Chapter 2
Multi-Class Support Vector Machine

Zhe Wang and Xiangyang Xue

Abstract Support vector machine (SVM) was initially designed for binary classification. To extend SVM to the multi-class scenario, a number of classification models were proposed such as the one by Crammer and Singer (J Mach Learn Res 2:265–292, 2001). However, the number of variables in Crammer and Singer's dual problem is the product of the number of samples (l) by the number of classes (k), which produces a large computational complexity. This chapter sorts the existing classical techniques for multi-class SVM into the indirect and direct ones and further gives the comparison for them in terms of theory and experiments. Especially, this chapter exhibits a new Simplified Multi-class SVM (SimMSVM) that reduces the size of the resulting dual problem from $l \times k$ to l by introducing a relaxed classification error bound. The experimental discussion demonstrates that the SimMSVM approach can greatly speed up the training process, while maintaining a competitive classification accuracy.

2.1 Introduction

Support vector machine (SVM) originally separates the binary classes ($k = 2$) with a maximized margin criterion [6]. However, real-world problems often require the discrimination for more than two categories. Thus, the multi-class pattern recognition has a wide range of applications including optical character recognition [27], intrusion detection [18], speech recognition [11], and bioinformatics [2]. In practice,

Z. Wang (✉)
Department of Computer Science and Engineering, East China University of Science and Technology, Shanghai 200237, China
e-mail: wangzhe@ecust.edu.cn

X. Xue
School of Computer Science, Fudan University, Shanghai 200433, China
e-mail: xyxue@fudan.edu.cn

Y. Ma and G. Guo (eds.), *Support Vector Machines Applications*,
DOI 10.1007/978-3-319-02300-7_2,

the multi-class classification problems ($k > 2$) are commonly decomposed into a series of binary problems such that the standard SVM can be directly applied. Two representative ensemble schemes are one-versus-rest (1VR) [36] and one-versus-one (1V1) [21] approaches. Both one-versus-rest and one-versus-one are special cases of the Error Correcting Output Codes (ECOC) [8] which decomposes the multi-class problem into a predefined set of binary problems. A main issue of this approach is to construct a good ECOC matrix. Another proposals [4, 7, 12, 38] are to directly address the multi-class problem in one single optimization processing. This kind of models combines multiple binary-class optimization problems into one single objective function and simultaneously achieves classification of multiple classes. However, a larger computational complexity is required for the size of the resulting Quadratic Programming (QP) problem.

Moreover, Szedmak et al. [33] proposed a multi-class model for L1-norm SVM. In their formulation, the one-versus-rest framework is used. A potential problem with one-versus-rest is that when the number of classes is large, each binary classification becomes highly unbalanced. The unbalanced classification problem occurs when there are many more samples of some classes than others. In this case, standard classifiers tend to be overwhelmed by the large-scale classes and ignore the small ones. The SVM algorithm is constructing a separating hyperplane with the maximal margin. Since only support vectors are used for classification and many majority samples far from the decision boundary can be removed, SVM can be more accurate on moderately unbalanced data. However, SVM is sensitive to high unbalanced classification since it is prone to generating a classifier that has a strong estimation bias towards the majority class and would give a bad accuracy in the classification performance for the minority class, which is discussed in the work of Chawla et al. [5] and Tang et al. [34]. Wang and Shen [37] proposed a method that circumvents the difficulties of one-versus-rest by treating multiple classes jointly. Suykens and Vandewalle [32] extended the LS-SVM methodology [31] to the multi-class case. A drawback of LS-SVM is that its solution is constructed from most of the training examples, which is referred to as the non-sparseness problem. Xia and Li [39] presented a new multi-class LS-SVM algorithm whose solution is sparse in the weight coefficient of support vectors. Fung and Mangasarian [10] followed the PSVM (proximal SVM) [9] idea and applied it to the multi-class problem. This approach is closely aligned with the one-versus-rest method. For each decomposed subproblem, the solution is similar to its binary case (PSVM) which classifies new samples by assigning them to the closer of the two parallel planes that are pushed apart as far as possible.

This chapter revisits the main multi-class methods including indirect and direct ones for SVM. Especially, in this chapter we introduce a new Simplified Multi-class SVM (SimMSVM) [13]. The SimMSVM gives a direct solution for training multi-class predictors. Following Crammer and Singer's work, SimMSVM introduces a relaxed classification error bound. By doing so, the resulting dual problem only involves l variables, where l is the size of training samples. That is, solving a single l-variable QP is enough for a multi-class classification task. We then investigate

the theoretical properties of the loss function of SimMSVM, including the Fisher consistency issue. We also give the experimental comparisons and find that the new simplified model achieves significant speed-up compared with its original model [7].

2.2 Indirect Multi-Class SVM

2.2.1 SVM Classification

SVM seeks to find the optimal separating hyperplane between binary classes by following the maximized margin criterion [6]. Given training vectors $\mathbf{x}_i \in \mathcal{R}^d$, $i = 1,\ldots,l$, in two classes, and the label vector $\mathbf{y} \in \{1,-1\}^l$, the support vector technique requires the solution of the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{H},b\in\mathcal{R},\xi_i\in\mathcal{R}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\varphi(\mathbf{x}_i)+b) \geq 1-\xi_i, \\ & \xi_i \geq 0, \quad i = 1,\ldots,l, \end{aligned} \tag{2.1}$$

where $\mathbf{w} \in \mathcal{R}^d$ is the weight vector, $C \in \mathcal{R}_+$ is the regularization constant, and the mapping function φ projects the training data into a suitable feature space $\mathcal{H}$ so as to allow for nonlinear decision surfaces. A common method to solve (2.1) is through its dual:

$$\begin{aligned} \min_{\alpha\in\mathcal{R}^l} \quad & \frac{1}{2}\alpha^T(\mathbf{K}\odot(\mathbf{y}\mathbf{y}^T))\alpha - \mathbf{e}^T\alpha \\ \text{subject to} \quad & \mathbf{0} \leq \alpha \leq C\mathbf{e}, \quad \mathbf{y}^T\alpha = 0, \end{aligned} \tag{2.2}$$

where $\mathbf{e} \in \mathcal{R}^l$ is a vector of all 1's, $\mathbf{K} \in \mathcal{R}^{l\times l}$ is the kernel matrix, and $\odot$ denotes the Hadamard–Schur (elementwise) product. Symbols in bold represent matrices or vectors, as particular case, the symbol $\mathbf{0}$ represents the vector with all components set to 0.

Instead of (2.1), Mangasarian and Musicant [25] proposed a new SVM formulation:

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{H},b\in\mathcal{R},\xi_i\in\mathcal{R}} \quad & \frac{1}{2}\left(\mathbf{w}^T\mathbf{w}+b^2\right) + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\varphi(\mathbf{x}_i)+b) \geq 1-\xi_i, \\ & \xi_i \geq 0, \quad i = 1,\ldots,l, \end{aligned} \tag{2.3}$$

whose dual becomes a bound-constrained problem:

$$\begin{aligned} \min_{\alpha \in \mathcal{R}^l} \quad & \frac{1}{2}\alpha^T \left((\mathbf{K} + \mathbf{e}\mathbf{e}^T) \odot (\mathbf{y}\mathbf{y}^T) \right) \alpha - \mathbf{e}^T \alpha \\ \text{subject to} \quad & \mathbf{0} \leq \alpha \leq C\mathbf{e}. \end{aligned} \tag{2.4}$$

The motivation behind (2.3) is that, it is supposed easier to handle its dual problem (2.4) without linear constraint. Hsu and Lin [15] demonstrate that (2.3) is an acceptable formulation in terms of generalization errors though an additional term $b^2/2$ is added to the objective. This method is implemented in the software BSVM [16] which is proved to have fast convergence in the literature [15].

2.2.2 One-Versus-Rest Approach

The one-versus-rest (1VR) approach [36] constructs k separate binary classifiers for k-class classification. The m-th binary classifier is trained using the data from the m-th class as positive examples and the remaining $k-1$ classes as negative examples. During test, the class label is determined by the binary classifier that gives maximum output value. A major problem of the one-versus-rest approach is the imbalanced training set. Suppose that all classes have an equal size of training examples, the ratio of positive to negative examples in each individual classifier is $\frac{1}{k-1}$. In this case, the symmetry of the original problem is lost.

2.2.3 One-Versus-One Approach

Another classical approach for multi-class classification is the one-versus-one (1V1) or pairwise decomposition [20]. It evaluates all possible pairwise classifiers and thus induces $k(k-1)/2$ individual binary classifiers. Applying each classifier to a test example would give one vote to the winning class. A test example is labeled to the class with the most votes. The size of classifiers created by the one-versus-one approach is much larger than that of the one-versus-rest approach. However, the size of QP in each classifier is smaller, which makes it possible to train fast. Moreover, compared with the one-versus-rest approach, the one-versus-one method is more symmetric. Platt et al. [28] improved the one-versus-one approach and proposed a method called Directed Acyclic Graph SVM (DAGSVM) that forms a tree-like structure to facilitate the testing phase. As a result, it takes only $k-1$ individual evaluations to decide the label of a test example.

2.3 Direct Multi-Class SVM

2.3.1 Weston and Watkins' Multi-Class SVM

Instead of creating several binary classifiers, a more natural way is to distinguish all classes in one single optimization processing, as proposed by Vapnik [36], Weston and Watkins [38], and Bredensteiner and Bennett [4]. For a k-class problem, these methods design a single objective function for training all k-binary SVMs simultaneously and maximize the margins from each class to the remaining ones. Here, we take the method by Weston and Watkins as an example. Given a labeled training set represented by $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$ of cardinality l, where $\mathbf{x}_i \in \mathcal{R}^d$ and $y_i \in \{1, \ldots, k\}$, the formulation proposed in [38] is given as follows:

$$\begin{aligned} \min_{\mathbf{w}_m \in \mathcal{H}, \mathbf{b} \in \mathcal{R}^k, \xi \in \mathcal{R}^{l \times k}} \quad & \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m^T \mathbf{w}_m + C \sum_{i=1}^{l} \sum_{t \neq y_i} \xi_{i,t} \\ \text{subject to} \quad & \mathbf{w}_{y_i}^T \varphi(\mathbf{x}_i) + b_{y_i} \geq \mathbf{w}_t^T \varphi(\mathbf{x}_i) + b_t + 2 - \xi_{i,t}, \\ & \xi_{i,t} \geq 0, \\ & i = 1, \ldots, l, \; t \in \{1, \ldots, k\} \setminus y_i. \end{aligned} \tag{2.5}$$

The resulting decision function is

$$\text{argmax}_m f_m(\mathbf{x}) = \text{argmax}_m (\mathbf{w}_m^T \varphi(\mathbf{x}) + b_m). \tag{2.6}$$

The main disadvantage of this approach is that the computational time may be very high due to the enormous size of the resulting QP.

2.3.2 Crammer and Singer's Multi-Class SVM

Crammer and Singer [7] presented another "all-together" approach by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}_m \in \mathcal{H}, \xi \in \mathcal{R}^l} \quad & \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m^T \mathbf{w}_m + C \sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & \mathbf{w}_{y_i}^T \varphi(\mathbf{x}_i) - \mathbf{w}_t^T \varphi(\mathbf{x}_i) \geq 1 - \delta_{y_i,t} - \xi_i, \\ & i = 1, \ldots, l, \; t \in \{1, \ldots, k\}, \end{aligned} \tag{2.7}$$

where $\delta_{i,j}$ is the Kronecker delta (defined as 1 for $i = j$ and as 0 otherwise). The resulting decision function is

$$\text{argmax}_m f_m(\mathbf{x}) = \text{argmax}_m \mathbf{w}_m^T \varphi(\mathbf{x}). \tag{2.8}$$

Note that the constraints $\xi_i \geq 0$, $i = 1, \ldots, l$, are implicitly indicated in the margin constraints of (2.7) when t equals y_i. In addition, (2.7) focuses on classification rule (2.8) without the bias terms b_m, $m = 1, \ldots, k$. A nonzero b_m can be easily modeled by adding an additional constant feature to each $\mathbf{x}$ (see, e.g., [26]).

2.3.3 Simplified Multi-Class SVM (SimMSVM)

2.3.3.1 Rationale of SimMSVM

Although Crammer and Singer's multi-class SVM gives a compact set of constraints, the number of variables in its dual problem is still $l \times k$ [7]. This value may explode even for small datasets. For instance, an English letter recognition problem with 2,600 samples (100 samples per letter) would require solving a QP of size $2{,}600 \times 26$, which will result in a large computational complexity. Here, we follow Crammer and Singer's work and further introduce a simplified method named SimMSVM for relaxing its constraints [13]. By doing so, solving one single l-variable QP is enough for a multi-class classification task.

Before we describe the new direct multi-class SVM method [13], we first compare the loss function of the above two "all-together" approaches. For a training example $\mathbf{x}_i$, we let

$$\zeta_{i,m} = 1 - f_{y_i}(\mathbf{x}_i) + f_m(\mathbf{x}_i), \tag{2.9}$$

for $m \in \{1, \ldots, k\} \setminus y_i$. If $\zeta_{i,m} > 0$, it depicts the pairwise margin violation between the true class y_i and some other class m. In Weston and Watkins' work, their loss function adds up all positive margin violation ($\zeta_{i,m} > 0$),

$$\xi_i^{(1)} = \sum_{m \neq y_i} [\zeta_{i,m}]_+, \tag{2.10}$$

where $[\cdot]_+ \equiv \max(\cdot, 0)$. In the original work proposed by Weston and Watkins [38], the term ζ adopts the "2" rather than "1" as follows:

$$\zeta_{i,m} = 2 - f_{y_i}(\mathbf{x}_i) + f_m(\mathbf{x}_i),$$

Here in order to compare the work proposed by Weston and Watkins [38] with the other methods consistently, we scale the "2" into "1," i.e. adopt the Eq. (2.9). As for Crammer and Singer's approach, sample loss counts only the maximum positive margin violation

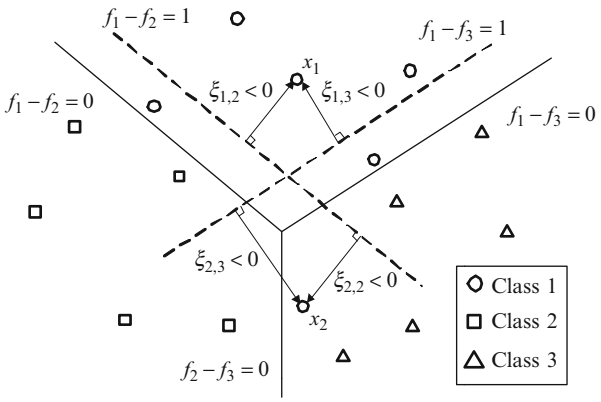

Fig. 2.1 Multi-class classification visualization. Class 1, 2, and 3 are represented by *circles*, *rectangles*, and *triangles*, respectively. The *bold lines* represent some possible class boundaries. The two *dash lines* are two positive margins for the first class. The pairwise margin violations for two examples from the first class, $\zeta_{1,2}$ and $\zeta_{1,3}$ for $\mathbf{x}_1$, and $\zeta_{2,2}$ and $\zeta_{2,3}$ for $\mathbf{x}_2$, are depicted in the figure

$$\xi_i^{(2)} = \left[\max_{m \neq y_i} \zeta_{i,m}\right]_+ . \tag{2.11}$$

Figure 2.1 gives a multi-class classification graphical illustration, where three classes are represented as circles, rectangles, and triangles, respectively. The bold lines represent some possible decision boundaries. We plot the two positive pairwise margins for the first class (shown in dash lines). For a correctly classified example $\mathbf{x}_1$, we have $\xi_1^{(1)} = 0$ and $\xi_1^{(2)} = 0$, i.e., no loss counted, since both $\zeta_{1,2}$ and $\zeta_{1,3}$ are negative. On the other hand, for an example $\mathbf{x}_2$ that violates two margin bounds ($\zeta_{2,2}, \zeta_{2,3} > 0$), both methods generate a loss.

In order to reduce the problem size, the number of constraints should be proportional to l instead of $l \times k$. To construct the multi-class predictors of SimMSVM, we introduce the following relaxed bound

$$\xi_i^{(3)} = \left[\frac{1}{k-1} \sum_{m=1, m \neq y_i}^{k} \zeta_{i,m}\right]_+ . \tag{2.12}$$

That is, we suffer a loss which is linearly proportional to the average of directed distances between an example and its pairwise margins. For the above example shown in Fig. 2.1, the loss (2.12) yields no loss for $\mathbf{x}_1$ as the other two loss functions.

To clearly analyze the SimMSVM, we compare the three loss functions (2.10)–(2.12). The loss (2.10) adds up all positive margin violation ($\zeta_{i,m} > 0$). The loss (2.11) counts only the maximum positive margin violation. The loss (2.12) represents a linear proportion to the average of directed distances between a sample and its pairwise margins. Thus, the relationship between these losses is

Table 2.1 The rate of f_{y_i} be the maximum value of $f_1, \ldots, f_k$ for the ten datasets

Dataset	Iris	Wine	Glass	Vowel	Segment
Rate	98.67%	99.44 %	89.25 %	100.00 %	99.70 %
Dataset	Waveform	DNA	Satimage	Letter	USPS
Rate	94.20 %	98.45 %	99.66 %	99.97 %	99.93 %

$$\xi_i^{(1)} \geq \xi_i^{(2)} \geq \xi_i^{(3)}. \tag{2.13}$$

Figure 2.1 gives a visual example for the three loss functions (2.10)–(2.12). According to their definition, the corresponding losses for the sample x_2 that violates two margin bounds in Fig. 2.1 are

$$\begin{aligned}
\xi_2^{(1)} &= \zeta_{2,2} + \zeta_{2,3} \\
\xi_2^{(2)} &= \max\{\zeta_{2,2}, \zeta_{2,3}\} = \zeta_{2,3} \\
\xi_2^{(3)} &= (\zeta_{2,2} + \zeta_{2,3})/2
\end{aligned}$$

From the inequality (2.13), the relationship in Fig. 2.1 for the losses is

$$\xi_2^{(1)} \geq \xi_2^{(2)} \geq \xi_2^{(3)}.$$

In order to clearly explore the chosen loss function, we rewrite the relaxed loss function (2.12) as

$$\xi_i = \left[1 - f_{y_i}(\mathbf{x}_i) + \frac{1}{k-1} \sum_{m \neq y} f_m(\mathbf{x}_i)\right]_+. \tag{2.14}$$

In other words, the SimMSVM will yield no loss for a training example $\mathbf{x}_i$ if $f_{y_i}(\mathbf{x}_i) \geq 1 + \frac{1}{k-1} \sum_{m \neq y_i} f_m(\mathbf{x}_i)$. Although we cannot guarantee if $\xi_i = 0$ that $f_{y_i}(\mathbf{x}_i) > f_m(\mathbf{x}_i)$ for all $m \neq y_i$, it is most likely that f_{y_i} will be the maximum value of $f_1, \ldots, f_k$. To further explore this statement, we empirically study the problem in all of the ten datasets. Table 2.1 gives the rates that f_{y_i} be the maximum value of $f_1, \ldots, f_k$ and thus validate the claim about the loss function of SimMSVM. In this sense, the (2.14) is reasonable as a classification criterion.

In theory, the relaxed loss function (2.14) may incur more classification error since it fails to upper-bound the $0-1$ loss. In practice, we admit some tolerable classification error in order to obtain a significant speed-up in the training process. The experimental result in Sect. 2.3.4 demonstrate that, SimMSVM gives competitive accuracies with the other multi-class SVM approaches on real world datasets. On the other hand, the multi-class hinge losses of Weston and Watkins [38], Crammer and Singer [7] both satisfy as the upper bound of $0-1$ loss. However, in practice, they all involve a QP of size $l \times k$ and thus cause a high computational complexity for a relatively large number of classes.

2.3.3.2 Architecture of SimMSVM

Induced by the above error bound, the unbiased primal problem of SimMSVM comes up as follows:

$$\begin{aligned} \min_{\mathbf{w}_m \in \mathcal{H}, \xi \in \mathcal{R}^l} \quad & \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m^T \mathbf{w}_m + C \sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & \mathbf{w}_{y_i}^T \varphi(\mathbf{x}_i) - \frac{1}{k-1} \sum_{m \neq y_i} \mathbf{w}_m^T \varphi(\mathbf{x}_i) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, \ldots, l. \end{aligned} \tag{2.15}$$

Note that the number of margin constraints (2.15) has already been reduced to l. To solve the optimization problem (2.15) we use the Karush—Kuhn—Tucker (KKT) theorem. We add a dual set of variables, one for each constraint and get the Lagrangian of the optimization problem

$$\begin{aligned} L(\mathbf{W}, \xi, \alpha, \lambda) = & \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m^T \mathbf{w}_m + C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \lambda_i \xi_i \\ & - \sum_{i=1}^{l} \alpha_i \left(\mathbf{w}_{y_i}^T \varphi(\mathbf{x}_i) - \frac{1}{k-1} \sum_{m \neq y_i} \mathbf{w}_m^T \varphi(\mathbf{x}_i) - 1 + \xi_i \right). \end{aligned} \tag{2.16}$$

By differentiating the Lagrangian with respect to the primal variables, we obtain the following constraints that the variables have to fulfill in order to be an optimal solution:

$$\frac{\partial L}{\partial \mathbf{w}_m} = 0 \Longleftrightarrow \mathbf{w}_m = \sum_{i: y_i = m} \alpha_i \varphi(\mathbf{x}_i) - \frac{1}{k-1} \sum_{i: y_i \neq m} \alpha_i \varphi(\mathbf{x}_i), \tag{2.17}$$

$$\frac{\partial L}{\partial \xi} = 0 \Longleftrightarrow C\mathbf{e} = \lambda + \alpha, \tag{2.18}$$

subject to the constraints $\alpha \geq \mathbf{0}$ and $\lambda \geq \mathbf{0}$.

By elimination of $\mathbf{w}_m$, ξ, and λ, the dual problem of (2.15) is reduced to the following succinct form

$$\begin{aligned} \min_{\alpha \in \mathcal{R}^l} \quad & \frac{1}{2} \alpha^T \mathbf{G} \alpha - \mathbf{e}^T \alpha \\ \text{subject to} \quad & \mathbf{0} \leq \alpha \leq C\mathbf{e}. \end{aligned} \tag{2.19}$$

The Hession $\mathbf{G}$ is an $l \times l$ matrix with its entries

$$G_{i,j} = \begin{cases} \frac{k}{k-1} K_{i,j}, & \text{if } y_i = y_j, \\ \frac{-k}{(k-1)^2} K_{i,j}, & \text{if } y_i \neq y_j, \end{cases} \tag{2.20}$$

where we abbreviate the kernel value $\kappa(\mathbf{x}_i, \mathbf{x}_j) \equiv \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j)$ for $K_{i,j}$. Furthermore, let $\mathbf{V}$ be an $l \times k$ matrix defined as:

$$V_{i,j} = \begin{cases} 1, & \text{if } y_i = j, \\ \frac{-1}{k-1}, & \text{if } y_i \neq j. \end{cases} \tag{2.21}$$

We have

$$\mathbf{G} = \mathbf{K} \odot (\mathbf{V}\mathbf{V}^T). \tag{2.22}$$

Since the kernel matrix $\mathbf{K}$ and $\mathbf{V}\mathbf{V}^T$ are both positive semi-definite, so is their Hadamard product $\mathbf{G}$. In the next section, we show that, up to a change of variables, (2.19) is equivalent to the dual problem of Crammer and Singer's approach with an additional constraint.

In order to avoid the division operation in the kernel computation of (2.20), we further let $\bar{\alpha} = \frac{k}{(k-1)^2}\alpha$. Therefore, we rewrite (2.19) as

$$\begin{aligned} \min_{\bar{\alpha} \in \mathcal{R}^l} \quad & \frac{1}{2}\bar{\alpha}^T \bar{\mathbf{G}} \bar{\alpha} - \mathbf{e}^T \bar{\alpha} \\ \text{subject to} \quad & 0 \leq \bar{\alpha} \leq \bar{C}, \end{aligned} \tag{2.23}$$

where $\bar{C} = \frac{k}{(k-1)^2}C$, and

$$\bar{G}_{i,j} = \begin{cases} (k-1)K_{i,j}, & \text{if } y_i = y_j, \\ -K_{i,j}, & \text{if } y_i \neq y_j. \end{cases} \tag{2.24}$$

As for the decision function, from (2.17) we obtain

$$\begin{aligned} f_m^*(\mathbf{x}) &= \sum_{i:y_i=m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x}) - \frac{1}{k-1} \sum_{i:y_i \neq m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x}) \\ &= \frac{k}{k-1} \sum_{i:y_i=m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x}) - \underbrace{\frac{1}{k-1} \sum_{i=1}^{l} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x})}_{\text{the same for all } f_m^*}. \end{aligned} \tag{2.25}$$

Finally, the resulting decision function can be simplified as

$$\arg\max_m f_m^*(\mathbf{x}) = \arg\max_m \sum_{i:y_i=m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x}). \tag{2.26}$$

Thus, each class model is built upon its own support vectors.

Table 2.2 Dataset description

Dataset	#Training data (l)	#Test data	#Class (k)	#Attr (d)
Iris	150	0	3	4
Wine	178	0	3	13
Glass	214	0	6	9
Vowel	528	0	11	10
Segment	2,310	0	7	19
Waveform	5,000	0	3	21
DNA	2,000	1,186	3	180
Satimage	4,435	2,000	6	36
Letter	15,000	5,000	26	16
USPS	7,291	2,007	10	256

2.3.4 Example: A Comparative Study

In this section, we validate the accuracy and efficiency of the new SimMSVM algorithm on several multi-class pattern recognition problems including **Iris**, **Wine**, **Glass**, **Vowel**, **Segment**, and **Waveform** from the **UCI** repository [1], **DNA**, **Satimage**, **Letter** from **Statlog** collection [19], and **USPS** [17]. For all the datasets except **DNA**, we linearly scale each attribute to be in the range $[-1, 1]$. Table 2.2 gives a detailed description about these datasets. Here we compare the new SimMSVM with some classical multi-class algorithms in terms of classification and computation time. Moreover, we also analyze the scaling behavior of the SimMSVM with the sample size and the number of classes in terms of training time. All the experiments are performed using the tool of BSVM [16] on a 1.50 GHz Intel Core^{TM}2 Duo PC running Windows XP with 1 GB main memory.

2.3.4.1 Classification Performance Comparisons

The tool of BSVM is used in the experiments. BSVM is developed to efficiently solve the bound-constrained problem (2.4). We find that the only difference between (2.19) and (2.4) is the Hessian of the objective function. To proceed the SimMSVM algorithm and take advantage of its sophisticated implementations, we only need to modify the kernel evaluation part of BSVM. For the one-versus-one (1V1) and one-versus-rest (1VR) approaches, we formulate their binary subproblems by (2.3) and solve each of them using BSVM. For the two so-called all-together methods proposed by Weston and Watkins and Crammer and Singer, their major disadvantage remains to the enormous size of the optimization problems. To tackle these difficulties, Hsu and Lin proposed decomposition methods for these two approaches, respectively. The implementations are integrated into BSVM as well.

In conclusion, we adopt BSVM for three reasons: (1) BSVM uses a simple working set selection which leads to a faster convergence for difficult cases.

The use of a special implementation of the optimization solver allows BSVM to stably identify bounded variables. (2) The implementations of Weston and Watkins' approach [38] and Crammer and Singer's approach [7] are integrated into BSVM. We can formulate the binary subproblems of the one-versus-one and one-versus-rest approaches and solve each of them using BSVM. (3) For the SimMSVM algorithm, we just modify the kernel evaluation part of BSVM. Furthermore, it should be stated that as described in Hsu and Lin [14], the BSVM software includes a decomposition implementation of Crammer and Singer's approach Crammer and Singer [7] for multi-class problems. The other three approaches (one-versus-one, one-versus-rest, and Weston and Watkins' approach) are also based on BSVM for the fair comparison. Therefore, the experiments of this manuscript are all implemented in a fair condition.

The most important criterion for evaluating the performance of the multi-class SVM is the classification accuracy. For all the five methods, we use the gaussian RBF kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right), \quad \gamma \geq 0.$$

We follow the strategy used in Hsu and Lin [14] and use model selection to get the optimal parameters. The hyper-parameter space is explored on a two-dimensional grid with $\gamma = [10^{-2}, 10^{-1}, 10^{0}, 10^{1}]$ and the regularization parameter $C = [10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}]$. We use two criteria to estimate the generalized accuracy.

For the used **Iris**, **Wine**, **Glass**, **Vowel**, **Segment**, and **Waveform**, we randomly subdivide each dataset into ten disjoint subsets. Then for each procedure, one of the subsets is used for testing and the others for training, where the tenfold cross-validation strategy is employed in the training set so as to select the optimal parameter C and γ. In order to evaluate the variability over different random splits for each dataset, the whole procedure above is repeated five times and the final experimental results are averaged. For the used **DNA**, **Satimage**, **Letter**, and **USPS** where both training and testing sets are available, we randomly draw the subsets of the original training set for training in order to evaluate the variability over different choices of the training set on each dataset. Here the 70 % data points are randomly chosen from the original training set, where the tenfold cross-validation is used for parameter selection on the selected training set. Then we train on the 70 % dataset with the optimal pair (C, γ) and predict for the test set. The whole procedure above is repeated five times and the results are averaged. We present the comparison results in Table 2.3, where "W&W" denotes the method proposed by Weston and Watkins and "C&S" indicates Crammer and Singers approach.

Table 2.3 shows the optimal classification accuracy and their corresponding number of support vectors (SV). Both the average standard deviation of the classification accuracy and the number of SV are also reported in Table 2.3 so as to evaluate the statistical significance of the cross-validation results and the number of SV are not integers since they are the average values of the tenfold cross-validation. From Table 2.3, we can find that the SimMSVM gives competitive accuracies with

Table 2.3 Classification accuracy (%) and the number of SV comparison between 1VR, 1V1, W&W, C&S, and SimMSVM (best rates boldfaced)

	1VR		1V1		W&W		C&S		SimMSVM	
Dataset	ACC	SV	ACC	SV	YACC	YSV	ACC	SV	ACC	SV
Iris	95.47	43.0	95.47	40.1	96.00	33.3	95.33	35.7	**96.53**	18.0
	±5.62	±17.5	±5.46	±24.5	±5.04	±18.8	±5.08	±23.6	±4.52	±15.4
Wine	96.82	64.2	**97.76**	70.8	97.53	64.9	97.06	64.0	97.18	101.6
	±4.15	±25.9	±3.74	±17.4	±3.58	±28.4	±3.41	±36.4	±3.80	±15.0
Glass	65.90	140.7	68.67	128.8	67.81	141.3	69.05	152.8	**71.62**	136.6
	±9.69	±10.7	±10.50	±8.2	±10.02	±19.2	±10.77	±17.1	±10.36	±4.2
Vowel	97.88	348.2	**99.31**	346.0	98.65	312.8	98.88	367.1	99.08	403.9
	±2.37	±5.5	±1.27	±4.8	±1.61	±60.8	±1.61	±64.0	±1.24	±4.1
Segment	96.73	483.5	97.23	444.4	97.07	338.5	97.31	626.1	**97.33**	1106.3
	±0.93	±43.9	±0.96	±53.7	±0.87	±57.8	±0.83	±384.8	±0.91	±16.5
Waveform	86.38	1843.6	**87.06**	2374.6	87.00	2187.1	86.75	2061.2	86.64	2870.0
	±1.77	±180.7	±1.16	±302.1	±1.17	±285.3	±1.52	±149.9	±1.32	±8.5
DNA	93.79	808.4	94.69	734.4	94.77	708.2	**94.95**	660.2	93.81	458.6
	±0.59	±10.3	±0.32	±84.4	±0.34	±72.6	±0.31	±37.3	±0.23	±4.7
Satimage	89.20	1312.2	**91.21**	1233.2	90.72	1122.8	90.60	1467.4	90.57	2731.0
	±0.44	±18.9	±0.50	±23.8	±0.31	±34.8	±0.24	±735.3	±0.41	±8.6
Letter	94.01	4464.4	**97.04**	5439.4	96.52	3541.8	96.39	3424.8	96.37	8126.2
	±0.24	±13.9	±0.16	±22.7	±0.21	±30.6	±0.17	±21.1	±0.14	±27.3
USPS	93.49	1348.6	94.96	1247.2	94.83	974.4	94.80	2486.2	**95.00**	3349.2
	±0.27	±14.4	±0.22	±10.8	±0.41	±20.2	±0.52	±138.2	±0.37	±17.5

the method by Crammer and Singer. In detail, we observe that: (1) not all the 1VR and 1V1 approaches outperform "all together" approaches (i.e., W&W, C&S, and SimMSVM) in the used ten datasets; (2) all the five implemented methods give the comparable results. The experimental results in the previous work such as Hsu and Lin [14] also suggest that no one is statistically better than the others for the 1VR and 1V1 approaches and all together approaches.

Since the test time of the W&W, C&S, and SimMSVM depends on the number of SV, we analyze their decision function expressions. Suppose that there are kernel function $\kappa(\mathbf{x}_i, \mathbf{x})$ and let k and n_{SV} be the number of class and SVs. The computational complexity of Eq. (2.25) (i.e. $\arg\max_m f_m^*(\mathbf{x}) = \arg\max_m \sum_{i:y_i=m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x})$) is $O(n_{SV})$ since each SV is operated once for testing. According to Hsu and Lin [14], Eq. (2.6) (i.e., $\arg\max_m f_m(\mathbf{x}) = \arg\max_m(\mathbf{w}_m^T \varphi(\mathbf{x}) + b_m)$) can be rewritten as

$$\arg\max_m f_m(\mathbf{x}) = \arg\max_m \sum_{i=1}^{l} (c_i^m A_i - \alpha_i^m)(\kappa(\mathbf{x}_i, \mathbf{x}) + 1)$$

where $A_i = \sum_m \alpha_i^m$, $c_i^m = 1$, if $y_i = m$ and $c_i^m = 0$, otherwise. Furthermore, Eq. (2.8) (i.e., $\arg\max_m f_m(\mathbf{x}) = \arg\max_m \mathbf{w}_m^T \varphi(\mathbf{x})$) can be rewritten as

$$\arg\max_m f_m(\mathbf{x}) = \arg\max_m \sum_{i=1}^{l} \alpha_i^m \kappa(\mathbf{x}_i, \mathbf{x})$$

Therefore, the computational complexity of Eqs. (2.6) and (2.8) is $O(kn_{SV})$. A quantitative test time comparison of the used datasets among these methods is also given in Table 2.4. From this table, the SimMSVM has a comparable test time to the multi-class SVM methods W&W and C&S.

2.3.4.2 Computational Complexity Comparisons

We have claimed that the SimMSVM achieves a significant speed-up compared with the method by Crammer and Singer for its simplified model. To demonstrate it, we report the average training time of the above model selection procedures. Due to the limit of the space here, we select the six out of ten datasets for comparison, e.g., **Vowel**, **Segment**, **Waveform**, **Satimage**, **Letter**, and **USPS**. For the six datasets, we report the average cross-validation time of the five competitive methods as shown in Figs. 2.2 and 2.3. Each row of Figs. 2.2 and 2.3 has the parameter γ with the value 10^{-2}, 10^{-1}, 10^0, and 10^1, respectively, and the C varies from 10^{-1} to 10^3 in a log scale. For the parameter pair (C, γ) for each learning algorithm, we report the average time on ten runs of the cross-validation. Most of the standard deviation values are below 0.05 in the experiments.

Figures 2.2 and 2.3 show that the SimMSVM has the best computational cost for most of the parameter combinations. For example, the 18 out of 20

Table 2.4 Testing time (in second) and the number of SV comparison between W&W, C&S, and SimMSVM

Dataset (#Class)	W&W			C&S			SimMSVM		
	Time	SV	$\frac{\text{Time}}{\text{SV}}$	Time	SV	$\frac{\text{Time}}{\text{SV}}$	Time	SV	$\frac{\text{Time}}{\text{SV}}$
Iris	2.44×10^{-4}	33.3	7.31×10^{-6}	3.69×10^{-4}	35.7	10.34×10^{-6}	0.63×10^{-4}	18.0	3.52×10^{-6}
(3)	$\pm 1.34 \times 10^{-4}$	± 18.8		$\pm 2.44 \times 10^{-4}$	± 23.6		$\pm 0.54 \times 10^{-4}$	± 15.4	
Wine	5.31×10^{-4}	64.9	8.19×10^{-6}	7.48×10^{-4}	64.0	11.69×10^{-6}	4.08×10^{-4}	101.6	4.01×10^{-6}
(3)	$\pm 2.30 \times 10^{-4}$	± 28.4		$\pm 4.26 \times 10^{-4}$	± 36.4		$\pm 0.64 \times 10^{-4}$	± 15.0	
Glass	35.40×10^{-4}	141.3	25.07×10^{-6}	44.30×10^{-4}	152.8	28.97×10^{-6}	6.75×10^{-4}	136.6	4.94×10^{-6}
(6)	$\pm 4.89 \times 10^{-4}$	± 19.2		$\pm 4.91 \times 10^{-4}$	± 17.1		$\pm 0.21 \times 10^{-4}$	± 4.2	
Vowel	3.87×10^{-2}	312.8	12.37×10^{-5}	4.83×10^{-2}	367.1	13.17×10^{-5}	0.49×10^{-2}	403.9	1.21×10^{-5}
(11)	$\pm 0.76 \times 10^{-2}$	± 60.8		$\pm 0.84 \times 10^{-2}$	± 64.0		$\pm 0.00 \times 10^{-2}$	± 4.1	
Segment	1.12×10^{-1}	338.5	32.98×10^{-5}	2.34×10^{-1}	626.1	37.32×10^{-5}	0.60×10^{-1}	1106.3	5.39×10^{-5}
(7)	$\pm 0.19 \times 10^{-1}$	± 57.8		$\pm 1.44 \times 10^{-1}$	± 384.8		$\pm 0.01 \times 10^{-1}$	± 16.5	
Waveform	0.52	2187.1	2.4×10^{-4}	0.72	2061.2	3.4×10^{-4}	0.34	2870.0	1.2×10^{-4}
(3)	± 0.07	± 285.3		± 0.05	± 149.9		± 0.00	± 8.5	
DNA	0.63	708.2	9×10^{-4}	0.82	660.2	12×10^{-4}	0.28	458.6	6×10^{-4}
(3)	± 0.01	± 72.6		± 0.01	± 37.3		± 0.00	± 4.7	
Satimage	2.66	1122.8	24×10^{-4}	3.07	1467.4	21×10^{-4}	1.28	2731.0	5×10^{-4}
(6)	± 0.31	± 34.8		± 0.24	± 735.3		± 0.41	± 8.6	
Letter	107.1	3541.8	30×10^{-3}	103.7	3424.8	30×10^{-3}	9.5	8126.2	1×10^{-3}
(26)	± 0.6	± 30.6		± 0.8	± 21.1		± 0.0	± 27.3	
USPS	4.19	974.4	43×10^{-4}	16.46	2486.2	66×10^{-4}	1.59	3349.2	4×10^{-4}
(10)	± 0.07	± 20.2		± 0.17	± 138.2		± 0.01	± 17.5	

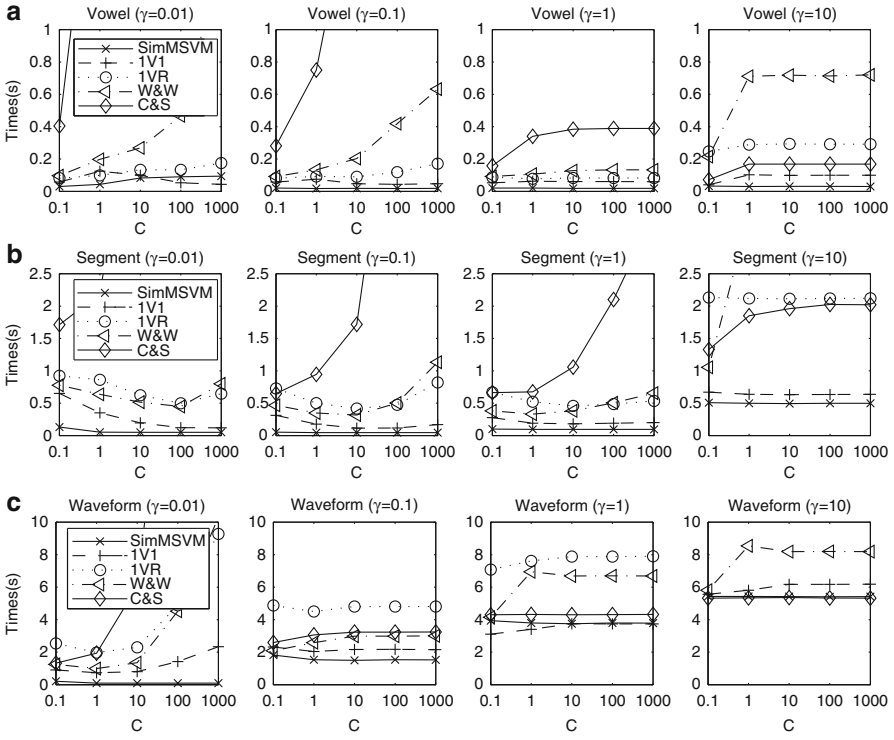

Fig. 2.2 Cross-validation time (in seconds) comparison as a function of C for $\gamma = 0.01,\ 0.1,\ 1,$ and 10 on (**a**) Vowel dataset, (**b**) Segment dataset, (**c**) Waveform dataset

parameter combinations in **Vowel** dataset by the SimMSVM outperform the other methods. All the parameter combinations in **Satimage** dataset by SimMSVM lead to better performance. Totally, the 81.7 % combinations by SimMSVM are the most efficient one for the used six datasets. In fact, it gives a significant speed-up from Crammer and Singer's approach (95.8 % combinations by SimMSVM perform better). Although the BSVM has applied decomposition methods for the two "all-together" approaches, their training speed remains slow especially for the relatively small γ. Since we find that the optimal parameters (C, γ) are in various ranges for different classification cases in the experiments, it is necessary to carry out more parameter pairs to obtain the optimal model. If the training speed is an important issue, the SimMSVM could be an option.

2.3.4.3 Speed vs. Data Size and Number of Class

In this subsection, we give a detailed analysis for the effect of the number of classes and the training set size on the SimMSVM. We compare the SimMSVM with the other multi-class SVM approaches on two datasets with the large training samples:

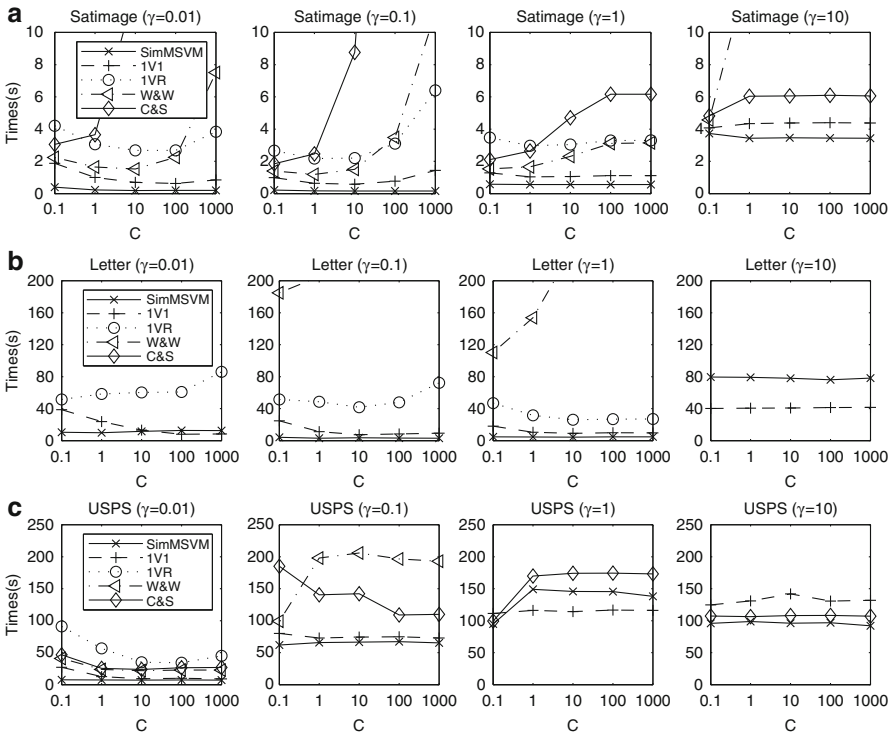

Fig. 2.3 Cross-validation time (in seconds) comparison as a function of C for $\gamma = 0.01,\ 0.1,\ 1,$ and 10 on (**a**) Satimage dataset, (**b**) Letter dataset, (**c**) USPS dataset

Letter and **USPS**. First, we restrict the analysis to binary-class case for the first two classes, then proceed to a multi-class analysis by gradually adding a new class. As described in Figs. 2.2 and 2.3, different kernel parameters (C, γ) would cause different computational time. For fairness, we choose the default parameter values of the BSVM package for the comparison: $C = 1, \gamma = 1/k$. Figure 2.4 summarizes the results.

For the two "all-together" multi-class SVM approaches, we have to solve an $(l \times k)$-variable QP. For the one-versus-rest approach, instead we need to solve k l-variable QPs. As for the SimMSVM, we solve a single l-variable QP, which is not affected by k. Figure 2.4 validates that the training time of SimMSVM scales slowly as the number of classes and dataset increase. We show an increasing training set size by keeping the number of class number ($k = 26$ for Letter and $k = 10$ for USPS) constant and adding training samples to each class. Each learning algorithm is repeated with five times and we report the average training time in Fig. 2.5. We further show an increase of class number by keeping the training set size constant. In the experiments, we set the constant training set size to be $5,000$

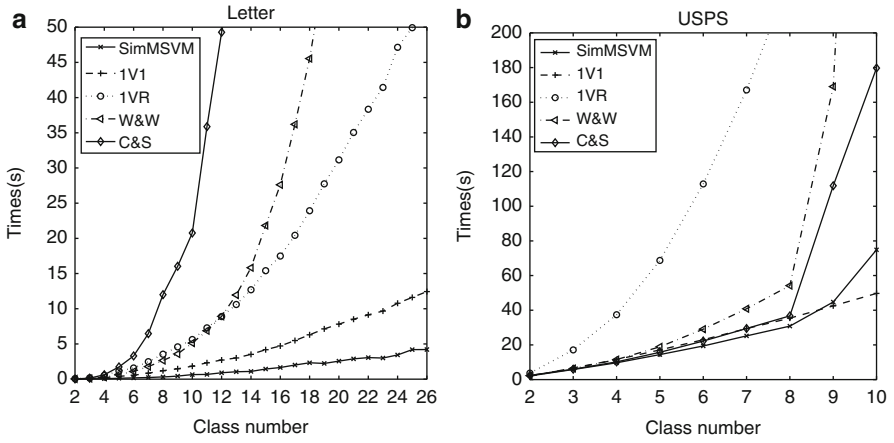

Fig. 2.4 Training time (seconds) vs. class number on (**a**) Letter dataset, (**b**) USPS dataset

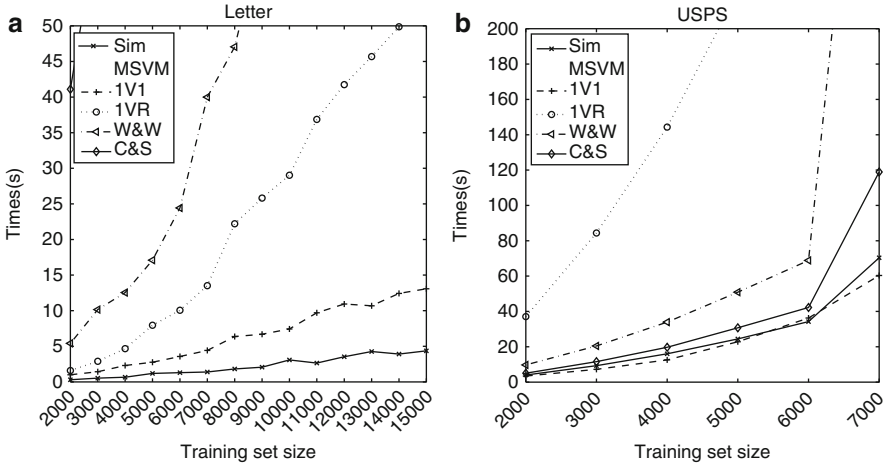

Fig. 2.5 Training time (seconds) vs. increasing training set size by keeping constant the class number on (**a**) Letter dataset, (**b**) USPS dataset

for Letter and 3,000 for USPS. The number of the training samples of each class is reduced accordingly while the class number increases. Figure 2.6 summarizes the results.

From Figs. 2.4, 2.5, and 2.6, it can be found that the SimMSVM takes the least computational cost in most cases except the **USPS**. In the **USPS**, the SimMSVM has a comparable computational cost to that of the one-versus-one method. The reason for this phenomenon is that the optimization for the dual problem of SVM with the 1V1 is much lower than that of the SimMSVM in the large-scale case. In this case, the advantage of the SimMSVM might be counteracted with that of the optimization

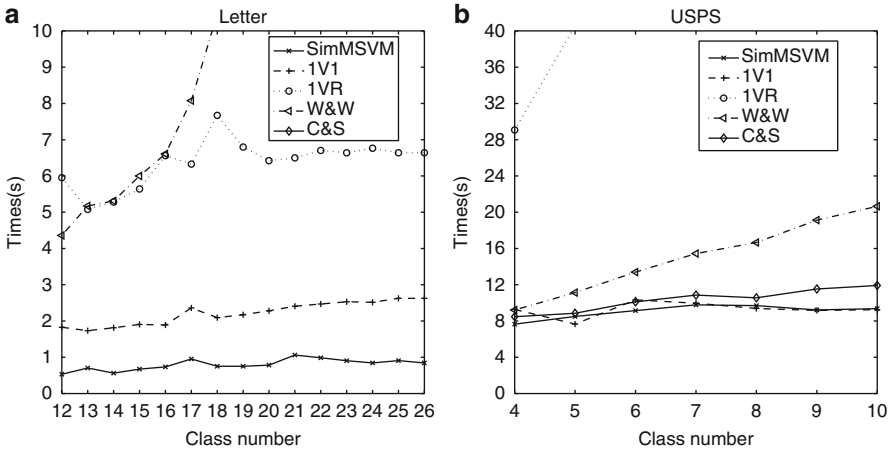

Fig. 2.6 Training time (seconds) vs. increasing class number by keeping constant the training set size on (**a**) Letter dataset, (**b**) USPS dataset

for the traditional SVM dual problem with 1V1. Compared with the other three algorithms (one-versus-rest, the C&S, and the W&W), the SimMSVM here has a significant advantage, which can demonstrate the efficiency of SimMSVM.

2.4 Discussions for SimMSVM

2.4.1 Relation to Binary SVM

In binary case where $k = 2$, from (2.17), the SimMSVM approach will produce $\mathbf{w}_1$ and $\mathbf{w}_2$ where $\mathbf{w}_1 = -\mathbf{w}_2$. Let $\mathbf{w} = 2\mathbf{w}_1$ and target $+1$ for the positive class and -1 negative, (2.15) is equivalent to

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{H},\xi\in\mathcal{R}^l} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + 2C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i\mathbf{w}^T\mathbf{x}_i \geq 1-\xi_i, \\ & \xi_i \geq 0, \quad i = 1,\ldots,l, \end{aligned} \tag{2.27}$$

which is the unbiased binary SVM formulation.

Lee et al. [22] proposed a variant of multi-class SVM by modifying the target values so that the positive class has target $+1$ and the negative class has target $-\frac{1}{k-1}$. For notational convenience, we define $\mathbf{v}_m$ for $m = 1,\ldots,k$ as a k-dimensional vector with 1 in the m-th coordinate and $-\frac{1}{k-1}$ elsewhere. If the label of a training example $\mathbf{x}_i$ is coded in vector form as $\mathbf{v}_{y_i}$, the margin constraints of (2.15) can be rewritten as

$$\mathbf{v}_{y_i}^T \mathbf{W}^T \varphi(\mathbf{x}_i) \geq 1 - \xi_i, \tag{2.28}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_k]$. Thus, it can be found that SimMSVM can be degenerated into the unbiased binary SVM.

Note that the transpose of the label vector $\mathbf{v}_{y_i}$ is exactly the i-th row of $\mathbf{V}$ defined in (2.21). From (2.22), the Hessian $\mathbf{G}$ contains not only a similarity measure between data samples but also their label information. In addition, we find that SimMSVM can be easily extended to the multi-label classification framework in which an instance may be simultaneously relevant to several labels. Multi-label classification is useful for text categorization, multimedia retrieval, and many other areas. In SimMSVM, we solve a multi-label classification problem by using the same dual formulation (2.19) except for a different representation of the label vector $\mathbf{v}_{y_i}$. For example, if a training example is relevant to r out of k labels simultaneously, we can set its label vector so that the relevant labels have target $+\frac{1}{r}$ and the other labels $-\frac{1}{k-r}$, where the values of the k labels are taken into two categories with $+\frac{1}{r}$ and $-\frac{1}{k-r}$. We then compute the Hessian $\mathbf{G}$ from (2.21) and solve the dual problem (2.19) to obtain the optimal α^*.

The resulting discriminant function is given as follows:

$$f_m^*(\mathbf{x}) = \sum_{i=1}^{l} v_{y_i,m} \alpha_i^* \kappa(\mathbf{x}_i, \mathbf{x}), \tag{2.29}$$

where $v_{y_i,m}$ is the m-th element of $\mathbf{v}_{y_i}$. Since the values of the k labels are taken into two categories with $+\frac{1}{r} > 0$ and $-\frac{1}{k-r} < 0$, the label set Y for the test sample $\mathbf{x}$ is determined as:

$$Y = \{m | f_m^*(\mathbf{x}) > 0, m \in \{1 \ldots k\}\} \tag{2.30}$$

2.4.2 Relation to Crammer and Singer's Multi-Class SVM

In [7], the dual problem is given as follows:

$$\begin{aligned} \min_{\alpha \in \mathcal{R}^{l \times k}} \quad & \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \sum_{m=1}^{k} K_{i,j} \alpha_{i,m} \alpha_{j,m} - \sum_{i=1}^{l} \alpha_{i,y_i} \\ \text{subject to} \quad & \sum_{m=1}^{k} \alpha_{i,m} = 0, \\ & \alpha_{i,m} \leq 0 \quad \text{for all } m \neq y_i, \\ & \alpha_{i,y_i} \leq C, \\ & i = 1, 2, \ldots, l, \quad m = 1, \ldots, k. \end{aligned} \tag{2.31}$$

Let $\mathbf{M}$ be the $l \times k$ dual matrix whose (i,m)-th element is the dual variable $\alpha_{i,m}$, and let

$$\alpha = [\alpha_{1,y_1}, \alpha_{2,y_2}, \ldots, \alpha_{l,y_l}]^T. \tag{2.32}$$

We then rewrite the dual objective function of (2.31) as

$$\frac{1}{2}\mathrm{tr}(\mathbf{M}^T\mathbf{K}\mathbf{M}) - \mathbf{e}^T\alpha. \tag{2.33}$$

From the linear constraint of (2.31), every k dual variables associated with the same x_i follows the sum-to-zero constraint. If $\alpha_{i,y_i} = 0$ for some $\mathbf{x}_i$, we have $\alpha_{i,m} = 0$ for all $m \neq y_i$. We carry over the notion of support vectors to the multi-class setting, and define support vectors as examples with $[\alpha_{i,1}, \ldots, \alpha_{i,k}] \neq \mathbf{0}$. A support vector $\mathbf{x}_i$ plays a positive role in its class model since $\alpha_{i,yi} > 0$, while punishing some other classes for those $\alpha_{i,m} < 0$. In order to reduce the size of dual variables, we further add to (2.31) a seemingly aggressive constraint that all those $\alpha_{i,m}$ $(m \neq y_i)$ corresponding to the i-th sample are of the same value (share equal punishment):

$$\alpha_{i,m} = -\frac{1}{k-1}\alpha_{i,y_i}, \text{ for all } m \neq y_i. \tag{2.34}$$

Thus, the sum-to-zero constraint of (2.31) is satisfied accordingly. As for the objective function of (2.31), from (2.34) we have

$$\sum_{m=1}^{k} \alpha_{i,m}\alpha_{j,m} = \begin{cases} \frac{k}{k-1}\alpha_i\alpha_j, & \text{if } y_i = y_j, \\ \frac{-k}{(k-1)^2}\alpha_i\alpha_j, & \text{if } y_i \neq y_j, \end{cases} \tag{2.35}$$

where we simply denote α_{i,y_i} as α_i, for $i = 1, \ldots, l$.

To sum up, Crammer and Singer's dual formulation (2.31) with additional constraint (2.34) is exactly the same as (2.19), which is the dual of SimMSVM.

2.4.3 Relation to One-Class SVM

There are also One-class SVMs, which solve an unsupervised learning problem related to probability density estimation. Two approaches that extend the SVM methodology have been proposed in [30,35]. The dual problem of the support vector domain description (SVDD) method described in Tax and Duin, [35] using RBF kernel is as follows:

$$\begin{aligned} \min_{\alpha \in \mathcal{R}^l} \quad & \frac{1}{2}\alpha^T\mathbf{K}\alpha - \mathbf{e}^T\alpha \\ \text{subject to} \quad & \mathbf{e}^T\alpha = 0, \\ & 0 \leq \alpha \leq C. \end{aligned} \tag{2.36}$$

If we ignore the equality constraint of (2.36), the only difference between (2.19) and (2.36) is the Hessian of the quadratic objective function. In fact, SVDD treats all training samples involved as one class, and the support vectors (those $\mathbf{x}_i$ with $\alpha_i > 0$) usually appear at the boundaries of data distribution. The next lemma proves that the Hessian $\mathbf{G} = \mathbf{K} \odot (\mathbf{P}\mathbf{P}^T)$ of (2.19) will endow the SimMSVM with discrimination power between all classes.

Lemma 1. *By solving the dual optimization problem (2.19), the decision value for the m-th class* $f_m(\mathbf{x}) = \sum_{i:y_i=m} \alpha_i \kappa(\mathbf{x_i}, \mathbf{x})$ *is not trivially 0 when the RBF kernel is adopted, i.e., there is at least one support vector for each class.*

Proof. Prove by contradiction. From the Karush—Kuhn—Tucker optimality condition of (2.19), a vector α is a stationary point of (2.19) if and only if there are two nonnegative vectors λ and μ such that

$$\begin{aligned} &\mathbf{G}\alpha - \mathbf{e} = \lambda - \mu, \\ &\lambda^T \alpha = 0,\ \mu^T (C\mathbf{e} - \alpha) = 0, \\ &\lambda \geq \mathbf{0},\ \mu \geq \mathbf{0}. \end{aligned}$$

We rewrite this condition as

$$\begin{aligned} &(\mathbf{G}\alpha)_i > 1 \quad \text{if } \alpha_i = 0, \\ &(\mathbf{G}\alpha)_i = 1 \quad \text{if } 0 < \alpha_i < C, \\ &(\mathbf{G}\alpha)_i < 1 \quad \text{if } \alpha_i = C. \end{aligned}$$

Assume that the m-th class has no support vectors, i.e., $\alpha_i = 0$ for all i where $y_i = m$. Since (2.20) indicates that $G_{ij} \leq 0$ for all $y_i \neq y_j$ when using a gaussian RBF kernel, we have $(\mathbf{G}\alpha)_i = \sum_{j:y_j \neq m} G_{ij}\alpha_j \leq 0$, which contradicts the above KKT condition. □

2.4.4 Fisher Consistency Issue

In binary case where the class label $y \in \{-1, +1\}$, we denote $P_+(\mathbf{x}) = P(Y = 1 | \mathbf{X} = \mathbf{x})$. Then a binary loss function $\xi(f(\mathbf{x}), y)$ is Fisher consistent if the minimizer of $E[\xi(f(\mathbf{X}), Y) | \mathbf{X} = \mathbf{x})]$ is the same as $\text{sign}(P_+(\mathbf{x}) - \frac{1}{2})$. In other words, Fisher consistency requires that the loss function should yield the Bayes decision boundary asymptotically [3, 23, 24]. In the multi-class setting where $y \in \{1, \ldots, k\}$, we let $P_m(\mathbf{x}) = P(Y = j | \mathbf{X} = \mathbf{x})$. Suppose $\xi(\mathbf{f}(\mathbf{x}), y)$ is a multi-class loss function, we denote the minimizer of $E(\xi(\mathbf{f}(\mathbf{X}, Y) | \mathbf{X} = \mathbf{x})$ as $\mathbf{f}^* = (f_1^*, \ldots, f_k^*)$. Fisher consistency requires $\text{argmax}_m f_m^* = \text{argmax}_m P_m$.

In the next lemma, we show that the loss (2.14) is Fisher inconsistent. For remedy, we introduce additional constraints to force the loss function of SimMSVM to be Fisher consistent.

Lemma 2. *The loss (2.14) is Fisher inconsistent when* $k \geq 3$.

Proof. From the above Fisher consistency definition, we have

$$E\left\{\left[1-f_Y(\mathbf{X})+\frac{1}{k-1}\sum_{M\neq Y} f_M(\mathbf{X})\right]_+\right\}$$
$$= E\left\{\sum_{y=1}^{k} P_y(\mathbf{X})\left[1-f_y(\mathbf{X})+\frac{1}{k-1}\sum_{m\neq y} f_m(\mathbf{X})\right]_+\right\}.$$

For any fixed $\mathbf{X}=\mathbf{x}$, the goal is to minimize

$$\sum_{y=1}^{k} P_y(\mathbf{x})\left[1-f_y(\mathbf{x})+\frac{1}{k-1}\sum_{m\neq y} f_m(\mathbf{x})\right]_+. \tag{2.37}$$

We let

$$g_y(\mathbf{x}) = f_y(\mathbf{x}) - \frac{1}{k-1}\sum_{m\neq y} f_m(\mathbf{x}), \tag{2.38}$$

for $y=1,\ldots,k$. Therefore, (2.37) is equivalent to

$$\sum_{y=1}^{k} P_y(\mathbf{x})\left[1-g_y(\mathbf{x})\right]_+. \tag{2.39}$$

Clearly, $\sum_{y=1}^{k} g_y(\mathbf{x}) = 0$. From the proof of **Lemma 1** in [24], we know that the minimizer $\mathbf{g}^*$ satisfies $g_m^* \leq 1, \forall m = 1,\ldots,k$. Thus, the problem (2.37) reduces to

$$\begin{aligned} \max_{\mathbf{g}} \quad & \sum_{m=1}^{k} P_m(\mathbf{x}) g_m(\mathbf{x}) \\ \text{subject to} \quad & \sum_{m=1}^{k} g_m(\mathbf{x}) = 0, \\ & g_m(\mathbf{x}) \leq 1, \quad m = 1,\ldots,k. \end{aligned} \tag{2.40}$$

It is easy to verify that the solution of (2.40) satisfies $g_m^*(\mathbf{x}) = -(k-1)$ if $m = \operatorname{argmin}_j P_j(\mathbf{x})$ and 1 otherwise. We then rewrite (2.38) as

$$f_y(\mathbf{x}) = \frac{k-1}{k}\left(g_y(\mathbf{x}) + A\right), \tag{2.41}$$

where $A = \frac{1}{k-1}\sum_{m=1}^{k} f_m(\mathbf{x})$. We have

$$\text{argmax}_m f^*(\mathbf{x}) = \text{argmax}_m g^*(\mathbf{x}). \tag{2.42}$$

Therefore, the loss function (2.14) is not Fisher consistent when $k \geq 3$. □

Although Fisher consistency is a desirable condition, a consistent loss may not always lead to better classification accuracy [15, 24, 29]. However, if Fisher consistency is required for the loss function of SimMSVM, one alternative solution is to add further constraints

$$(k-1)f_y(\mathbf{x}) - \sum_{m \neq y} f_m(\mathbf{x}) + 1 \geq 0, \tag{2.43}$$

for $y = 1, \ldots, k$. These constraints (2.43) amount to $g_m(\mathbf{x}) \geq -\frac{1}{k-1}$. We have the following Lemma:

Lemma 3. *The maximizer* $\mathbf{g}^*$ *of* $E[g_Y(\mathbf{X})|\mathbf{X} = \mathbf{x}]$, *subject to* $\sum_{m=1}^{k} g_m(\mathbf{x}) = 0$ *and* $-\frac{1}{k-1} \leq g_m(\mathbf{x}) \leq 1$, $\forall m$, *satisfies the following:* $g_m^*\mathbf{x}) = 1$ *if* $m = \text{argmax}_m P_m(\mathbf{x})$ *and* $-\frac{1}{k-1}$ *otherwise.*

For the detailed proof, please see Lemma 5 in [24]. □

Lemma 3 justifies that, with additional constraints (2.43), the loss function of SimMSVM is Fisher consistent. Fisher consistency is an attractive property for a loss function, the related Fisher consistency issue can be studied detailedly in the future work.

2.5 Conclusion

In this chapter, we discuss the main approaches for the multi-class SVM and especially introduce a new SimMSVM algorithm that directly solves a multi-class classification problem. Through modifying Crammer and Singer's multi-class SVM by introducing a relaxed classification error bound, the SimMSVM reduces the size of the dual variables from $l \times k$ to l, where l and k are the size of training data and the number of classes, respectively. We here prove that the dual formulation of the proposed SimMSVM is exactly the dual of Crammer and Singer's approach with an additional constraint. The experimental evaluations on real-world datasets show that the new SimMSVM approach can greatly speed-up the training process and achieve competitive or better classification accuracies.

References

1. Asuncion, A., Newman, D.: UCI machine learning repository. http://archive.ics.uci.edu/ml/datasets.html (2007)
2. Baldi, P., Pollastri, G.: A machine-learning strategy for protein analysis. IEEE Intell. Syst. **17**(2), 28–35 (2002)
3. Bartlett, P., Jordan, M., McAuliffe, J.: Convexity, classification, and risk bounds. J. Am. Stat. Assoc. **101**, 138–156 (2006)
4. Bredensteiner, E., Bennett, K.: Multicategory classification by support vector machines. Comput. Optim. Appl. **12**, 53–79 (1999)
5. Chawla, N.V., Japkowicz, N., Kolcz, A.: Editorial: special issue on learning from imbalanced data sets. ACM SIGKDD Explor. **6**(1), 1–6 (2004)
6. Cortes, C., Vapnik, V.: Support vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. J. Mach. Learn. Res. **2**, 265–292 (2001)
8. Dietterich, T., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. **2**, 263–286 (1995)
9. Fung, G., Mangasarian, O.: Proximal support vector machine classifiers. In: Provost, F., Srikant, R. (eds.) Proceedings KDD-2001: Knowledge Discovery and Data Mining, August 26–29, 2001, San Francisco, CA, pp. 77–86. Asscociation for Computing Machinery, New York (2001)
10. Fung, G., Mangasarian, O.: Multicategory proximal support vector machine classifiers. Mach. Learn. **59**(1–2), 77–97 (2005)
11. Ganapathiraju, A., Hamaker, J., Picone, J.: Applications of support vector machines to speech recognition. IEEE Trans. Signal Process. **52**(8), 2348–2355 (2004)
12. Guermeur, Y.: Combining discriminant models with new multi-class svms. Pattern Anal. Appl. **5**(2), 168–179 (2002)
13. He, X., Wang, Z., Jin, C., Zheng, Y., Xue, X.Y.: A simplified multi-class support vector machine with reduced dual optimization. Pattern Recognit. Lett. **33**, 71–82 (2012)
14. Hsu, C., Lin, C.: A comparison of methods for multi-class support vector machines. IEEE Trans. Neural Netw. **13**, 415–425 (2002)
15. Hsu, C., Lin, C.: A simple decomposition method for support vector machines. Mach. Learn. **46**, 291–314 (2002)
16. Hsu, C., Lin, C.: Bsvm. http://mloss.org/software/view/62/ (2008)
17. Hull, J.: A database for handwritten text recognition research. IEEE Trans. Pattern Anal. Mach. Intell. **16**(5), 550–554 (1994)
18. Khan, L., Awad, M., Thuraisingham, B.: A new intrusion detection system using support vector machines and hierarchical clustering. VLDB J. **16**(4), 507–521 (2007)
19. King, R., Feng, C., Sutherland, A.: Statlog: comparison of classification algorithms on large real-world problems. Appl. Artif. Intell. **9**(3), 289–333 (1995)
20. Knerr, S., Personnaz., L., Dreyfus, G.: Single-layer learning revisited: a stepwise procedure for building and training neural network. In: Fogelman, J. (ed.) Neurocomputing: Algorithms, Architectures and Applications. Springer, Berlin (1990)
21. Kreßel, U.: Pairwise classification and support vector machines. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods: Support Vector Learning, pp. 255–268. MIT Press, Cambridge (1999)
22. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines. In: Wegman, E., Braverman, A., Goodman, A., Smyth, P. (eds.) Computing Science and Statistics, vol. 33, pp. 498–512. Interface Foundation of North America, Inc., Fairfax Station, VA, USA (2002)
23. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines: theory and application to the classification of microarray data and satellite radiance data. J. Am. Stat. Assoc. **99**, 67–81 (2004)

24. Liu, Y.: Fisher consistency of multicategory support vector machines. In: Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07) (2007)
25. Mangasarian, O., Musicant, D.: Successive overrelaxation for support vector machines. IEEE Trans. Neural Netw. **10**(5), 1032–1037 (1999)
26. Mangasarian, O., Musicant, D.: Lagrangian support vector machines. J. Mach. Learn. Res. **1**, 161–177 (2001)
27. Mori, S., Suen, C., Yamamoto, K.: Historical review of OCR research and development, pp. 244–273. IEEE Computer Society Press, Los Alamitos (1995)
28. Platt, J., Cristianini, N., Shawe-Taylor, J.: Large margin dags for multiclass classification. In: Advances in Neural Information Processing Systems, vol. 12, pp. 547–553. MIT Press, Cambridge (2000)
29. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. J. Mach. Learn. Res. **5**, 101–141 (2004)
30. Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
31. Suykens, J., Vandewalle, J.: Least squares support vector machine classifiers. Neural Process. Lett. **9**(3), 293–300 (1999)
32. Suykens, J., Vandewalle, J.: Multiclass least squares support vector machines. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN99). World Scientific, Washington, DC (1999)
33. Szedmak, S., Shawe-Taylor, J., Saunders, C., Hardoon, D.: Multiclass classification by l1 norm support vector machine. In: Pattern Recognition and Machine Learning in Computer Vision Workshop (2004)
34. Tang, Y., Zhang, Y., Chawla, N., Krasser, S.: Svms modeling for highly imbalanced classification. IEEE Trans. Syst. Man Cybern. Part B **39**(1), 281–288 (2009)
35. Tax, D., Duin, R.: Data domain description using support vectors. In: Proceedings of the European Symposium on Artificial Neural Networks, pp. 251–256 (1999)
36. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
37. Wang, L., Shen, X.: On l1-norm multiclass support vector machines: methodology and theory. J. Am. Stat. Assoc. **102**, 583–594 (2007)
38. Weston, J., Watkins, C.: Multi-class support vector machines. In: Proceedings of ESANN99 (1999)
39. Xia, X., Li, K.: A sparse multi-class least-squares support vector machine. In: IEEE International Symposium on Industrial Electronics, 2008 (ISIE 2008), pp. 1230–1235 (2008)

http://www.springer.com/978-3-319-02299-4

Support Vector Machines Applications

Ma, Y.; Guo, G. (Eds.)

2014, VII, 302 p. 87 illus., 56 illus. in color., Hardcover
ISBN: 978-3-319-02299-4