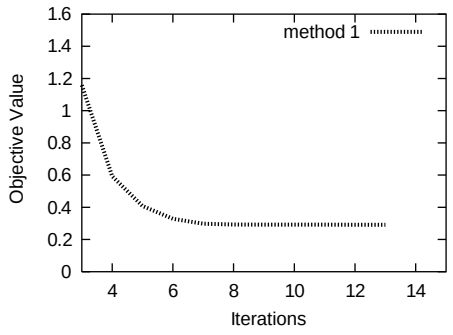

Figure 2: Objective function values for Method 1 after the second iteration

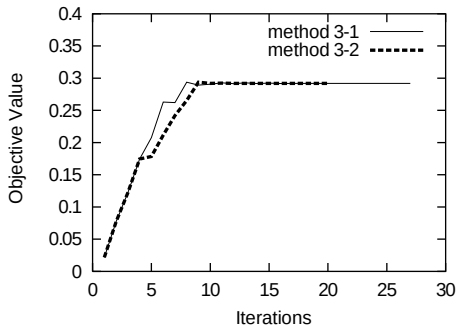

Figure 3: Objective function values for Methods 3-1 and 3-2

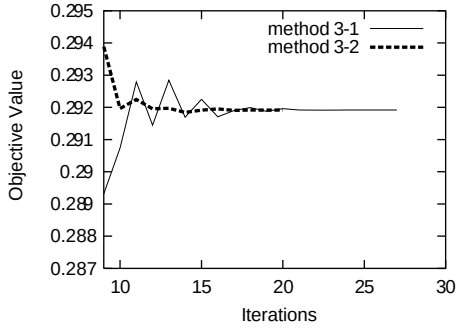

Figure 4: Objective function values for Methods 3-1 and 3-2 after the eighth iteration

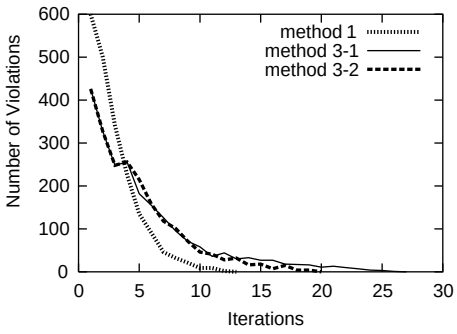

Figure 5: Numbers of violations for the three methods

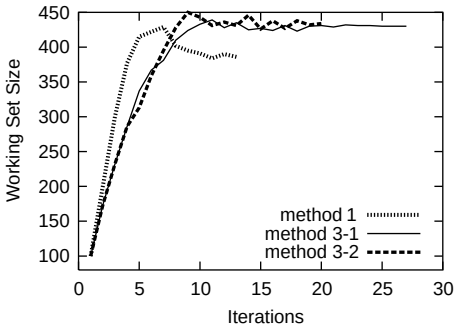

Figure 6: Working set sizes for the three methods

to 500, the training time did not change very much. By the decomposition technique, although the number of iterations was increased, the training was speeded up because of shorter execution time per iteration.

Table 4: Training time of Method 1 for the breast cancer (3) data

q	Time (s)	Iterations	Time/Iteration
—	37	1	37
50	9.2	35	0.26
100	8.7	24	0.36
200	7.8	17	0.46
300	8.9	21	0.42
400	8.1	16	0.50
500	8.4	18	0.47

To determine the value of q for Method 1 with nonlinear kernels and Method 3-2, we measured the training time by Methods 1 and 3-2 for the numeral training data set listed in Table 2. We used the RBF kernels with $\gamma = 1$ and $C = 10$. Table 5 shows the execution time, the number of iterations, and the execution time per iteration for Method 1. The number of iterations is the total number of iteration steps. Thus without decomposition techniques, the number of iterations is the number of classes because we used one-against all classification. The first row of the result in the table is the result without the decomposition techniques.

From the table, using Method 1, training time was the shortest for $q = 500$ but much longer than without using Method 1. This is because the execution time per iteration was comparable with that without Method 1. Since Method 1 uses all the constraints, it is not suited to use Method 1 for nonlinear kernels. Thus, in the following study we do not evaluate Method 1 for nonlinear kernels.

Table 5: Training time of Method 1 for the numeral data

q	Time (s)	Iterations	Time/Iteration
—	1827	10	183
50	5396	39	138
100	4916	35	140
200	4256	31	137
300	3875	27	144
400	3936	27	146
500	3865	26	149

Table 6 shows the execution time, the number of iterations, and the execution time per iteration for Method 3-2. From the table, for $q = 25$ the training time was the shortest. From $q = 25$ to 100, the execution time per iteration was drastically reduced and 400 to 1000 times speedup was obtained.

Table 6: Training time of Method 3-2 for the numeral data

q	Time (s)	Iterations	Time/Iteration
—	1827	10	183
25	1.8	117	0.015
50	1.9	75	0.025
75	3.0	69	0.043
100	4.6	69	0.067

4.3 Effect of Decomposition Techniques

Using the microarray data sets, we investigated the speedup by Method 1 for the linear LP SVM. We set $C = 1000$ and $q = 50$. Table 7 lists the results. The "Rates" column lists the recognition rates for the test data and the training data in parentheses. The numeral in the parentheses in the "SVs" column shows the number of nonzero α_i. Only the nonzero α_i do not constitute the solution of the LP SVM. But if this number is smaller that that of support vectors, speedup of classification is achieved. The "Method 1" column lists the training time using Method 1. From the table, the speedup by Method 1 is 1.3 to 4. The low speedup ratio is because the number of constraints is too small and without decomposition, training is not so slow.

Table 7: Speedup by linear LP SVM combined with Method 1 for the microarray problems

Data	Rates	SVs	No Decomp. [s]	Method 1 [s]	Speedup
C. cancer	77.27 (100)	19 (18)	0.50	0.37	1.4
Leukemia	61.76 (100)	34 (33)	2.2	1.0	2.2
B. cancer (1)	87.50 (100)	12 (11)	0.19	0.15	1.3
B. cancer (2)	87.50 (100)	12 (11)	0.19	0.15	1.3
B. cancer (s)	37.50 (100)	14 (13)	0.20	0.15	1.3
Carcinoma	81.48 (100)	32 (31)	2.0	1.0	2.0
Glioma	48.28 (100)	20 (10)	1.7	1.3	1.3
P. cancer	26.47 (100)	97 (96)	51	15	3.4
B. cancer (4)	52.63 (100)	76 (75)	37	9.2	4.0

We investigated the speedup of Methods 3-1 and 3-2 for the nonlinear LP SVM using the multiclass problems. We considered two cases: 1) RBF kernels with $\gamma = 1$ and $C = 10$ and 2) RBF kernels with $\gamma = 10$ and $C = 10000$. We set $q = 50$ and measured the training time. Table 8 shows the result. In the table, in the "Cond." column, for example, "Numeral 1" and "Numeral 2" denote that RBF kernels with $\gamma = 1$ and $C = 10$ and RBF kernels with $\gamma = 10$ and $C = 10000$ are used for the numeral training data set, respectively. The "Rates" column lists the recognition rates of the test data and the training data in the parentheses. And the "SVs" column lists the number of support vectors and the numeral in parentheses shows the number of nonzero α_i. The "Speedup" column lists the speedup of Method 3-2 over LP SVM training without using the decomposition techniques for the numeral, blood cell, and thyroid data sets

but for the other data sets we list the speedup of Method 3-2 over Method 3-1 because the training time was too long if decomposition techniques were not used. Method 3-2 obtained the speedup of three to four orders of magnitudes over the training without the decomposition technique and speedup from 31 to 1.3 over Method 3-1.

Table 8: Speedup by the decomposition techniques for multiclass problems

Cond.	Rates	SVs	No Decomp. [s]	Method 3-1 [s]	Method 3-2 [s]	Speedup
Numeral 1	99.27 (99.63)	18 (6)	1827	2.54	1.88	972
Numeral 2	99.51 (100)	18 (9)	1606	14.7	2.18	737
Blood 1	88.77 (91.19)	200 (9)	787073	1141	681	1156
Blood 2	91.52 (99.77)	120 (62)	1164302	27089	4831	241
Thyroid (M) 1	94.22 (94.43)	315 (11)	541896	2605	2485	218
Thyroid (M) 2	97.23 (99.81)	179 (83)	1481248	8517	2443	606
H50 1	89.61 (91.32)	107 (14)	—	2710	1387	2.0
H50 2	98.11 (100)	57 (31)	—	4838	478	10
H13 1	91.49 (91.47)	183 (9)	—	4434	3014	1.5
H13 2	99.23 (100)	57 (29)	—	46663	1503	31
H105 1	96.69 (96.97)	134 (22)	—	3627	2792	1.3
H105 2	100 (100)	70 (38)	—	26506	1645	16

4.4 Comparison with Support Vector Machines

We compared the performance of LP SVMs with that of SVMs. For linear kernels we determined the values of C for the LP SVM and SVM by fivefold cross-validation. For nonlinear kernels, we performed fivefold cross-validation for the SVM using RBF kernels and determined the values of γ and C. Then using the same kernel parameter value we determined the value of C for the LP SVM by fivefold cross-validation. This means that we compared the performance of LP SVM and SVM in the same feature space. For C we selected the value from $\{0, 1, 50, 100, 500, 1000, 2000, 3000, 5000, 10000, 50000, 100000\}$ and for γ from $\{0, 0.5, 1, 5, 10, 15\}$.

We trained the SVM by the primal-dual interior-point method combined with the decomposition technique. For the LP SVM and SVM we set $q = 50$ if the decomposition technique was used.

For the microarray problems, we used linear kernels. Table 9 shows the results. In the table, LP SVM denotes that (31) and (32) are used and LP SVM (KE) (37) and (38). For LP SVM (KE) we did not use the decomposition technique. In the table, in each row the maximum recognition rate for the test data is shown in boldface. Comparing the LP SVM, LP SVM (KE), and SVM, there is not much difference in the number of support vectors and training time but the recognition rate of the test data for the SVM is a little better than that for LP SVM or LP SVM (KE).

The number of nonzero support vectors for the breast cancer (2) is zero. This means that the decision function contains only the bias term and all the data are classified into one class. The degenerate solution happened because of a small number of training data and a small value of C [38].

Table 9: Performance comparison for microarray problems.

Data	LP SVM				LP SVM (KE)				SVM			
	C	SVs	Time (s)	Rates	C	SVs	Time (s)	Rates	C	SVs	Time (s)	Rates
C. cancer	10	19 (18)	0.37	**77.27** (100)	100	19 (13)	0.39	72.73 (100)	100	23	0.50	**77.27** (100)
Leukemia	1	34 (33)	0.97	61.76 (100)	50	29 (23)	1.1	**91.18** (100)	50	31	1.4	85.29 (100)
B. cancer (1)	1	12 (11)	0.15	**87.5** (100)	100	10 (9)	0.18	**87.5** (100)	1	13	0.23	62.5 (71.43)
B. cancer (2)	1	12 (11)	0.16	**87.5** (100)	1	10 (0)	0.17	62.5 (64.29)	100	14	0.20	**87.5** (100)
B. cancer (s)	1	14 (13)	0.15	37.5 (100)	1	8 (0)	0.18	**62.5** (71.43)	1	12	0.20	**62.5** (71.43)
Carcinoma	1	32 (31)	0.99	**81.48** (100)	1	24 (0)	1.0	70.37 (63.64)	1	31	1.7	70.37 (63.63)
Glioma	1	20 (19)	1.3	48.28 (100)	50	17 (16)	1.5	48.28 (100)	50	19	1.7	**55.17** (100)
P. cancer	1	97 (96)	14	26.47 (100)	50	79 (75)	7.9	26.47 (99.02)	100	98	11	**58.82** (100)
B. cancer (3)	1	76 (75)	10	52.63 (100)	100	60 (55)	12	78.95 (96.15)	100	74	29	**84.21** (100)

Table 10: Performance comparison for multiclass problems.

Data	γ	LP SVM				SVM			
		C	SVs	Time (s)	Rates	C	SVs	Time (s)	Rates
Numeral	1	50	15 (7)	2.07	**99.63** (100)	50	16	0.694	99.27 (99.88)
Blood	5	500	107 (35)	582	92.55 (97.38)	1000	93	16	**93.71** (97.29)
Thyroid (M)	5	5000	196 (74)	2069	97.32 (99.42)	10^5	174	14	**97.40** (99.52)
H50	15	50	58 (33)	441	98.11 (100)	50	80	110	**99.28** (100)
H13	15	50	59 (30)	1266	99.50 (99.88)	1000	42	118	**99.75** (100)
H105	5	50	71 (38)	1951	99.92 (100)	100	77	379	**100** (100)

Table 10 shows the performance comparison of the LP SVM and SVM using RBF kernels for the multiclass problems. Except for the numeral data set, the recognition rate of the test data for the SVM is better than that for LP SVM and training is faster. But the number of nonzero α_i for the LP SVM is much smaller than that for the SVM. Thus a sparser classifier was obtained by the LP SVM.

Table 11 shows the performance comparison of the LP SVM and SVM using RBF kernels for the two-class problems. The "E. Rate" lists the average classification error and the standard deviation. The boldface denotes that the average error rate or the standard deviation is statistically better with the significance level of 0.05. From the table, it is seen that the tendency is the same with that of multiclass problems. Namely, Training time for the SVM is shorter than that for the LP SVM and the classification error tends to be smaller. But the number of nonzero α_i for the LP SVM is smaller than that of the SVM and thus the sparsity of the LP SVM is much higher than that of the SVM.

4.5 Discussions

Slow training of LP SVMs compared to SVMs may be caused by the fact that optimization of a subproblem may produce violating variables and/or constraints associated with the fixed set. Another reasons may be using the simplex method

Table 11: Performance comparison for two-class problems.

Data	γ	LP SVM				SVM			
		C	SVs	Time (s)	E. Rate	C	SVs	Time (s)	E. Rate
Banana	15	100	106 (18)	3.9	10.6±0.49	100	173	0.45	**10.4**±0.46
B. Cancer	1	50	115 (24)	4.4	26.9±4.6	10	118	0.32	**25.6**±4.5
Diabetes	10	1	263 (8)	5.3	23.4±1.8	1	268	1.3	23.4±1.7
German	5	1	410 (28)	397	23.9±2.2	1	416	7.5	23.8±2.1
Heart	0.1	50	75 (10)	0.57	16.5±3.4	50	74	0.090	16.1±3.1
Image	10	500	179 (9)	181	3.52±0.59	1000	149	1.5	**2.84**±0.50
Ringnorm	15	1	57 (16)	0.85	**1.90±0.23**	1	131	0.91	2.64±0.35
F. Solar	1	1	461 (4)	756	33.3±1.8	1	522	7.8	**32.3**±1.8
Splice	10	10	501 (464)	2689	13.2±0.86	10	749	13	**10.8**±0.71
Thyroid	5	50	19 (7)	0.038	4.76±2.9	1000	13	0.010	4.05±**2.3**
Titanic	10	10	68 (8)	0.35	22.5±0.99	10	113	0.23	22.4±1.0
Twonorm	1	1	132 (9)	0.90	2.86±**0.44**	1	193	1.4	**2.02**±0.64
Waveform	5	10	107 (24)	4.4	10.8±0.44	10	114	0.58	**10.3**±0.40

in training LP SVMs and the strict convergence test using the complementarity conditions. The latter reasons may be solved by using the primal-dual interior-point method and the slack convergence test. In our computer experiments, the maximum number of training data was 8375. To handle larger size problems, we may still need to speed up training by the above methods.

We used $q = 50$ for all the computer experiments. For the problems tested that value seemed to be suboptimal. But for huge size problems, a larger value may lead to faster training. In such problems to set a suboptimal value we need trials and errors.

5 Conclusions

In this paper, we proposed three decomposition techniques for LP SVMs. Method 1 decomposes variables but use all the constraints. Method 2 decomposes constraints but use all the variables. Method 3 decomposes variables and constraints. We proved that by Methods 1 and 2, the objective function values are monotonic during training but by Method 3 the objective function values are not guaranteed to be monotonic. To guarantee convergence in finite steps, we proposed infinite loop resolution for Method 3. Then we discussed how to improve convergence of Method 3 resolving infinite loops and clarified relationship of the proposed methods with the decomposition techniques for SVMs.

By the computer experiments, we demonstrated that Method 1 worked to speedup training of linear LP SVMs for the microarray problems with a large number of variables and a small number of constraints. And using the improved Method 3, we demonstrated that training of nonlinear LP SVMs was accelerated considerably for the several multiclass problems.

References

[1] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.

[2] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[3] E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Neural Networks for Signal Processing VII—Proceedings of the 1997 IEEE Signal Processing Society Workshop*, pages 276–285, 1997.

[4] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[5] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. MIT Press, Cambridge, MA, 1999.

[6] C.-J. Lin. On the convergence of the decomposition method for support vector machines. *IEEE Transactions on Neural Networks*, 12(6):1288–1298, 2001.

[7] S. S. Keerthi and E. G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, 46:351–360, 2002.

[8] N. Takahashi and T. Nishi. Global convergence of decomposition learning methods for support vector machines. *IEEE Transactions on Neural Networks*, 17(6):1362–1369, 2006.

[9] 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*, 6:1889–1918, 2005.

[10] P.-H. Chen, R.-E. Fan, and C.-J. Lin. A study on SMO-type decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 17(4):893–908, 2006.

[11] T. Glasmachers and C. Igel. Maximum-gain working set selection for SVMs. *Journal of Machine Learning Research*, 7:1437–1466, 2006.

[12] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 169–184. MIT Press, Cambridge, MA, 1999.

[13] C.-W. Hsu and C.-J. Lin. A simple decomposition method for support vector machines. *Machine Learning*, 46(1–3):291–314, 2002.

[14] P. Laskov. Feasible direction decomposition algorithms for training support vector machines. *Machine Learning*, 46(1–3):315–349, 2002.

[15] D. Hush and C. Scovel. Polynomial-time decomposition algorithms for support vector machines. *Machine Learning*, 51(1):51–71, 2003.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2002.

[17] V. Kecman and I. Hadzic. Support vectors selection by linear programming. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, volume 5, pages 193–198, Como, Italy, 2000.

[18] W. Zhou, L. Zhang, and L. Jiao. Linear programming support vector machines. *Pattern Recognition*, 35(12):2927–2936, 2002.

[19] P. S. Bradley and O. L. Mangasarian. Massive data discrimination via linear support vector machines. *Optimization Methods and Software*, 13(1):1–10, 2000.

[20] Y. Torii and S. Abe. Fast training of linear programming support vector machines using decomposition techniques. In F. Schwenker and S. Marinai, editors, *Artificial Neural Networks in Pattern Recognition: Second IAPR Workshop, ANNPR 2006, Ulm, Germany, August/September 2006, Proceedings*, pages 165–176. Springer-Verlag, Berlin, Germany, 2006.

[21] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, New York, 1983.

[22] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98)*, pages 82–90, Madison, 1998.

[23] B. Schölkopf, P. Simard, A. Smola, and V. Vapnik. Prior knowledge in support vector kernels. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 640–646. MIT Press, Cambridge, MA, 1998.

[24] R. J. Vanderbei. *Linear Programming: Foundations and Extensions, second edition*. Kluwer Academic Publishers, Norwell, MA, 2001.

[25] C. Saunders, M. O. Stitson, J. Weston, L. Bottou, B. Schölkopf, and A. Smola. Support vector machine reference manual. Technical Report CSD-TR-98-03, Royal Holloway, University of London, London, 1998.

[26] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *National Academy of Sciences*, 96(12):6745–6750, 1999.

[27] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[28] I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, M. Raffeld, Z. Yakhini, A. Ben-Dor, E. Dougherty, J. Kononen, L. Bubendorf, W. Fehrl, S. Pittaluga, S. Gruvberger, N. Loman, O. Johannsson, H. Olsson, B. Wilfond, G. Sauter, O.-P. Kallioniemi, A. Borg, and J. Trent. Gene-expression profiles in hereditary breast cancer. *The New England Journal of Medicine*, 344(8):539–548, 2001.

[29] N. Iizuka, M. Oka, H. Yamada-Okabe, M. Nishida, Y. Maeda, N. Mori, T. Takao, T. Tamesa, A. Tangoku, and H. Tabuchi. Oligonucleotide microarray for prediction of early intrahepatic recurrence of hepatocellular carcinoma after curative resection. *The Lancet*, 361(9361):923–929, 2003.

[30] C. L. Nutt, D. R. Mani, R. A. Betensky, P. Tamayo, J. G. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M. E. McLaughlin, T. T. Batchelor, P. M. Black, A. von Deimling, S. L. Pomeroy, T. R. Golub, and D. N. Louis. Gene expression-based classification of malignant gliomas correlates better with survival than histological classification 1. *Cancer Research*, 63(7):1602–1607, 2003.

[31] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1(2):203–209, 2002.

[32] L. J. van't Veer, H. Y. Dai, M. J. van de Vijver, Y. D. D. He, A. A. M. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[33] H. Takenaga, S. Abe, M. Takatoo, M. Kayama, T. Kitamura, and Y. Okuyama. Input layer optimization of neural networks by sensitivity analysis and its application to recognition of numerals. *Electrical Engineering in Japan*, 111(4):130–138, 1991.

[34] A. Hashizume, J. Motoike, and R. Yabe. Fully automated blood cell differential system and its application. In *Proceedings of the IUPAC Third International Congress on Automation and New Technology in the Clinical Laboratory*, pages 297–302, Kobe, Japan, 1988.

[35] S. M. Weiss and I. Kapouleas. An empirical comparison of pattern recognition, neural nets, and machine learning classification methods. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 781–787, Detroit, 1989.

[36] S. Abe. *Pattern Classification: Neuro-Fuzzy Methods and Their Comparison*. Springer-Verlag, London, 2001.

[37] M.-S. Lan, H. Takenaga, and S. Abe. Character recognition using fuzzy rules extracted from data. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems*, volume 1, pages 415–420, Orlando, 1994.

[38] S. Abe. *Support Vector Machines for Pattern Classification*. Springer-Verlag, London, 2005.