<table>
<thead>
<tr>
<th>タイトル</th>
<th>Title</th>
<th>Is Primal Better Than Dual</th>
</tr>
</thead>
<tbody>
<tr>
<td>著者</td>
<td>Author(s)</td>
<td>Abe, Shigeo</td>
</tr>
<tr>
<td>掲載誌・巻号・ページ</td>
<td>Citation</td>
<td>Lecture Notes in Computer Science,5768 - Artificial Neural Networks - ICANN 2009:854-863</td>
</tr>
<tr>
<td>刊行日</td>
<td>Issue date</td>
<td>2009</td>
</tr>
<tr>
<td>資源タイプ</td>
<td>Resource Type</td>
<td>Journal Article / 学術雑誌論文</td>
</tr>
<tr>
<td>版区分</td>
<td>Resource Version</td>
<td>author</td>
</tr>
<tr>
<td>権利</td>
<td>Rights</td>
<td></td>
</tr>
<tr>
<td>DOI</td>
<td>10.1007/978-3-642-04274-4</td>
<td></td>
</tr>
<tr>
<td>JaLCDOI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>URL</td>
<td></td>
<td><a href="http://www.lib.kobe-u.ac.jp/handle_kernel/90001050">http://www.lib.kobe-u.ac.jp/handle_kernel/90001050</a></td>
</tr>
</tbody>
</table>

PDF issue: 2018-12-08
Abstract. Chapelle proposed to train support vector machines (SVMs) in the primal form by Newton’s method and discussed the advantages of training SVMs in the primal. In this paper we propose training L2 SVMs in the dual form in the similar way that Chapelle proposed. Namely, we solve the quadratic programming problem for the initial working set of training data by Newton’s method, delete from the working set the data with negative Lagrange multipliers as well as the data with the associated margins larger than or equal to 1, add to the working set training data with the associated margins less than 1, and repeat training the SVM until the working set does not change. The matrix associated with the dual quadratic form is positive definite while that with the primal quadratic form is positive semi-definite. And the former matrix requires less kernel evaluation. Computer experiments show that for most cases training the SVM by the proposed method is more stable and faster than training the SVM in the primal.

1 Introduction

In a support vector machine (SVM), the input space is mapped into a high-dimensional feature space, and since the mapping function is not explicitly treated by the kernel trick, usually the SVM is trained in the dual form. And many training methods have been developed [1–9]. But because the coefficient vector of the hyperplane is expressed by the kernel expansion, substituting the kernel expansion to the coefficient vector, the SVM in the primal form can be solvable. Based on this idea Chapelle [10] proposed the SVM in the primal form. Namely, starting from the initial working set selected from the training data, the associated primal quadratic programming program is solved by Newton’s method. Then the training data with the associated margins larger than or equal to 1 are deleted from the set, the training data with the margins smaller than 1 are added to the working set, and the above procedure is repeated until no deletion or addition is made to the working set. Chapelle discussed several advantages in training the SVM in the primal over in the dual.

In this paper we show that this is not the case for L2 SVMs. We propose training SVMs in the dual form in the similar way that Chapelle proposed training SVMs in the primal. Namely, starting from the initial working set, we solve
the associated dual quadratic programming problem, delete from the working set the data whose associated Lagrange multipliers are negative as well as the data with the associated margins larger than or equal to 1, add to the working set the data with the associated margins smaller than 1, and the above procedure is repeated until no data are added to, or deleted from, the working set. We call this method dual SVM and Chapelle’s primal SVM. We discuss the advantages of dual SVMs over primal SVM and by computer experiments demonstrate the effectiveness of the dual SVM.

In Section 2, we briefly summarize L1 and L2 SVMs, and in Section 3 we discuss primal SVMs. In Section 4, we compare primal and dual SVMs and clarify the advantages of dual SVMs and in Section 5, we demonstrate the advantages of dual SVMs over primal SVMs for some benchmark data sets.

2 Support Vector Machines

In this section, we summarize support vector machines for two-class problems. In training an SVM, we solve the following optimization problem:

\[
\begin{align*}
\text{minimize} & \quad Q(w, \xi) = \frac{1}{2}||w||^2 + \frac{C}{p} \sum_{i=1}^{M} \xi_i^p \\
\text{subject to} & \quad y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \quad \text{for} \quad i = 1, \ldots, M,
\end{align*}
\]

where \( w \) is the weight vector, \( \phi(x) \) is the mapping function that maps an \( m \)-dimensional input vector \( x \) into the feature space, \( b \) is the bias term, \( (x_i, y_i) \) \((i = 1, \ldots, M)\) are \( M \) training input-output pairs, with \( y_i = 1 \) if \( x_i \) belongs to Class 1, and \( y_i = -1 \) if Class 2, \( C \) is the margin parameter that determines the tradeoff between the maximization of the margin and minimization of the classification error, \( \xi_i \) are the nonnegative slack variables for \( x_i \), and \( p = 1 \) for an L1 SVM and \( p = 2 \) for an L2 SVM. We call the value of \( y_i (w^T \phi(x_i) + b) \) the margin for \( x_i \).

Introducing the Lagrange multipliers \( \alpha_i \), we obtain the following dual problem for the L1 SVM:

\[
\begin{align*}
\text{maximize} & \quad Q(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\
\text{subject to} & \quad \sum_{i=1}^{M} y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C \quad \text{for} \quad i = 1, \ldots, M,
\end{align*}
\]

and for the L2 SVM:

\[
\begin{align*}
\text{maximize} & \quad Q(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j \left( K(x_i, x_j) + \delta_{ij} \frac{C}{C} \right) \\
\text{subject to} & \quad \sum_{i=1}^{M} y_i \alpha_i = 0, \quad \alpha_i \geq 0 \quad \text{for} \quad i = 1, \ldots, M,
\end{align*}
\]
where \( K(\mathbf{x}, \mathbf{x}') \) is a kernel function that is given by \( K(\mathbf{x}, \mathbf{x}') = \phi^T(\mathbf{x}) \phi(\mathbf{x}') \).

In this study we use polynomial kernels with degree \( d \):
\[
K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x'} + 1)^d,
\]
and radial basis function (RBF) kernels:
\[
K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \| \mathbf{x} - \mathbf{x}' \|^2),
\]
where \( d \) is a positive integer and \( \gamma \) is a positive parameter for slope control.

The KKT complementarity conditions for the L1 SVM are given by
\[
\alpha_i \left( \sum_{j=1}^{M} y_j y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + y_i b - 1 + \xi_i \right) = 0 \quad \text{for} \quad i = 1, \ldots, M, \quad (7)
\]
\[
(C - \alpha_i) \xi_i = 0 \quad \alpha_i \geq 0, \quad \xi_i \geq 0 \quad \text{for} \quad i = 1, \ldots, M. \quad (8)
\]
For the solution of (3) and (4), if \( \alpha_i > 0 \), \( \mathbf{x}_i \) are called support vectors; especially if \( \alpha_i = C \), bounded support vectors and if \( 0 < \alpha_i < C \), unbounded support vectors.

The KKT complementarity conditions for the L2 SVM are given by
\[
\alpha_i \left( \sum_{j=1}^{M} y_j y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + y_i b - 1 + \frac{\alpha_i}{C} \right) = 0, \quad \alpha_i \geq 0 \quad \text{for} \quad i = 1, \ldots, M. \quad (9)
\]
Here, \( \alpha_i = C \xi_i \).

3 Training Support Vector Machines in the Primal

The optimization problem given by (1) and (2) is converted to the following optimization problem without constraints:

\[
\text{minimize} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{C}{p} \sum_{i=1}^{M} \max(0, 1 - y_i (\mathbf{w}^T \phi(\mathbf{x}_i) + b))^p \quad (10)
\]

Assume that \( \mathbf{w} \) is expressed by
\[
\mathbf{w} = \sum_{i=1}^{M} \beta_i \phi(\mathbf{x}_i), \quad (11)
\]
where \( \beta_i \) \( (i = 1, \ldots, M) \) are constants. Substituting (11) into (10), we obtain

\[
\text{minimize} \quad Q(\beta, b) = \frac{1}{2} \sum_{i,j=1}^{M} K(\mathbf{x}_i, \mathbf{x}_j) \beta_i \beta_j
\]
\[
+ \frac{C}{p} \sum_{i=1}^{M} \max \left( 0, 1 - y_i \left( \sum_{j=1}^{M} \beta_j K(\mathbf{x}_j, \mathbf{x}_i) + b \right) \right)^p. \quad (12)
\]

Let define a set of indices associated with the data that give the optimal solution for the L1 SVM:
\[
S = \{ i \mid y_i, D(\mathbf{x}_i) \leq 1 \text{ for } i \in \{1, \ldots, M\} \}, \quad (13)
\]
where \( D(\mathbf{x}) \) is the decision function and is given by \( D(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b \) and for the L2 SVM

\[
S = \{ i \mid y_i D(\mathbf{x}_i) < 1 \text{ for } i \in \{1, \ldots, M\} \}. \tag{14}
\]

Here equality does not hold because of \( \alpha_i = C \xi_i \).

We consider whether we can obtain the solution restricting the training data to the data associated with \( S \). Then (12) reduces to

\[
\text{minimize} \quad Q(\beta, b) = \frac{1}{2} \sum_{i,j \in S} K(\mathbf{x}_i, \mathbf{x}_j) \beta_i \beta_j + \frac{C}{p} \sum_{i \in S} \left( 1 - y_i \left( \sum_{j \in S} \beta_j K(\mathbf{x}_j, \mathbf{x}_i) + b \right) \right)^p. \tag{15}
\]

For the L1 SVM (\( p = 1 \)), from the KKT conditions (7) and (8), the slack variables \( \xi_i \) associated with unbounded support vectors \( \mathbf{x}_i \) (\( i \in S \)) are zero. But in (15), the sum of the slack variables is minimized. Thus, each constraint is not necessarily enforced to zero. In addition, because there is no quadratic term for \( b \), \( b \) cannot be determined by this formulation. Therefore, we cannot obtain the solution by solving (15). To solve this problem, Chappell used a Huber loss function, in which a linear loss is combined with a quadratic loss. Because this method gives the approximate solution for the L1 SVM, we do not consider solving L1 SVMs in the primal form.

For the L2 SVM, from (14) \( \xi_i \) associated with support vectors \( \mathbf{x}_i \) (\( i \in S \)) are positive. Thus (12) with the restriction of \( i \in S \) is equivalent to (15). Let \( \beta'_S = (\mathbf{\beta}'_S, b)^T \), where \( \mathbf{\beta}'_S = \{ \beta_i \mid i \in S \} \) and (15) be

\[
\text{minimize} \quad Q(\beta'_S) = \mathbf{c}'_S^T \beta'_S + \frac{1}{2} \beta'_S^T K_S \beta'_S, \tag{16}
\]

where \( K_S \) is the \((|S| + 1) \times (|S| + 1) \) matrix, \( \mathbf{c}_S \) is the \((|S| + 1) \)-dimensional vector, and

\[
\mathbf{c}_S = \frac{\partial Q(\beta'_S)}{\partial \alpha_i} = -C \sum_{k \in S} y_k K(\mathbf{x}_k, \mathbf{x}_i) \quad \text{for } i \in S, \tag{17}
\]

\[
\mathbf{c}_{Sb} = \frac{\partial Q(\beta'_S)}{\partial b} = -C \sum_{i \in S} y_i, \tag{18}
\]

\[
K_{Sij} = \frac{\partial^2 Q(\beta'_S)}{\partial \alpha_i \partial \alpha_j} = K(\mathbf{x}_i, \mathbf{x}_j) + C \sum_{k \in S} K(\mathbf{x}_i, \mathbf{x}_k) K(\mathbf{x}_k, \mathbf{x}_j) \quad \text{for } i, j \in S, \tag{19}
\]

\[
K_{Sib} = \frac{\partial^2 Q(\beta'_S)}{\partial \alpha_i \partial b} = C \sum_{j \in S} K(\mathbf{x}_i, \mathbf{x}_j) \quad \text{for } i \in S, \tag{20}
\]

\[
K_{Sbb} = \frac{\partial^2 Q(\beta'_S)}{\partial b^2} = C |S|. \tag{21}
\]
The optimal solution is obtained by
\[ \beta_S' = -K_S^{-1}c_S. \]  \hspace{1cm} (22)

Here, notice that \( K_S \) is positive semi-definite. If \( K_S \) is singular, usually a small value is added to the diagonal elements [10]. But because this will increase the number of support vectors, we delete data that cause singularity of the matrix from the working set.

4 Comparison of Training SVMs in the Primal and the Dual

4.1 Training in the Primal

The differences of the training method of the primal SVM and Chapelle’s method are that we start training with a small number of chunking data and we avoid singularity of \( K_S \) by deleting the associated row and column in decomposing \( K_S \) by the Cholesky factorization, instead of adding a small value to the diagonal elements of \( K_S \). We use the variable chunking algorithm. Namely, We solve (22) for the initial working set, delete, from the working set, the data with zero slack variables (the associated margins larger than or equal to 1), add to the working set the data with the positive slack variables (the associated margins smaller than 1), and solve (22), and repeat the above procedure until the same working set is obtained. Let the chunk size be \( h \), where \( h \) is a positive integer. Then the procedure is as follows:

1. Set \( h \) training data to the working set and go to Step 2.
2. Solve (22) for the working set by the Cholesky factorization. If the diagonal element is smaller than the prescribed value, delete the associated row and column, overwrite the column and row using the next data sample, and resume factorization and obtain \( \beta_S' \).
3. Delete from the working set the data with zero slack variables, namely \( x_i \) that satisfy \( y_i D(x_i) \geq 1 \). And add to the working set at most \( h \) most violating data, namely \( x_i \) that satisfy \( y_i D(x_i) < 1 \) from the smallest \( y_i D(x_i) \) in order. If the obtained working set is the same with the previous iteration, stop training. Otherwise, go to Step 2.

4.2 Training in the Dual

Similar to the primal SVM, we train the dual SVM. The idea is to eliminate the equality constraint (6) by solving it for one variable and substitute it into (5). Then the problem is reduced to the maximization problem with the positive constraints. We solve the sub-problem without considering the positive constraints, and delete negative variables from the working set. Other procedure is the same with that of the primal SVM.
Consider solving (5) and (6) for the index set $S$. Solving the equality constraint in (6) for $\alpha_s (s \in S)$, we obtain

$$\alpha_s = - \sum_{\substack{i \in S, \\ i \neq s}} y_s y_i \alpha_i.$$ \hfill (23)

Substituting (23) into (6), we obtain the following optimization problem

$$\begin{align*}
\text{maximize} & \quad Q(\alpha_S) = c_S^T \alpha_S' - \frac{1}{2} \alpha_S'^T K_S \alpha_S' \\
\text{subject to} & \quad \alpha_S \geq 0,
\end{align*}$$ \hfill (24)

where $\alpha_S' = \{ \alpha_i | i \neq s, i \in S \}$, $c_S$ is the $(|S| - 1)$-dimensional vector, $K_S$ is the $(|S| - 1) \times (|S| - 1)$ positive definite matrix, and

$$c_{si} = \frac{\partial Q(\alpha_S)}{\partial \alpha_i} = 1 - y_s y_i \quad \text{for} \quad i \neq s, \ i \in S$$ \hfill (26)

and

$$K_{sij} = \frac{\partial^2 Q(\alpha_W)}{\partial \alpha_i \partial \alpha_j} = y_i y_j (K(x_i, x_j) - K(x_s, x_i) - K(x_s, x_j)) + K(x_s, x_s) + \frac{\alpha_s + \alpha_i \delta_{ij}}{C} \quad \text{for} \quad i, j \neq s, \ i, j \in S.$$ \hfill (27)

Now the procedure for training the dual SVM is as follows.

1. Set $h$ training data to the working set and go to Step 2.
2. Solve $K_S \alpha_S' = c_S$ for $\alpha_S$ and using (23) obtain $\alpha_s$. Determine $b$ by

$$b = y_i - \sum_{j \in S} \alpha_j y_j \left( K(x_i, x_j) + \frac{\delta_{ij}}{C} \right) \quad \text{for} \quad i \in S.$$ \hfill (28)

3. Delete from the working set the data with negative variables, as well as the data $x_i$ that satisfy $y_i D(x_i) > 1$. And add to the working set at most $h$ most violating data, namely $x_i$ that satisfy $y_i D(x_i) < 1$ from the smallest $y_i D(x_i)$ in order. If the obtained working set is the same with the previous iteration, stop training. Otherwise, go to Step 2.

Because we ignore the positive constraints in solving $K_S \alpha_S' = c_S$ and delete negative variables afterward, the convergence of the above algorithm is not guaranteed.

### 4.3 Comparison

The differences of the dual SVM and the primal SVM are summarized as follows.

1. Matrix $K_S$ for the dual SVM is positive definite while that for the primal is positive semi-definite. Comparing (17)-(21) and (26), (27), $K_S$ and $c_S$ for the dual SVM require less kernel evaluations than the primal. Thus, the dual SVM will give more stable solution with less computation time.
2. Mapped support vectors for the primal SVM are interpreted as the independent data that span the empirical feature space [11]. Thus for the linear kernels the number of support vectors for the linear primal SVM is at most the number of the input variables. And any data can be support vectors so long as they span the empirical feature space.

3. Unlike the conventional training methods based on the decomposition technique such as SMO, training for primal and dual SVMs does not guarantee monotonic convergence or may not converge. This is because there is no guarantee that the objective function is monotonic as the iteration proceeds. Thus, to avoid divergence, we may need to incorporate acceleration mechanism such as developed for linear programming SVMs [12].

5 Performance Evaluation

We evaluated performance of the proposed dual SVM with that of the primal SVM using the benchmark data sets shown in Table 1, which lists the number of inputs, classes, training data, and test data. The table also shows the parameter values for L2 SVMs determined by fivefold cross-validation. For instance, \(d_4\) and \(\gamma_{10}\) mean that the kernels are polynomial kernel with degree 4 and RBF kernels with \(\gamma = 10\), and \(C_{10^5}\) means that the value of the margin parameter is \(10^5\).

We used fuzzy one-against-all SVMs [13] and measured the training time using a personal computer (3GHz, 2GB memory, Windows XP operating system). As in [10], we prepared a cache memory with the size equal to the kernel matrix.

<table>
<thead>
<tr>
<th>Data</th>
<th>Inputs</th>
<th>Classes</th>
<th>Train.</th>
<th>Test.</th>
<th>Parm.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thyroid [14]</td>
<td>21</td>
<td>3</td>
<td>3,772</td>
<td>3,428</td>
<td>(d_4C_{10^5})</td>
</tr>
<tr>
<td>Blood cell [13]</td>
<td>13</td>
<td>12</td>
<td>3,097</td>
<td>3,100</td>
<td>(\gamma_{10}C_{100})</td>
</tr>
<tr>
<td>Hiragana-50 [13]</td>
<td>50</td>
<td>39</td>
<td>4,610</td>
<td>4,610</td>
<td>(\gamma_{10}C_{1000})</td>
</tr>
<tr>
<td>Hiragana-105 [13]</td>
<td>105</td>
<td>38</td>
<td>8.375</td>
<td>8.356</td>
<td>(\gamma_{10}C_{10^4})</td>
</tr>
<tr>
<td>Satimage [14]</td>
<td>36</td>
<td>6</td>
<td>4,435</td>
<td>2,000</td>
<td>(\gamma_{200}C_{10})</td>
</tr>
<tr>
<td>USPS [15]</td>
<td>256</td>
<td>10</td>
<td>7,291</td>
<td>2,007</td>
<td>(\gamma_{10}C_{100})</td>
</tr>
</tbody>
</table>

Table 2 shows the effect of the chunk size on the performance of primal and dual SVMs for the USPS data set. In the table “Chunk,” “SVs,” “Iterations,” “Kernels,” “Rec.,” and “Time” denote, respectively, the chunk size, the average number of support vectors per one decision function, the number of iterations, the total number of kernel accesses, the recognition rate of the test (training) data set, and training time. “Kernels accesses” means that kernel values are provided through the cache memory if they are in the cache memory, and af-
ter evaluation if they are not. For “SVs,” “Iterations,” “kernels,” and “Time” columns, the better value is shown in boldface between dual and primal SVMs.

From the table, the numbers of support vectors for the dual SVM are the same for the five cases, but for the primal SVM, the number increases as the chunk size increases. Although the number of iterations for the primal SVM is smaller, the number of kernel accesses is smaller and training time is shorter for the dual SVM. This means that the computation burden per iteration for the primal SVM is larger as discussed previously. For the primal SVM, the result did not change even if we added the small value to the diagonal elements.

Table 2. Effect of the chunk size for primal and dual SVMs for the USPS data set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Chunk</th>
<th>SVs</th>
<th>Iterations</th>
<th>Kernels</th>
<th>Rec.</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dual</td>
<td>10</td>
<td>597</td>
<td>68</td>
<td>1,889,358,476</td>
<td>95.47 (99.99)</td>
<td>253</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>597</td>
<td>19</td>
<td>593,529,638</td>
<td>95.47 (100)</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>597</td>
<td>15</td>
<td>585,495,205</td>
<td>95.47 (99.99)</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>597</td>
<td>26</td>
<td>1,926,210,878</td>
<td>95.47 (99.99)</td>
<td>650</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>597</td>
<td>26</td>
<td>2,627,126,816</td>
<td>95.47 (99.99)</td>
<td>1527</td>
</tr>
<tr>
<td>Primal</td>
<td>10</td>
<td>597</td>
<td>66</td>
<td>28,647,597,296</td>
<td>95.47 (99.99)</td>
<td>1,437</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>604</td>
<td>16</td>
<td>8,116,273,966</td>
<td>95.47 (99.99)</td>
<td>412</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>604</td>
<td>11</td>
<td>6,928,602,363</td>
<td>95.42 (99.99)</td>
<td>385</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>724</td>
<td>7</td>
<td>33,628,904,338</td>
<td>95.47 (99.99)</td>
<td>1,826</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>724</td>
<td>7</td>
<td>103,084,629,720</td>
<td>95.47 (99.99)</td>
<td>5,074</td>
</tr>
</tbody>
</table>

We examined performance of the case where singularity of the matrix occurs. Table 3 shows the result for the blood cell data set for the linear kernel and $C = 100$. For the primal SVM we also include the results when the diagonal elements are added with 0.00001 denoted as Primal (added). The numeral in the parentheses in the SVs column shows the working set size after training. Thus, for example, for the chunk size of 50, among 497 data only 15 data are support vectors and the remaining data are deleted because of singularity of the matrix.

From the table, training of the dual SVM was the fastest for all the chunk sizes. Comparing the results for the primal SVM with the primal SVM (added), training is faster for the primal SVM (added) but the solutions are different and the number of support vectors increased as the number of the chunk size was increased. The results clearly show that the addition of a small positive value to the diagonal elements is not a good strategy for avoiding the singularity of the matrix.

Table 4 lists the results for the primal and dual SVMs using the benchmark data sets. We set the chunk size of 50 for all the cases. For the thyroid data set, training of the dual SVM was very slow. And for the primal SVM the working set size fluctuated considerably and the training did not converge within 10,000 iterations. Except for hiragana-13 and hiragana-105 data sets, the dual SVM was
Table 3. Comparison of primal and dual SVMs for the blood cell data set with $d = 1$ and $C = 100$.

<table>
<thead>
<tr>
<th>Method</th>
<th>Chunk</th>
<th>SVs</th>
<th>Iterations</th>
<th>Kernels</th>
<th>Rec.</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual</td>
<td>10</td>
<td>497</td>
<td>55</td>
<td>1,043,872,194</td>
<td>88.97 (91.99)</td>
<td>212</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>497</td>
<td>17</td>
<td>326,186,052</td>
<td>88.97 (91.99)</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>497</td>
<td>13</td>
<td>274,730,520</td>
<td>88.97 (91.99)</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>497</td>
<td>13</td>
<td>561,872,500</td>
<td>88.97 (91.99)</td>
<td>252</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>15 (497)</td>
<td>78</td>
<td>110,167,253,737</td>
<td>88.97 (91.99)</td>
<td>4,735</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>15 (497)</td>
<td>20</td>
<td>27,464,672,230</td>
<td>88.97 (91.99)</td>
<td>1,203</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>16 (497)</td>
<td>14</td>
<td>19,333,796,617</td>
<td>88.97 (91.99)</td>
<td>848</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>16 (497)</td>
<td>9</td>
<td>36,858,682,541</td>
<td>88.97 (91.99)</td>
<td>1,753</td>
</tr>
<tr>
<td>Primal</td>
<td>10</td>
<td>498</td>
<td>53</td>
<td>53,685,575,039</td>
<td>88.97 (91.99)</td>
<td>2,387</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>517</td>
<td>14</td>
<td>14,051,320,440</td>
<td>88.97 (91.99)</td>
<td>644</td>
</tr>
<tr>
<td>(Added)</td>
<td>100</td>
<td>545</td>
<td>9</td>
<td>10,042,016,165</td>
<td>88.90 (91.73)</td>
<td>475</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>1,073</td>
<td>5</td>
<td>27,336,218,781</td>
<td>87.45 (89.60)</td>
<td>1,287</td>
</tr>
</tbody>
</table>

faster and except for the hiragana-50 data set the number of support vectors for the dual SVM was smaller.

From our experiments it is clear that the dual SVM is better than the primal SVM from the standpoints of stable convergence and fast training.

6 Conclusions

In this paper we proposed a new training method for SVMs in the dual form (dual SVMs). Namely, starting with an initial working set, we solve the subproblem expressed in the dual form by Newton’s method, delete the data if the associated Lagrange multipliers are non-positive as well as the data with zero slack variables, add the data with positive slack variables, and repeat solving the subproblem until the same working set is obtained. We clarified the advantages of the dual SVM over the primal SVM from the standpoint of positive definiteness of the matrix. By computer experiments we show that training of the dual SVMs was faster than that of the primal SVMs and usually the number of support vectors was smaller. We also showed that adding a small value to the diagonal element for the primal SVMs makes a larger number of support vectors and gives different solutions for the change of the chunk size.

References

## Table 4. Comparison of primal and dual SVMs for the benchmark data sets

<table>
<thead>
<tr>
<th>SVM</th>
<th>Data</th>
<th>SVs</th>
<th>Iterations</th>
<th>Kernels</th>
<th>Rec. (Time)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thyroid</td>
<td>98</td>
<td>1439</td>
<td>2,660,742,540</td>
<td>97.81 (100)</td>
<td><strong>220</strong></td>
<td></td>
</tr>
<tr>
<td>Blood cell</td>
<td>188</td>
<td>13</td>
<td>86,324,885</td>
<td>93.58 (97.19)</td>
<td><strong>10</strong></td>
<td></td>
</tr>
<tr>
<td>Dual</td>
<td>H-50</td>
<td>77</td>
<td>386,391,968</td>
<td>99.28 (100)</td>
<td><strong>50</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H-13</td>
<td>39</td>
<td>2,343,986,783</td>
<td>99.55 (100)</td>
<td>282</td>
<td></td>
</tr>
<tr>
<td></td>
<td>H-105</td>
<td>91</td>
<td>812,259,183</td>
<td>100 (100)</td>
<td>147</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Satimage</td>
<td>1001</td>
<td>260,106,924</td>
<td>91.70 (99.71)</td>
<td><strong>157</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>USPS</td>
<td>597</td>
<td>593,529,638</td>
<td>95.47 (99.99)</td>
<td><strong>114</strong></td>
<td></td>
</tr>
<tr>
<td>Primal</td>
<td>Thyroid</td>
<td>203</td>
<td>445,319,582</td>
<td>93.61 (97.19)</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blood cell</td>
<td>70 (78)</td>
<td>605,637,116</td>
<td>99.28 (100)</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>H-50</td>
<td>99</td>
<td>749,712,635</td>
<td>99.70 (99.96)</td>
<td><strong>93</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>H-105</td>
<td>111</td>
<td>907,360,383</td>
<td>100 (100)</td>
<td><strong>140</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Satimage</td>
<td>1006</td>
<td>26,125,955,619</td>
<td>91.70 (99.71)</td>
<td>1258</td>
<td></td>
</tr>
<tr>
<td></td>
<td>USPS</td>
<td>604</td>
<td>8,116,273,966</td>
<td>95.47 (99.99)</td>
<td><strong>412</strong></td>
<td></td>
</tr>
</tbody>
</table>


