A Study on L2-Loss (Squared Hinge-Loss) Multi-Class SVM

Ching-Pei Lee and Chih-Jen Lin

Department of Computer Science, National Taiwan University, Taipei 10617, Taiwan

Keywords: Support vector machines, Multi-class classification, Squared hinge loss, L2 loss.

Abstract

Crammer and Singer's method is one of the most popular multi-class SVMs. It considers L1 loss (hinge loss) in a complicated optimization problem. In SVM, squared hinge loss (L2 loss) is a common alternative to L1 loss, but surprisingly we have not seen any paper studying details of Crammer and Singer's method using L2 loss. In this note, we conduct a thorough investigation. We show that the derivation is not trivial and has some subtle differences from the L1 case. Details provided in this work can be a useful reference for those who intend to use Crammer and Singer's method with L2 loss. They do not need a tedious process to derive everything by themselves. Further, we present some new results/discussion for both L1- and L2-loss formulations.

1 Introduction

Support Vector Machines (SVM) (Boser et al., 1992; Cortes and Vapnik, 1995) were originally designed for binary classification. In recent years, many approaches have been proposed to extend SVM to handle multi-class classification problems; see, for example, a detailed comparison in Hsu and Lin (2002). Among these works, the method proposed by Crammer and Singer (2001, 2002) has been widely used. They extend the optimization problem of L1-loss (hinge-loss) SVM to a multi-class formula. In binary classification, squared hinge loss (L2 loss) is a common alternative to L1 loss, but surprisingly we have not found any paper studying details of Crammer and Singer's method with L2 loss. This is inconvenient because, for example, we do not know what the dual problem is.[1] Although the dual problem of two-class SVM using L2 loss is well known, it cannot be directly extended to the multi-class case. In fact, the derivation is non-trivial and has some subtle differences from the L1 case. Also, the algorithm to solve the dual problem (for both kernel and linear situations) must be modified. We think there is a need to give all the details for future references. Then those who intend to use Crammer and

[1] Indeed, the dual problem has been provided in some works of structured SVM, where Crammer and Singer's multi-class SVM is a special case. However, their form can be simplified for multi-class SVM; see a detailed discussion in Section 2.2.

Singer's method with L2 loss do not need a tedious procedure to derive everything by themselves.

In addition to the main contribution to investigate L2-loss multi-class SVM, we present some new results for both L1- and L2-loss cases. First, we discuss the differences between the dual problem of Crammer and Singer's multi-class SVM and that of structured SVM, although we focus more on the comparison of L2-loss formulation. Second, we give a simpler derivation for solving the sub-problem in the decomposition method to minimize the dual problem.

This paper is organized as follows. In Section 2, we introduce L2-loss multi-class SVM and derive its dual problem. We discuss the connection to structured SVM, which is a generalization of Crammer and Singer's multi-class SVM. Then in Section 3, we extend a decomposition method to solve the optimization problem. In particular, we obtain the sub-problem to be solved at each iteration. A procedure to find the solution of the sub-problem is given in Section 4. Our derivation and proof are simpler than Crammer and Singer's. In Section 5, we discuss some implementation issues and extensions. Experiments in Section 6 compare the performance of L1-loss and L2-loss multi-class SVM using both linear and nonlinear kernels. Section 7 then concludes this paper.

2 Formulation

Given a set of instance-label pairs $(\boldsymbol{x}_i, y_i),\ \boldsymbol{x}_i \in R^n,\ y_i \in \{1, \dots, k\},\ i = 1, \dots, l$, Crammer and Singer (2002) proposed a multi-class SVM approach by solving the following optimization problem.

$$\begin{aligned} \min_{\boldsymbol{w}_1,\dots,\boldsymbol{w}_k,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{m=1}^{k} \boldsymbol{w}_m^T \boldsymbol{w}_m + C\sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & \boldsymbol{w}_{y_i}^T \boldsymbol{x}_i - \boldsymbol{w}_m^T \boldsymbol{x}_i \geq e_i^m - \xi_i, \\ & i = 1, \dots, l,\ m = 1, \dots, k, \end{aligned} \tag{1}$$

where

$$e_i^m = \begin{cases} 0 & \text{if } y_i = m, \\ 1 & \text{if } y_i \neq m. \end{cases} \tag{2}$$

Note that if $y_i = m$, the constraint is the same as $\xi_i \geq 0$.

The decision function for predicting the label of an instance $\boldsymbol{x}$ is

$$\arg\max_{m=1,\dots,k} \boldsymbol{w}_m^T \boldsymbol{x}.$$

The dual problem of (1) is

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{m=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l} K_{i,j}\alpha_i^m\alpha_j^m + \sum_{i=1}^{l}\sum_{m=1}^{k}\alpha_i^m e_i^m \\
\text{subject to} \quad & \sum_{m=1}^{k}\alpha_i^m = 0,\ i = 1,\dots,l, && (3)\\
& \alpha_i^m \le 0,\ i = 1,\dots,l,\ m = 1,\dots,k,\ m \ne y_i, && (4)\\
& \alpha_i^{y_i} \le C,\ i = 1,\dots,l, && (5)
\end{aligned}$$

where

$$\boldsymbol{\alpha} = [\alpha_1^1,\dots,\alpha_1^k,\dots,\alpha_l^1,\dots,\alpha_l^k]^T \quad \text{and} \quad K_{i,j} = \boldsymbol{x}_i^T\boldsymbol{x}_j. \tag{6}$$

Constraints (4) and (5) are often combined as

$$\alpha_i^m \le C_{y_i}^m,\ \text{where } C_{y_i}^m = \begin{cases} 0 & \text{if } y_i \ne m,\\ C & \text{if } y_i = m. \end{cases}$$

We separate them in order to compare with the dual problem of using L2 loss. After solving (3), one can compute the optimal $\boldsymbol{w}_m$ by

$$\boldsymbol{w}_m = \sum_{i=1}^{l}\alpha_i^m\boldsymbol{x}_i,\ m = 1,\dots,k. \tag{7}$$

In this paper, we extend problem (1) to use L2 loss. By changing the loss term from ξ_i to ξ_i^2, the primal problem becomes

$$\begin{aligned}
\min_{\boldsymbol{w}_1,\dots,\boldsymbol{w}_k,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{m=1}^{k}\boldsymbol{w}_m^T\boldsymbol{w}_m + C\sum_{i=1}^{l}\xi_i^2 \\
\text{subject to} \quad & \boldsymbol{w}_{y_i}^T\boldsymbol{x}_i - \boldsymbol{w}_m^T\boldsymbol{x}_i \ge e_i^m - \xi_i, && (8)\\
& i = 1,\dots,l,\ m = 1,\dots,k.
\end{aligned}$$

The constraint $\boldsymbol{w}_{y_i}^T\boldsymbol{x}_i - \boldsymbol{w}_m^T\boldsymbol{x}_i \ge e_i^m - \xi_i$ when $m = y_i$ can be removed because for L2 loss, $\xi_i \ge 0$ holds at an optimum without this constraint. We keep it here in order to compare with the formulation of using L1 loss.

We will derive the following dual problem.

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \quad & f(\boldsymbol{\alpha}) \\
\text{subject to} \quad & \sum_{m=1}^{k}\alpha_i^m = 0,\ i = 1,\dots,l, && (9)\\
& \alpha_i^m \le 0,\ i = 1,\dots,l,\ m = 1,\dots,k,\ m \ne y_i,
\end{aligned}$$

where

$$f(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{m=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l} K_{i,j}\alpha_i^m\alpha_j^m + \sum_{i=1}^{l}\sum_{m=1}^{k}\alpha_i^m e_i^m + \sum_{i=1}^{l}\frac{(\alpha_i^{y_i})^2}{4C}.$$

Problem (9) is similar to (3), but it possesses an additional quadratic term in the objective function. Further, the constraint on $\alpha_i^{y_i}$ is different. In (5), $\alpha_i^{y_i} \le C$, but in (9), $\alpha_i^{y_i}$ is unconstrained.[2]

We discuss two methods to derive the dual problem. The first is from a direct calculation, while the second follows from the derivation of structured SVM.

2.1 A Direct Calculation to Obtain the Dual Problem

The Lagrange function of (8) is

$$\begin{aligned} L(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, \boldsymbol{\xi}, \hat{\boldsymbol{\alpha}}) = \frac{1}{2} \sum_{m=1}^{k} \boldsymbol{w}_m^T \boldsymbol{w}_m + C \sum_{i=1}^{l} \xi_i^2 \\ - \sum_{i=1}^{l} \sum_{m=1}^{k} \hat{\alpha}_i^m (\boldsymbol{w}_{y_i}^T \boldsymbol{x}_i - \boldsymbol{w}_m^T \boldsymbol{x}_i - e_i^m + \xi_i), \end{aligned}$$

where $\hat{\alpha}_i^m \ge 0,\ m = 1, \dots, k,\ i = 1, \dots, l$, are Lagrange multipliers.

The Lagrange dual problem is

$$\max_{\hat{\boldsymbol{\alpha}}:\ \hat{\alpha}_i^m \ge 0,\ \forall i,m} \big(\inf_{\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, \boldsymbol{\xi}} L(\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, \boldsymbol{\xi}, \hat{\boldsymbol{\alpha}}) \big).$$

To minimize L under fixed $\hat{\boldsymbol{\alpha}}$, we rewrite the following term in the Lagrange function

$$\sum_{i=1}^{l} \sum_{m=1}^{k} \hat{\alpha}_i^m \boldsymbol{w}_{y_i}^T \boldsymbol{x}_i = \sum_{m=1}^{k} \sum_{i:y_i=m} \sum_{s=1}^{k} \hat{\alpha}_i^s \boldsymbol{w}_{y_i}^T \boldsymbol{x}_i = \sum_{m=1}^{k} \boldsymbol{w}_m^T \sum_{i=1}^{l} (1 - e_i^m) \sum_{s=1}^{k} \hat{\alpha}_i^s \boldsymbol{x}_i, \quad (10)$$

and have

$$\nabla_{\boldsymbol{w}_m} L = 0 \quad \Rightarrow \quad \boldsymbol{w}_m^* - \sum_{i=1}^{l} \big((1 - e_i^m) \sum_{s=1}^{k} \hat{\alpha}_i^s - \hat{\alpha}_i^m \big) \boldsymbol{x}_i = 0,\ m = 1, \dots, k, \quad (11)$$

$$\nabla_{\xi_i} L = 2C\xi_i - \sum_{m=1}^{k} \hat{\alpha}_i^m = 0 \quad \Rightarrow \quad \xi_i^* = \frac{\sum_{m=1}^{k} \hat{\alpha}_i^m}{2C},\ i = 1, \dots, l. \quad (12)$$

We simplify (11) by defining

$$\alpha_i^m \equiv (1 - e_i^m) \sum_{s=1}^{k} \hat{\alpha}_i^s - \hat{\alpha}_i^m,\ i = 1, \dots, l,\ m = 1, \dots, k. \quad (13)$$

This definition is equivalent to

$$\alpha_i^m = -\hat{\alpha}_i^m,\ \forall m \ne y_i, \quad (14)$$

$$\alpha_i^{y_i} = \sum_{m:m \ne y_i} \hat{\alpha}_i^m = - \sum_{m:m \ne y_i} \alpha_i^m. \quad (15)$$

[2]Indeed, using $\sum_{m=1}^{k} \alpha_i^m = 0$ and $\alpha_i^m \le 0,\ \forall m \ne y_i$, we have $\alpha_i^{y_i} \ge 0$ for both dual problems of L1 and L2 cases.

Therefore, we can rewrite the solution of minimizing L under fixed $\hat{\boldsymbol{\alpha}}$ as

$$\boldsymbol{w}_m^* = \sum_{i=1}^{l} \alpha_i^m \boldsymbol{x},\ m = 1, \ldots, k, \tag{16}$$

$$\xi_i^* = \frac{\hat{\alpha}_i^{y_i} + \alpha_i^{y_i}}{2C},\ i = 1, \ldots, l. \tag{17}$$

By (2), (10), (11), and (14)-(17), the Lagrange dual function is

$$\begin{aligned}
& L(\boldsymbol{w}_1^*, \ldots, \boldsymbol{w}_k^*, \boldsymbol{\xi}^*, \hat{\boldsymbol{\alpha}}) \\
= & \frac{1}{2}\sum_{m=1}^{k}(\boldsymbol{w}_m^*)^T\boldsymbol{w}_m^* - \sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m\left((\boldsymbol{w}_{y_i}^*)^T\boldsymbol{x}_i - (\boldsymbol{w}_m^*)^T\boldsymbol{x}_i\right) \\
& - C\sum_{i=1}^{l}(\xi_i^*)^2 + \sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m e_i^m \qquad (18) \\
= & \frac{1}{2}\sum_{m=1}^{k}(\boldsymbol{w}_m^*)^T\boldsymbol{w}_m^* - \Big(\sum_{m=1}^{k}(\boldsymbol{w}_m^*)^T\sum_{i=1}^{l}(1 - e_i^m)\sum_{s=1}^{k}\hat{\alpha}_i^s\boldsymbol{x}_i \\
& - \sum_{m=1}^{k}(\boldsymbol{w}_m^*)^T\sum_{i=1}^{l}\hat{\alpha}_i^m\boldsymbol{x}_i\Big) - C\sum_{i=1}^{l}(\xi_i^*)^2 + \sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m e_i^m \\
= & -\frac{1}{2}\sum_{m=1}^{k}(\boldsymbol{w}_m^*)^T\boldsymbol{w}_m^* - C\sum_{i=1}^{l}(\xi_i^*)^2 + \sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m e_i^m \\
= & -\frac{1}{2}\sum_{m=1}^{k}\|\sum_{i=1}^{l}\alpha_i^m\boldsymbol{x}_i\|^2 - \sum_{i=1}^{l}\frac{(\hat{\alpha}_i^{y_i} + \alpha_i^{y_i})^2}{4C} - \sum_{i=1}^{l}\sum_{m=1}^{k}\alpha_i^m e_i^m \\
= & -\frac{1}{2}\sum_{m=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l}K_{i,j}\alpha_i^m\alpha_j^m - \sum_{i=1}^{l}\sum_{m=1}^{k}\alpha_i^m e_i^m - \sum_{i=1}^{l}\frac{(\hat{\alpha}_i^{y_i} + \alpha_i^{y_i})^2}{4C}. \qquad (19)
\end{aligned}$$

Because $\hat{\alpha}_i^m,\ \forall m \neq y_i$ do not appear in (19), from (14) and (15), the dual problem is

$$\begin{aligned}
\min_{\boldsymbol{\alpha},\hat{\boldsymbol{\alpha}}}\quad & \frac{1}{2}\sum_{m=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l}K_{i,j}\alpha_i^m\alpha_j^m + \sum_{i=1}^{l}\sum_{m=1}^{k}\alpha_i^m e_i^m + \sum_{i=1}^{l}\frac{(\hat{\alpha}_i^{y_i} + \alpha_i^{y_i})^2}{4C}. \\
\text{subject to}\quad & \hat{\alpha}_i^{y_i} \geq 0,\ i = 1, \ldots, l, \\
& \sum_{m=1}^{k}\alpha_i^m = 0,\ i = 1, \ldots, l, \qquad (20) \\
& \alpha_i^m \leq 0,\ i = 1, \ldots, l,\ m = 1, \ldots, k,\ m \neq y_i. \qquad (21)
\end{aligned}$$

Because (20) and (21) imply $\alpha_i^{y_i} \geq 0$ and $\hat{\alpha}_i^{y_i}$ appears only in the term $(\hat{\alpha}_i^{y_i} + \alpha_i^{y_i})^2$, the optimal $\hat{\alpha}_i^{y_i}$ must be zero. After removing $\hat{\alpha}_i^{y_i}$, the derivation of the dual problem is complete.

We discuss the difference from L1 loss. If L1 loss is used, (12) becomes

$$\sum_{m=1}^{k} \hat{\alpha}_i^m = C, \tag{22}$$

and $-C\sum_{i=1}^{l}(\xi_i^*)^2$ in (18) disappears. Equation (22) and the fact $\hat{\alpha}_i^{y_i} \geq 0$ lead to the constraint (5) as follows.

$$\alpha_i^{y_i} = \sum_{m:m\neq y_i} \hat{\alpha}_i^m = C - \hat{\alpha}_i^{y_i} \leq C.$$

For L2 loss, without the condition in (22), $\alpha_i^{y_i}$ is unconstrained.

2.2 Using Structured SVM Formulation to Obtain the Dual Problem

It is well known that Crammer and Singer's multi-class SVM is a special case of structured SVM (Tsochantaridis et al., 2005). By defining

$$\boldsymbol{w} \equiv \begin{bmatrix} \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_k \end{bmatrix} \in R^{kn\times 1} \quad \text{and} \quad \delta(i,m) \in R^{kn\times 1},$$

with

$$\delta(i,m) \equiv \begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}_i \\ \mathbf{0} \\ -\boldsymbol{x}_i \\ \mathbf{0} \end{bmatrix} \begin{matrix} \\ \leftarrow y_i\text{-th position} \\ \\ \leftarrow m\text{-th position} \\ \end{matrix}, \text{ if } y_i \neq m, \text{ and } \mathbf{0} \text{ if } y_i = m,$$

problem (8) can be written as

$$\begin{aligned} \min_{\boldsymbol{w},\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i^2 \\ \text{subject to} \quad & \boldsymbol{w}^T\delta(i,m) \geq e_i^m - \xi_i, \\ & i = 1,\ldots,l,\ m = 1,\ldots,k. \end{aligned} \tag{23}$$

This problem is in a similar form to L2-loss binary SVM, so the derivation of the dual problem is straight forward. Following Tsochantaridis et al. (2005), the dual problem is[3]

[3]Problem (24) is slightly different from that in Tsochantaridis et al. (2005) because they remove the constraints $\xi_i \geq 0,\ \forall i$ by setting $m \neq y_i$ in (23).

$$\begin{aligned}\min_{\hat{\boldsymbol{\alpha}}} \quad & \frac{1}{2}\sum_{m=1}^{k}\sum_{s=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l}\delta(i,m)^T\delta(j,s)\hat{\alpha}_i^m\hat{\alpha}_j^s \\ & -\sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m e_i^m + \sum_{i=1}^{l}\frac{(\sum_{m=1}^{k}\hat{\alpha}_i^m)^2}{4C} \\ \text{subject to} \quad & \hat{\alpha}_i^m \geq 0, \quad i=1,\ldots,l,\ m=1,\ldots,k.\end{aligned} \tag{24}$$

Also, at an optimal solution, we have

$$\boldsymbol{w} = \sum_{i=1}^{l}\sum_{m=1}^{k}\hat{\alpha}_i^m\delta(i,m) \quad \text{and} \quad \xi_i = \frac{\sum_{m=1}^{k}\hat{\alpha}_i^m}{2C}. \tag{25}$$

Problem (24) seems to be very different from problem (9) obtained in Section 2.1. In fact, problem (24) is an intermediate result in our derivation. A careful check shows

1. $\hat{\boldsymbol{\alpha}}$ is the same as the Lagrange multiplier used in Section 2.1.
2. $\boldsymbol{w}$ in (25) is the same as that in (7); see Equation (11).

In Section 2.1, we introduce a new variable $\boldsymbol{\alpha}$ and simplifies the two terms

$$\sum_{m=1}^{k}\sum_{s=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l}\delta(i,m)^T\delta(j,s)\hat{\alpha}_i^m\hat{\alpha}_j^s \quad \text{and} \quad \sum_{i=1}^{l}(\sum_{m=1}^{k}\hat{\alpha}_i^m)^2$$

to

$$\sum_{m=1}^{k}\sum_{i=1}^{l}\sum_{j=1}^{l}K_{i,j}\alpha_i^m\alpha_j^m \quad \text{and} \quad \sum_{i=1}^{l}\frac{(\alpha_i^{y_i})^2}{4C},$$

respectively. An advantage of problem (9) is that $K_{i,j} = \boldsymbol{x}_i^T\boldsymbol{x}_j$ explicitly appears in the objective function. In contrast, $\delta(i,m)^T\delta(j,m)$ does not reveal details of the inner product between instances. However, a caveat of (9) is that it contains some linear constraints.

An interesting question is whether the simplification from (24) to (9) allows us to apply a simpler or more efficient optimization algorithm. This issue already occurs for using L1 loss because we can either solve problem (3) or a form similar to (24). However, the dual problem of L1-loss structured SVM contains a linear constraint, but problem (24) does not.[4] Therefore, for the L1 case, it is easy to see that the simplified form (3) should be used. However, for L2 loss, problem (24) possesses an advantage of being a bound-constrained problem. We will give some discussion about solving (9) or (24) in Section 5.5. In all remaining places we focus on problem (9) because existing implementations for the L1-loss formulation all solve the corresponding problem (3).

[4] See Proposition 5 in Tsochantaridis et al. (2005).

3 Decomposition Method and Sub-problem

Decomposition methods are currently the major method to solve the dual problem (3) of the L1 case (Crammer and Singer, 2002; Keerthi et al., 2008). At each iteration, the k variables $\alpha_i^1, \ldots, \alpha_i^k$ associated with an instance $\boldsymbol{x}_i$ are selected for updating, while other variables are fixed. For (3), the following sub-problem is solved.

$$\begin{aligned} \min_{\alpha_i^1,\ldots,\alpha_i^k} \quad & \sum_{m=1}^{k} \left(\frac{1}{2} A(\alpha_i^m)^2 + B_m \alpha_i^m\right) \\ \text{subject to} \quad & \sum_{m=1}^{k} \alpha_i^m = 0, \\ & \alpha_i^m \le C_{y_i}^m,\ m = 1, \ldots, k, \end{aligned} \tag{26}$$

where

$$A = K_{i,i} \quad \text{and} \quad B_m = \sum_{j=1}^{l} K_{j,i} \bar{\alpha}_j^m + e_i^m - A\bar{\alpha}_i^m. \tag{27}$$

In (27), $\bar{\boldsymbol{\alpha}}$ is the solution obtained in the previous iteration. We defer the discussion on the selection of the index i in Section 5.

For problem (9), we show that the sub-problem is:

$$\begin{aligned} \min_{\alpha_i^1,\ldots,\alpha_i^k} \quad & \sum_{m=1}^{k} \left(\frac{1}{2} A(\alpha_i^m)^2 + B_m \alpha_i^m\right) + \frac{(\alpha_i^{y_i})^2}{4C} \\ \text{subject to} \quad & \sum_{m=1}^{k} \alpha_i^m = 0, \\ & \alpha_i^m \le 0,\ m = 1, \ldots, k,\ m \ne y_i, \end{aligned} \tag{28}$$

where A and B_m are the same as (27). The derivation of (28) is as follows.

Because all elements except $\alpha_i^1, \ldots, \alpha_i^k$ are fixed, the objective function of (9) becomes

$$\begin{aligned} & \sum_{m=1}^{k} \frac{1}{2}\big(K_{i,i}(\alpha_i^m)^2 + 2\sum_{j:j\ne i} K_{i,j}\bar{\alpha}_j^m \alpha_i^m\big) + \sum_{m=1}^{k} \alpha_i^m e_i^m + \frac{(\alpha_i^{y_i})^2}{4C} + \text{constants} \\ = & \sum_{m=1}^{k} \big(\frac{1}{2}K_{i,i}(\alpha_i^m)^2 + (\sum_{j=1}^{l} K_{j,i}\bar{\alpha}_j^m + e_i^m - K_{i,i}\bar{\alpha}_i^m)\alpha_i^m\big) + \frac{(\alpha_i^{y_i})^2}{4C} + \text{constants}. \end{aligned} \tag{29}$$

Equation (29) then leads to the objective function of (28), while the constraints are directly obtained from (9). Note that

$$B_m = \bar{\boldsymbol{w}}_m^T \boldsymbol{x}_i + e_i^m - K_{i,i}\bar{\alpha}_i^m \tag{30}$$

if

$$\bar{\boldsymbol{w}}_m = \sum_{i=1}^{l} \bar{\alpha}_i^m \boldsymbol{x}_i,\ m = 1, \ldots, k$$

are maintained.[5]

4 Solving the Sub-problem

We discuss how to solve the sub-problem when $A > 0$. If $A = 0$, then $\boldsymbol{x}_i = \mathbf{0}$. Thus this instance gives a constant value $\xi_i = 1$ to the primal objective function (8), and the value of $\alpha_i^m,\ m = 1, \ldots, k$ have no effect on $\boldsymbol{w}_m$ defined in (16), so we can skip solving the sub-problem.

We follow the approach by Crammer and Singer to solve the sub-problem, although there are some interesting differences. Their method first computes

$$D_m = B_m + AC_{y_i}^m,\ m = 1, \ldots, k.$$

Then it starts with a set $\Phi = \phi$ and sequentially adds one index m to Φ by the decreasing order of D_m until the following inequality is satisfied.

$$\beta = \frac{-AC + \sum_{m \in \Phi} D_m}{|\Phi|} \geq \max_{m \notin \Phi} D_m. \tag{31}$$

The optimal solution of (26) is computed by:

$$\alpha_i^m = \min(C_{y_i}^m, \frac{\beta - B_m}{A}),\ m = 1, \ldots, k. \tag{32}$$

Crammer and Singer gave a lengthy proof to show the correctness of this method. Our contribution here is to derive the algorithm and prove its correctness by easily analyzing the KKT optimality condition.

We now derive an algorithm for solving (28). Let us define $A_{y_i} \equiv A + 1/2C$. The KKT conditions of (28) indicate that there are scalars $\beta,\ \rho_m,\ m = 1, \ldots, k$, such that

$$\sum_{m=1}^{k} \alpha_i^m = 0, \tag{33}$$

$$\alpha_i^m \leq 0,\ \forall m \neq y_i, \tag{34}$$

$$\rho_m \alpha_i^m = 0,\ \rho_m \geq 0,\ \forall m \neq y_i, \tag{35}$$

$$A\alpha_i^m + B_m - \beta = -\rho_m,\ \forall m \neq y_i, \tag{36}$$

$$A_{y_i} \alpha_i^{y_i} + B_{y_i} - \beta = 0. \tag{37}$$

[5] See details of solving linear Crammer and Singer's multi-class SVM in Keerthi et al. (2008).

Using (34), Equations (35) and (36) are equivalent to

$$A\alpha_i^m + B_m - \beta = 0, \text{ if } \alpha_i^m < 0,\ m \neq y_i, \tag{38}$$
$$A\alpha_i^m + B_m - \beta = B_m - \beta \leq 0, \text{ if } \alpha_i^m = 0,\ m \neq y_i. \tag{39}$$

Now KKT conditions become (33)-(34), (37), and (38)-(39). If β is known, we prove that

$$\alpha_i^m \equiv \begin{cases} \min(0, \frac{\beta - B_m}{A}) & \text{if } m \neq y_i, \\ \frac{\beta - B_{y_i}}{A_{y_i}} & \text{if } m = y_i, \end{cases} \tag{40}$$

satisfies all KKT conditions except (33). Clearly, the way to get α_i^m in (40) ensures $\alpha_i^m \leq 0,\ \forall m \neq y_i$, so (34) holds. From (40), when $\beta < B_m$, we have $\alpha_i^m < 0$ and $\beta - B_m = A\alpha_i^m$. Thus, (38) is satisfied. Otherwise, $\beta \geq B_m$ and $\alpha_i^m = 0$ satisfy (39). Also notice that $\alpha_i^{y_i}$ is directly obtained from (37).

The remaining task is how to find β such that (33) holds. From (33), (37), and (38) we obtain

$$A_{y_i} \sum_{m:\alpha_i^m < 0} (\beta - B_m) + A(\beta - B_{y_i}) = 0.$$

Hence,

$$\beta = \frac{AB_{y_i} + A_{y_i} \sum_{m:\alpha_i^m<0} B_m}{A + \sum_{m:\alpha_i^m<0} A_{y_i}} = \frac{\frac{A}{A_{y_i}} B_{y_i} + \sum_{m:\alpha_i^m<0} B_m}{\frac{A}{A_{y_i}} + |\{m|\alpha_i^m < 0\}|}. \tag{41}$$

Combining (41) and (39), we begin with a set $\Phi = \phi$, and then sequentially add one index m to Φ by the decreasing order of $B_m,\ m = 1, \ldots, k,\ m \neq y_i$, until

$$h = \frac{\frac{A}{A_{y_i}} B_{y_i} + \sum_{m\in\Phi} B_m}{\frac{A}{A_{y_i}} + |\Phi|} \geq \max_{m\notin\Phi} B_m. \tag{42}$$

Let $\beta = h$ when (42) is satisfied. Algorithm 1 lists the details for solving the sub-problem (28). To prove (33), it is sufficient to show that β and $\alpha_i^m,\ \forall m$, obtained by Algorithm 1 satisfies (41). This is equivalent to showing that the set Φ of indices included in step 5 of Algorithm 1 satisfies

$$\Phi = \{m \mid \alpha_i^m < 0\}.$$

From (40), we prove the following equivalent result.

$$\beta < B_m,\ \forall m \in \Phi \text{ and } \beta \geq B_m,\ \forall m \notin \Phi. \tag{43}$$

The second inequality immediately follows from (42). For the first, assume t is the last element added to Φ. When it is considered, (42) is not satisfied yet, so

$$\frac{\frac{A}{A_{y_i}} B_{y_i} + \sum_{m\in\Phi\setminus\{t\}} B_m}{\frac{A}{A_{y_i}} + |\Phi| - 1} < B_t. \tag{44}$$

ALGORITHM 1: Solving the sub-problem

1. Given $A,\ A_{y_i}$, and $B = \{B_1, \ldots, B_k\}$.
2. $D \leftarrow B$
3. Swap D_1 and D_{y_i}, then sort $D \setminus \{D_1\}$ in decreasing order.
4. $r \leftarrow 2,\ \beta \leftarrow D_1 \times A/A_{y_i}$
5. While $r \le k$ and $\beta/(r - 2 + A/A_{y_i}) < D_r$
   5.1. $\beta \leftarrow \beta + D_r$
   5.2. $r \leftarrow r + 1$
6. $\beta \leftarrow \beta/(r - 2 + A/A_{y_i})$
7. $\alpha_i^m \leftarrow \min\big(0, (\beta - B_m)/A\big),\ \forall m \ne y_i$
8. $\alpha_i^{y_i} \leftarrow (\beta - B_{y_i})/A_{y_i}$

Using (44) and the fact that elements in Φ are added in the decreasing order of B_m,

$$\begin{aligned}
&\frac{A}{A_{y_i}} B_{y_i} + \sum_{m \in \Phi} B_m = \frac{A}{A_{y_i}} B_{y_i} + \sum_{m \in \Phi \setminus \{t\}} B_m + B_t \\
&< (|\Phi| - 1 + \frac{A}{A_{y_i}}) B_t + B_t = (|\Phi| + \frac{A}{A_{y_i}}) B_t \\
&\le (|\Phi| + \frac{A}{A_{y_i}}) B_s,\ \forall s \in \Phi.
\end{aligned}$$

Thus, we have the first inequality in (43).

With all KKT conditions satisfied, Algorithm 1 obtains an optimal solution of (28).

By comparing (31), (32) and (42), (40) respectively, we can see that the procedures for L1 loss and L2 loss are similar but different in several aspects. In particular, because $\alpha_i^{y_i}$ is unconstrained, B_{y_i} is considered differently from other B_m's in (42).

5 Other Issues and Extensions

In this section, we discuss other details of the decomposition method. Some of them are similar to those for the L1 case. We also extend problems (8)-(9) to more general settings. In the end we discuss advantages and disadvantages of solving two dual forms (9) and (24).

5.1 Extensions to Use Kernels

It is straightforward to extend our algorithm to use kernels. The only change is to replace

$$K_{i,j} = \boldsymbol{x}_i^T \boldsymbol{x}_j$$

in (6) with

$$K_{i,j} = \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j), \tag{45}$$

where $\phi(\boldsymbol{x})$ is a function mapping data to a higher dimensional space.

5.2 Working Set Selection

We mentioned in Section 3 that at each iteration of the decomposition method, an index i is selected so that $\alpha_i^1, \ldots, \alpha_i^k$ are updated. This procedure is called working set selection. If kernels are not used, we follow Keerthi et al. (2008) to sequentially select $i \in \{1, \ldots, l\}$.[6] For linear SVM, it is known that more sophisticated selections such as using gradient information may not be cost-effective; see the detailed discussion in Section 4.1 of Hsieh et al. (2008).

For kernel SVM, we can use gradient information for working set selection because the cost is relatively low compared to that of kernel evaluations. In Crammer and Singer (2001), to solve problems with L1 loss, they select an index by

$$i = \arg \max_{i \in \{1,\ldots,l\}} \varphi_i, \tag{46}$$

where

$$\varphi_i = \max_{1 \le m \le k} \hat{g}_i^m - \min_{m: \alpha_i^m < C_{y_i}^m} \hat{g}_i^m, \tag{47}$$

and $\hat{g}_i^m,\ i = 1, \ldots, l,\ m = 1, \ldots, k$, are the gradient of (3)'s objective function. The reason behind (46) is that φ_i shows the violation of the optimality condition. Note that for problem (3), $\boldsymbol{\alpha}$ is optimal if and only if $\boldsymbol{\alpha}$ is feasible and

$$\varphi_i = 0,\ i = 1, \ldots, l. \tag{48}$$

See the derivation in Crammer and Singer (2001, Section 5). For L2 loss, we can apply a similar setting by

$$\varphi_i = \max_{1 \le m \le k} g_i^m - \min_{m: \alpha_i^m < 0 \text{ or } m = y_i} g_i^m,\ i = 1, \ldots, l,$$

where

$$g_i^m = \sum_{j=1}^{l} K_{i,j} \alpha_j^m + e_i^m + (1 - e_i^m) \frac{\alpha_i^m}{2C},\ i = 1, \ldots, l,\ m = 1, \ldots, k,$$

are the gradient of the objective function in (9). Note that $C_{y_i}^m$ in (47) becomes 0 here.

5.3 Stopping Condition

From (48), a stopping condition of the decomposition method can be

$$\max_i \varphi_i \le \epsilon,$$

where ϵ is the stopping tolerance. The same stopping condition can be used for the L2 case.

[6] In practice, for faster convergence, at each cycle of l steps, they sequentially select indices from a permutation of $\{1, \ldots, l\}$.

5.4 Extension to Assign Different Regularization Parameters to Each Class

In some applications, we may want to assign different regularization parameter C_i to class i. This can be easily achieved by replacing C in earlier discussion with C_i.

5.5 Solving Problem (9) Versus Problem (24)

In Section 2.2, we mentioned an issue of solving problem (9) or problem (24). Based on the investigation of decomposition methods so far, we give some brief discussion.

Some works for structured SVM have solved the dual problem, where (24) is a special case. For example, in Chang et al. (2010), a dual coordinate descent method is applied for solving the dual problem of L2-loss structured SVM. Because (24) does not contain any linear constraint, they are able to update a single $\hat{\alpha}_i^m$ at a time.[7] This setting is related to the decomposition method discussed in Section 3, although ours update k variables at a time. If $\hat{\alpha}_i^m$ is selected for update, the computational bottleneck is on calculating

$$\boldsymbol{w}^T \delta(i,m) = \boldsymbol{w}_{y_i}^T \boldsymbol{x}_i - \boldsymbol{w}_m^T \boldsymbol{x}_i \tag{49}$$

for constructing a one-variable sub-problem.[8] From (11), Equation (49) involves the calculation of

$$\sum_{j=1}^{l} \hat{\alpha}_j^m K_{j,i} \quad \text{and} \quad \sum_{j=1}^{l} \hat{\alpha}_j^{y_i} K_{j,i}. \tag{50}$$

The cost of $2l$ kernel evaluations is $O(ln)$ if each kernel evaluation takes $O(n)$. For our decomposition method to solve (9), to update k variables $\alpha_i^m,\ m = 1, \ldots, k$, together, the number of kernel evaluations is only l; see Equations (27) and (29). More precisely, the complexity of Algorithm 1 to solve the sub-problem (28) is

$$O(k \log k + ln + kl), \tag{51}$$

where $O(k \log k)$ is for sorting $B_m,\ \forall m \neq y_i$, and $O(kl)$ is for obtaining $B_m,\ m = 1, \ldots, k$ in Equation (27). If k is not large, $O(ln)$ is the dominant term in (51). This analysis indicates that regardless of how many elements in $\alpha_i^m,\ m = 1, \ldots, k$, are updated, we always need to calculate the i-th kernel column $K_{j,i},\ j = 1, \ldots, l$. In this regard, the decomposition method for problem (9) by solving a sequence of sub-problems (28) nicely allows us to update as many variables as possible under a similar number of kernel evaluations.

If kernel is not applied, interestingly the situation becomes different. The $O(ln)$ cost of computing (50) is reduced to $O(n)$ because $\boldsymbol{w}_{y_i}$ and $\boldsymbol{w}_m$ are available. If

[7] This is not possible for the dual problem of L1-loss structured SVM. We have mentioned in Section 2.2 that it contains a linear constraint.

[8] We omit details because the derivation is similar to that for deriving the sub-problem (28).

Algorithm 1 is used, from (30), the complexity in (51) for updating k variables becomes

$$O(k \log k + kn).$$

For updating an $\hat{\alpha}_i^m$ by (49), the cost is $O(n)$. Therefore, if $\log k < n$, the cost of updating $\alpha_i^m,\ m = 1, \ldots, k$, together is k times of updating a single variable. Then, the decomposition method for solving problem (9) and sub-problem (28) may not be better than a coordinate descent method for solving problem (24). Note that we have focused on the cost per sub-problem, but there are many other issues such as the convergence speed (i.e., the number of iterations). Memory access also affects the computational time. For the coordinate descent method to update a variable $\hat{\alpha}_i^m$, the corresponding $\boldsymbol{w}_m, \boldsymbol{x}_i$, and $\hat{\alpha}_i^m$ must be accessed. In contrast, the approach of solving sub-problem (28) accesses data and variables more systematically. An important future work is to conduct a serious comparison and identify the better approach.

6 Experiments

In this section, we compare the proposed method for L2 loss with an existing implementation for L1 loss. We check linear as well as kernel multi-class SVMs. Moreover, a comparison of sensitivity to parameters is also conducted. Our implementation is extended from those in LIBLINEAR (Fan et al., 2008) and BSVM (Hsu and Lin, 2002), which respectively include solvers for linear and kernel L1-loss Crammer and Singer multi-class SVM. Programs for experiments in this paper are available at `http://www.csie.ntu.edu.tw/~cjlin/papers/l2mcsvm/codes.zip`. All data sets used are available at
`http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

6.1 Linear Multi-class SVM

We check both training time and test accuracy of using L1 and L2 losses. We consider the four data sets used in Keerthi et al. (2008): news20, MNIST, sector and rcv1. We select the regularization parameter C by checking five-fold cross-validation (CV) accuracy of using values in $\{2^{-5}, 2^{-4}, \ldots, 2^5\}$. The stopping tolerance is $\epsilon = 0.1$. The details of the data sets are listed in Table 1, and the experiment results can be found in Table 2

The accuracy values are comparable. One may observe that the training time of using L1 loss is less. This result is opposite to that of binary classification; see experiments in Hsieh et al. (2008). In binary classification, when C approaches zero, the Hessian matrix of L2-loss SVM is close to the matrix $I/(2C)$, where I is the identity matrix. Thus, the optimization problem is easier to solve. However, for Crammer and Singer's multi-class SVM, when C approaches zero, only l of the Hessian's kl diagonal elements become close to $1/(2C)$. This may be the reason why for multi-class SVM, using L2 loss does not lead to faster training time.

Table 1: Data sets for experiments of linear multi-class SVMs. n is the number of features and k is the number of classes.

data set	#training	#testing	n	k	C for L1 loss	C for L2 loss
news20	15,395	3,993	62,061	20	2^{-1}	2^{-1}
MNIST	60,000	10,000	780	10	2^{-5}	2^{-5}
sector	6,412	3,207	55,197	105	2^{0}	2^{0}
rcv1	15,564	518,571	47,236	53	2^{0}	2^{0}

Table 2: Linear multi-class SVMs: we compare training time (in seconds) and test accuracy between L1 loss and L2 loss.

	L1 loss		L2 loss	
data set	training time	test accuracy	training time	test accuracy
news20	0.69	85.50%	0.80	85.00%
MNIST	2.98	92.93%	11.64	92.54%
sector	2.98	94.36%	3.59	94.14%
rcv1	1.66	88.64%	1.53	88.50%

6.2 Kernel Multi-class SVM

We use the same data sets and the same procedure in Hsu and Lin (2002) to compare test accuracy, training time and sparsity (i.e., percentage of training data as support vectors) of using L1 and L2 losses. We use the RBF kernel

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}.$$

We fix the cache size for the kernel matrix as 2048 MB. The stopping tolerance is set to be $\epsilon = 0.001$ in all data sets except letter and shuttle, whose stopping tolerance is $\epsilon = 0.1$ for avoiding lengthy training time.

The data set description is in Table 3 and the results are listed in Table 4. For dna, satimage, letter and shuttle, both training and test sets are available. We follow Hsu and Lin (2002) to split the training data to 70% training and 30% validation for finding parameters among $C = \{2^{-2}, 2^{-1}, \ldots, 2^{12}\}$ and $\gamma = \{2^{-10}, 2^{-9}, \ldots, 2^{4}\}$. We then train the whole training set by the best parameters and report the test accuracy and the model sparsity. For the rest data sets whose test sets are not available, we report the best ten-fold CV accuracy and the model sparsity.[9] From Table 4, we can see that L2-loss multi-class SVM gives comparable accuracy to L1-loss SVM. Note that the accuracy and the parameters of L1-loss multi-class SVM on some data sets are slightly different from those in Hsu and Lin (2002) because of the random data segmentation in the validation procedure and the different versions of the BSVM code.

Training time and sparsity are very different between using L1 and L2 losses because they highly depend on the parameters used. To remove the effect of different parameters, in Section 6.3, we present the average result over a set of parameters.

[9]The sparsity is the average of the 10 models in the CV procedure.

Table 3: Data sets for experiments of kernel multi-class SVMs. n is the number of features and k is the number of classes.

data set	#training	#testing	n	k	(C,γ) for L1 loss	(C,γ) for L2 loss
iris	150	0	4	3	$(2^3, 2^{-5})$	$(2^{10}, 2^{-7})$
wine	178	0	13	3	$(2^0, 2^{-2})$	$(2^1, 2^{-3})$
glass	214	0	13	6	$(2^1, 2^3)$	$(2^{-1}, 2^3)$
vowel	528	0	10	11	$(2^2, 2^1)$	$(2^4, 2^1)$
vehicle	846	0	18	4	$(2^7, 2^{-3})$	$(2^5, 2^{-4})$
segment	2,310	0	19	7	$(2^3, 2^3)$	$(2^7, 2^0)$
dna	2,000	1,186	180	3	$(2^1, 2^{-6})$	$(2^1, 2^{-6})$
satimage	4,435	2,000	36	6	$(2^2, 2^2)$	$(2^4, 2^2)$
letter*	15,000	5,000	16	26	$(2^4, 2^2)$	$(2^{11}, 2^4)$
shuttle*	43,500	14,500	9	7	$(2^{10}, 2^4)$	$(2^9, 2^4)$

*: $\epsilon = 0.1$ is used.

6.3 Sensitivity to Parameters

Parameter selection is a time consuming process. To avoid checking many parameters, we hope a method is not sensitive to parameters. In this section, we compare the sensitivity of L1 and L2 losses by presenting the average performance over a set of parameters.

For linear multi-class SVM, 11 values of C are selected: $\{2^{-5}, 2^{-4}, \ldots, 2^5\}$, and we present the average and the standard deviation of training time and test accuracy. The results are in Table 5. For the kernel case, We pick C and γ from the two sets $\{2^{-1}, 2^2, 2^5, 2^8\}$ and $\{2^{-6}, 2^{-3}, 2^0, 2^3\}$, respectively, so 16 different results are generated.[10] We then report average and standard deviation in Table 6.

From Tables 5 and 6, L2 loss is worse than L1 loss on the average training time and sparsity. The higher percentage of support vectors is the same as the situation in binary classification because the squared hinge loss leads to many small but non-zero α_i^m. Interestingly, the average performance (test or CV accuracy) of L2 loss is better. Therefore, if using L2 loss, it may be easier to locate a good parameter setting. We find that the same situation occurs in binary classification, although this result was not clearly mentioned in previous studies. An investigation shows that L2 loss gives better accuracy when C is small. In this situation, both L1- and L2-loss SVM suffer from the underfitting of training data. However, because L2 loss gives a higher penalty than L1 loss, underfitting is less severe.

6.4 Summary of Experiments

Based on the experiments, we have the following findings.

[10]We use a subset of (C, γ) values in Section 6.2 to save the running time. To report the average training time, we must run all jobs in the same machine. In contrast, several machines were used in Section 6.2 to obtain CV accuracy of all parameters.

Table 4: Kernel multi-class SVMs: we compare training time (in seconds), test or CV accuracy, and sparsity between L1 loss and L2 loss. nSV represents the percentage of training data that are support vectors.

	L1 loss			L2 loss		
data set	training time	test or CV accuracy	nSV	training time	test or CV accuracy	nSV
iris	0.07	96.67%	37.33%	1.53	98.00%	16.67%
wine	0.03	98.31%	28.65%	0.03	98.31%	33.96%
glass	0.19	73.83%	80.01%	0.18	74.30%	98.91%
vowel	2.61	98.86%	67.93%	2.59	98.86%	72.31%
vehicle	27.73	86.52%	53.73%	24.14	86.41%	65.29%
segment	14.37	97.62%	46.65%	21.68	97.62%	19.08%
dna	1.43	95.87%	46.90%	1.68	95.62%	56.10%
satimage	5.35	92.35%	60.41%	5.35	92.45%	60.92%
letter*	87.46	97.76%	42.56%	136.99	97.14%	78.56%
shuttle*	64.88	99.94%	0.66%	60.98	99.94%	1.41%

*: $\epsilon = 0.1$ is used.

Table 5: Sensitivity to parameters: linear multi-class SVMs. We present average±standard deviation.

	L1 loss		L2 loss	
data set	training time	test accuracy	training time	test accuracy
news20	1.81 ± 1.83	$83.85 \pm 1.33\%$	3.59 ± 4.70	$84.39 \pm 0.41\%$
MNIST	340.67 ± 532.21	$92.68 \pm 0.20\%$	1348.45 ± 1934.30	$92.25 \pm 0.21\%$
sector	5.96 ± 5.45	$93.59 \pm 0.77\%$	7.66 ± 8.02	$93.91 \pm 0.46\%$
rcv1	2.66 ± 2.27	$86.78 \pm 2.42\%$	3.83 ± 4.28	$87.78 \pm 0.89\%$

1. If using the best parameter, L2 loss gives comparable accuracy to L1 loss. For the training time and the number of support vectors, L2 loss is better for some problems, but worse for some others. The situation highly depends on the chosen parameter.

2. If we take the whole procedure of parameter selection into consideration, L2 loss is worse than L1 loss on training time and sparsity. However, the region of suitable parameters is larger. Therefore, we can check fewer parameters if using L2 loss.

7 Conclusions

This paper extends Crammer and Singer's multi-class SVM to apply L2 loss. We give detailed derivations and discuss some interesting differences from the L1 case. Our results serve as a useful reference for those who intend to use Crammer and Singer's method with L2 loss. Finally, we have extended the software BSVM (after

Table 6: Sensitivity to parameters: kernel multi-class SVMs. We present average±standard deviation. nSV represents the percentage of training data that are support vectors.

		L1 loss	
data set	training time	test or CV accuracy	nSV
iris	0.10 ± 0.12	$93.38 \pm 6.93\%$	$36.02 \pm 17.74\%$
wine	0.09 ± 0.03	$96.45 \pm 0.95\%$	$54.38 \pm 31.15\%$
glass	5.57 ± 9.55	$66.59 \pm 5.24\%$	$84.07 \pm 8.74\%$
vowel	53.86 ± 116.73	$86.38 \pm 15.12\%$	$81.94 \pm 13.17\%$
vehicle	37.21 ± 69.87	$76.85 \pm 5.99\%$	$75.84 \pm 16.68\%$
segment	57.43 ± 72.18	$95.35 \pm 2.91\%$	33.30 ± 14.37
dna	6.29 ± 3.98	$82.05 \pm 7.85\%$	$84.59 \pm 20.68\%$
satimage	80.70 ± 167.67	$89.37 \pm 2.83\%$	$47.63 \pm 19.66\%$
letter*	1698.97 ± 3908.05	$90.51 \pm 9.18\%$	$54.96 \pm 18.30\%$
shuttle*	396.53 ± 692.35	$98.66 \pm 1.89\%$	$8.07 \pm 6.63\%$
		L2 loss	
iris	0.18 ± 0.20	$94.96 \pm 2.12\%$	$45.00 \pm 25.68\%$
wine	0.11 ± 0.06	$96.73 \pm 1.11\%$	$58.66 \pm 30.97\%$
glass	14.86 ± 28.12	$68.84 \pm 3.40\%$	$88.99 \pm 9.61\%$
vowel	59.05 ± 118.68	$89.96 \pm 12.15\%$	$87.80 \pm 12.72\%$
vehicle	75.25 ± 157.46	$78.93 \pm 4.91\%$	$81.10 \pm 16.52\%$
segment	138.66 ± 232.03	$96.11 \pm 2.07\%$	$44.12 \pm 20.86\%$
dna	14.49 ± 8.11	$82.21 \pm 7.73\%$	$88.75 \pm 19.72\%$
satimage	173.93 ± 397.54	$89.84 \pm 2.21\%$	$55.05 \pm 18.89\%$
letter*	3205.32 ± 7926.28	$92.28 \pm 7.24\%$	$63.53 \pm 21.32\%$
shuttle*	2844.68 ± 6056.00	$98.79 \pm 1.62\%$	$16.95 \pm 11.00\%$

*: $\epsilon = 0.1$ is used.

version 2.07) to include the proposed implementation.

Acknowledgment

This work was supported in part by the National Science Council of Taiwan via the grant 98-2221-E-002-136-MY3. The authors thank the anonymous reviewers and Ming-Wei Chang for valuable comments. We also thank Yong Zhuang and Wei-Sheng Chin for their help in finding errors of this paper.

References

Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

Ming-Wei Chang, Vivek Srikumar, Dan Goldwasser, and Dan Roth. Structured

output learning with indirect supervision. In *Proceedings of the Twenty Seven International Conference on Machine Learning (ICML)*, pages 199–206, 2010.

Corina Cortes and Vladimir Vapnik. Support-vector network. *Machine Learning*, 20:273–297, 1995.

Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2: 265–292, 2001.

Koby Crammer and Yoram Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, (2–3):201–233, 2002.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf`.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the Twenty Fifth International Conference on Machine Learning (ICML)*, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/cddual.pdf`.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

S. Sathiya Keerthi, Sellamanickam Sundararajan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A sequential dual method for large scale multi-class linear SVMs. In *Proceedings of the Forteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 408–416, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/sdm_kdd.pdf`.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

A Solving the Sub-problems when $A \leq 0$

Our decomposition method only solves the sub-problem when $A > 0$. To cover the case when K is not a valid kernel and $K_{i,i}$ is any possible value, we still need to solve the sub-problems when $A \leq 0$.

A.1 $A = 0$

When $A = 0$, for L1 loss, the sub-problem (26) reduces to a linear programming problem. Define

$$\bar{m} \equiv \arg\max_{m:m\neq y_i} B_m,$$

then the optimal solution is

$$\begin{cases} \alpha_i^m = 0,\ m = 1, \ldots, k & \text{if } B_{y_i} - B_{\bar{m}} \geq 0, \\ \begin{cases} \alpha_i^{y_i} = C \\ \alpha_i^{\bar{m}} = -\alpha_i^{y_i} \\ \alpha_i^m = 0,\ \forall m \neq y_i \text{ and } m \neq \bar{m}, \end{cases} & \text{if } B_{y_i} - B_{\bar{m}} < 0. \end{cases}$$

It is more complicated in the L2-loss case, because there is a quadratic term of $\alpha_i^{y_i}$. To solve the sub-problem (28), we reformulate it by the following procedure. From footnote 2, we know $\alpha_i^{y_i} \geq 0$. For any fixed $\alpha_i^{y_i}$, the sub-problem becomes

$$\begin{aligned} \min_{\alpha_i^m, m\neq y_i} \quad & \sum_{m:m\neq y_i} B_m \alpha_i^m \\ \text{subject to} \quad & \sum_{m:m\neq y_i} \alpha_i^m = -\alpha_i^{y_i}, \\ & \alpha_i^m \leq 0,\ \forall m \neq y_i. \end{aligned}$$

Clearly, the solution is

$$\alpha_i^m = \begin{cases} -\alpha_i^{y_i} & \text{if } m = \bar{m}, \\ 0 & \text{otherwise.} \end{cases} \tag{52}$$

Therefore, the sub-problem (28) is reduced to the following one-variable problem.

$$\min_{\alpha_i^{y_i} \geq 0} \quad \frac{(\alpha_i^{y_i})^2}{4C} + (B_{y_i} - B_{\bar{m}})\alpha_i^{y_i}. \tag{53}$$

The solution of (53) is

$$\alpha_i^{y_i} = \max\big(0, -2(B_{y_i} - B_{\bar{m}})C\big). \tag{54}$$

Using (52) and (54), the optimal solution can be written as

$$\begin{cases} \alpha_i^m = 0,\ m = 1, \ldots, k & \text{if } B_{y_i} - B_{\bar{m}} \geq 0, \\ \begin{cases} \alpha_i^{y_i} = -2(B_{y_i} - B_{\bar{m}})C \\ \alpha_i^{\bar{m}} = -\alpha_i^{y_i} \\ \alpha_i^m = 0,\ \forall m \neq y_i \text{ and } m \neq \bar{m} \end{cases} & \text{if } B_{y_i} - B_{\bar{m}} < 0. \end{cases}$$

A.2 $A < 0$

For any given $\alpha_i^{y_i}$ that satisfies their corresponding constraints, both (26) and (28) are equivalent to

$$\begin{aligned} \min_{\alpha_i^m, m\neq y_i} \quad & \sum_{m\neq y_i} \frac{1}{2} A\big((\alpha_i^m + \frac{B_m}{A})^2\big) \\ \text{subject to} \quad & \sum_{m\neq y_i} \alpha_i^m = -\alpha_i^{y_i}, \\ & \alpha_i^m \leq 0,\ \forall m \neq y_i. \end{aligned}$$

When $A < 0$, it is equivalent to

$$\begin{aligned} \max_{\alpha_i^m, m\neq y_i} \quad & \sum_{m\neq y_i} (\alpha_i^m)^2 + \sum_{m\neq y_i} 2\frac{B_m}{A}\alpha_i^m \\ \text{subject to} \quad & \sum_{m\neq y_i} \alpha_i^m = -\alpha_i^{y_i}, \qquad (55) \\ & \alpha_i^m \leq 0,\ \forall m \neq y_i. \qquad (56) \end{aligned}$$

By constraints (55) and (56), we have

$$\begin{aligned} \sum_{m\neq y_i} (\alpha_i^m)^2 \leq (\sum_{m\neq y_i} \alpha_i^m)^2 = (-\alpha_i^{y_i})^2, \text{ and} \\ \sum_{m\neq y_i} \frac{B_m}{A}\alpha_i^m \leq (\frac{B_{\bar{m}}}{A} \sum_{m\neq y_i} \alpha_i^m) = -\frac{B_{\bar{m}}}{A}\alpha_i^{y_i}. \end{aligned}$$

Note that when $A < 0$,

$$\bar{m} = \arg \max_{m:m\neq y_i} B_m = \arg \min_{m:m\neq y_i} \frac{B_m}{A}.$$

Thus clearly the optimal solution is

$$\alpha_i^m = \begin{cases} -\alpha_i^{y_i} & \text{if } m = \bar{m}, \\ 0 & \text{otherwise.} \end{cases}$$

Sub-problem (26) is then reduced to the following one-variable problem.

$$\min_{C\geq \alpha_i^{y_i} \geq 0} \quad A(\alpha_i^{y_i})^2 + (B_{y_i} - B_{\bar{m}})\alpha_i^{y_i} = A(\alpha_i^{y_i} + \frac{B_{y_i} - B_{\bar{m}}}{2A})^2 + \text{ constants.}$$

Its solution is

$$\alpha_i^{y_i} = \begin{cases} 0 & \text{if } \frac{B_{y_i} - B_{\bar{m}}}{2A} \leq -\frac{C}{2}, \\ C & \text{otherwise.} \end{cases}$$

Combine them together, the optimal solution of (26) when $A < 0$ is

$$\begin{cases} \alpha_i^m = 0,\ m = 1, \ldots, k & \text{if } B_{y_i} - B_{\bar{m}} \geq -AC, \\ \alpha_i^{y_i} = C & \\ \alpha_i^{\bar{m}} = -\alpha_i^{y_i} & \text{if } B_{y_i} - B_{\bar{m}} < -AC. \qquad (57) \\ \alpha_i^m = 0,\ \forall m \neq y_i \text{ and } m \neq \bar{m} & \end{cases}$$

When $A = 0$, $-AC = 0$. Therefore, (57) can be used in L1 loss for $A \leq 0$.

For problem (28), when $A < 0$, it is reduced to another one-variable problem.

$$\min_{\alpha_i^{y_i} \geq 0} \quad A^*(\alpha_i^{y_i})^2 + (B_{y_i} - B_{\bar{m}})\alpha_i^{y_i}, \tag{58}$$

where $A^* \equiv A + \frac{1}{4C}$. If $A^* = 0$, then (58) reduces to a trivial problem with optimal solution

$$\alpha_i^{y_i} = \begin{cases} 0 & \text{if } B_{y_i} - B_{\bar{m}} \geq 0, \\ \infty & \text{if } B_{y_i} - B_{\bar{m}} < 0. \end{cases}$$

Thus the optimal solution of (28) when $A^* = 0$ is

$$\begin{cases} \alpha_i^m = 0, \ m = 1, \ldots, k & \text{if } B_{y_i} - B_{\bar{m}} \geq 0, \\ \begin{cases} \alpha_i^{y_i} = \infty \\ \alpha_i^{\bar{m}} = -\infty \\ \alpha_i^m = 0, \ \forall m \neq y_i \text{ and } m \neq \bar{m}, \end{cases} & \text{if } B_{y_i} - B_{\bar{m}} < 0. \end{cases}$$

If $A^* \neq 0$, (58) is equivalent to

$$\min_{\alpha_i^{y_i} \geq 0} \quad A^*(\alpha_i^{y_i} + \frac{B_{y_i} - B_{\bar{m}}}{2A^*})^2.$$

When $A^* < 0$, the optimal solution of (28) is

$$\alpha_i^{y_i} = \infty, \ \alpha_i^{\bar{m}} = -\infty, \text{ and } \alpha_i^m = 0, \ \forall m \neq y_i \text{ and } m \neq \bar{m}.$$

While if $A^* > 0, A < 0$, the optimum occurs at

$$\begin{cases} \alpha_i^m = 0, \ m = 1, \ldots, k & \text{if } B_{y_i} - B_{\bar{m}} \geq 0, \\ \begin{cases} \alpha_i^{y_i} = -\dfrac{(B_{y_i} - B_{\bar{m}})}{2A^*} \\ \alpha_i^{\bar{m}} = -\alpha_i^{y_i} \\ \alpha_i^m = 0, \ \forall m \neq y_i \text{ and } m \neq \bar{m} \end{cases} & \text{if } B_{y_i} - B_{\bar{m}} < 0. \end{cases} \tag{59}$$

Note that when $A = 0$, $1/2A^* = 2C$. Thus (59) can be used in L2 loss for $A^* > 0, A \leq 0$.