Multi-Class Optimal Margin Distribution Machine

Teng Zhang [1] **Zhi-Hua Zhou** [1]

Abstract

Recent studies disclose that maximizing the minimum margin like support vector machines does not necessarily lead to better generalization performances, and instead, it is crucial to optimize the margin distribution. Although it has been shown that for binary classification, characterizing the margin distribution by the first- and second-order statistics can achieve superior performance. It still remains open for multi-class classification, and due to the complexity of margin for multi-class classification, optimizing its distribution by mean and variance can also be difficult. In this paper, we propose mcODM (multi-class Optimal margin Distribution Machine), which can solve this problem efficiently. We also give a theoretical analysis for our method, which verifies the significance of margin distribution for multi-class classification. Empirical study further shows that mcODM always outperforms all four versions of multi-class SVMs on all experimental data sets.

1. Introduction

Support vector machines (SVMs) and Boosting have been two mainstream learning approaches during the past decade. The former (Cortes & Vapnik, 1995) roots in the statistical learning theory (Vapnik, 1995) with the central idea of searching a large *margin* separator, i.e., maximizing the smallest distance from the instances to the classification boundary in a RKHS (reproducing kernel Hilbert space). It is noteworthy that there is also a long history of applying margin theory to explain the latter (Freund & Schapire, 1995; Schapire et al., 1998), due to its tending to be empirically resistant to over-fitting (Reyzin & Schapire, 2006; Wang et al., 2011; Zhou, 2012).

[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. Correspondence to: Zhi-Hua Zhou <zhouzh@lamda.nju.edu.cn>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

Recently, the margin theory for Boosting has finally been defended (Gao & Zhou, 2013), and has disclosed that the margin distribution rather than a single margin is more crucial to the generalization performance. It suggests that there may still exist large space to further enhance for SVMs. Inspired by this recognition, (Zhang & Zhou, 2014; 2016) proposed a binary classification method to optimize margin distribution by characterizing it through the first- and second-order statistics, which achieves quite satisfactory experimental results. Later, (Zhou & Zhou, 2016) extends the idea to an approach which is able to exploit unlabeled data and handle unequal misclassification cost. A brief summary of this line of early research can be found in (Zhou, 2014).

Although it has been shown that for binary classification, optimizing the margin distribution by maximizing the margin mean and minimizing the margin variance simultaneously can get superior performance, it still remains open for multi-class classification. Moreover, the margin for multi-class classification is much more complicated than that for binary class classification, which makes the resultant optimization be a difficult non-differentiable non-convex programming. In this paper, we propose mcODM (multi-class Optimal margin Distribution Machine) to solve this problem efficiently. For optimization, we relax mcODM into a series of convex quadratic programming (QP), and extend the Block Coordinate Descent (BCD) algorithm (Tseng, 2001) to solve the dual of each QP. The sub-problem of each iteration of BCD is also a QP. By exploiting its special structure, we derive a sorting algorithm to solve it which is much faster than general QP solvers. We further provide a generalization error bound based on Rademacher complexity, and further present the analysis of the relationship between generalization error and margin distribution for multi-class classification. Extensive experiments on twenty two data sets show the superiority of our method to all four versions of multi-class SVMs.

The rest of this paper is organized as follows. Section 2 introduces some preliminaries. Section 3 formulates the problem. Section 4 presents the proposed algorithm. Section 5 discusses some theoretical analyses. Section 6 reports on our experimental studies and empirical observations. Finally Section 7 concludes with future work.

2. Preliminaries

We denote by $\mathcal{X} \in \mathbb{R}^d$ the instance space and $\mathcal{Y} = [k]$ the label set, where $[k] = \{1, \ldots, k\}$. Let $\mathcal{D}$ be an unknown (underlying) distribution over $\mathcal{X} \times \mathcal{Y}$. A training set $S = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m$ is drawn identically and independently (i.i.d.) according to distribution $\mathcal{D}$. Let $\phi : \mathcal{X} \mapsto \mathbb{H}$ be a feature mapping associated to some positive definite kernel κ. For multi-class classification setting, the hypothesis is defined based on k weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k \in \mathbb{H}$, where each weight vector $\boldsymbol{w}_l, l \in \mathcal{Y}$ defines a scoring function $\boldsymbol{x} \mapsto \boldsymbol{w}_l^\top \phi(\boldsymbol{x})$ and the label of instance $\boldsymbol{x}$ is the one resulting in the largest score, i.e., $h(\boldsymbol{x}) = \operatorname{argmax}_{l \in \mathcal{Y}} \boldsymbol{w}_l^\top \phi(\boldsymbol{x})$. This decision function naturally leads to the following definition of the margin for a labeled instance $(\boldsymbol{x}, y)$:

$$\gamma_h(\boldsymbol{x}, y) = \boldsymbol{w}_y^\top \phi(\boldsymbol{x}) - \max_{l \neq y} \boldsymbol{w}_l^\top \phi(\boldsymbol{x}).$$

Thus h misclassifies $(\boldsymbol{x}, y)$ if and only if it produces a negative margin for this instance.

Given a hypothesis set H of functions mapping $\mathcal{X}$ to $\mathcal{Y}$ and the labeled training set S, our goal is to learn a function $h \in H$ such that the generalization error $R(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[1_{h(\boldsymbol{x})\neq y}]$ is small.

3. Formulation

To design optimal margin distribution machine for multi-class classification, we need to understand how to optimize the margin distribution. (Gao & Zhou, 2013) proved that, to characterize the margin distribution, it is important to consider not only the margin mean but also the margin variance. Inspired by this idea, (Zhang & Zhou, 2014; 2016) proposed the optimal margin distribution machine for binary classification, which characterizes the margin distribution according to the first- and second-order statistics, that is, maximizing the margin mean and minimizing the margin variance simultaneously. Specifically, let $\bar{\gamma}$ denote the margin mean, and the optimal margin distribution machine can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{w},\bar{\gamma},\xi_i,\epsilon_i} \quad & \Omega(\boldsymbol{w}) - \eta\bar{\gamma} + \frac{\lambda}{m}\sum_{i=1}^{m}(\xi_i^2 + \epsilon_i^2) \\ \text{s.t.} \quad & \gamma_h(\boldsymbol{x}_i, y_i) \geq \bar{\gamma} - \xi_i, \\ & \gamma_h(\boldsymbol{x}_i, y_i) \leq \bar{\gamma} + \epsilon_i, \ \forall i, \end{aligned}$$

where $\Omega(\boldsymbol{w})$ is the regularization term to penalize the model complexity, η and λ are trading-off parameters, ξ_i and ϵ_i are the deviation of the margin $\gamma_h(\boldsymbol{x}_i, y_i)$ to the margin mean. It's evident that $\sum_{i=1}^{m}(\xi_i^2 + \epsilon_i^2)/m$ is exactly the margin variance.

By scaling $\boldsymbol{w}$ which doesn't affect the final classification results, the margin mean can be fixed as 1, then the deviation of the margin of $(\boldsymbol{x}_i, y_i)$ to the margin mean is $|\gamma_h(\boldsymbol{x}_i, y_i) - 1|$, and the optimal margin distribution machine can be reformulated as

$$\begin{aligned} \min_{\boldsymbol{w},\xi_i,\epsilon_i} \quad & \Omega(\boldsymbol{w}) + \frac{\lambda}{m}\sum_{i=1}^{m}\frac{\xi_i^2 + \mu\epsilon_i^2}{(1-\theta)^2} \\ \text{s.t.} \quad & \gamma_h(\boldsymbol{x}_i, y_i) \geq 1 - \theta - \xi_i, \\ & \gamma_h(\boldsymbol{x}_i, y_i) \leq 1 + \theta + \epsilon_i, \ \forall i. \end{aligned}$$

where $\mu \in (0, 1]$ is a parameter to trade off two different kinds of deviation (larger or less than margin mean). $\theta \in [0, 1)$ is a parameter of the zero loss band, which can control the number of support vectors, i.e., the sparsity of the solution, and $(1-\theta)^2$ in the denominator is to scale the second term to be a surrogate loss for 0-1 loss.

For multi-class classification, let the regularization term $\Omega(\boldsymbol{w}) = \sum_{l=1}^{k} \|\boldsymbol{w}_l\|_{\mathbb{H}}^2/2$ and combine with the definition of margin, and we arrive at the formulation of mcODM,

$$\begin{aligned} \min_{\boldsymbol{w}_l,\xi_i,\epsilon_i} \quad & \frac{1}{2}\sum_{l=1}^{k}\|\boldsymbol{w}_l\|_{\mathbb{H}}^2 + \frac{\lambda}{m}\sum_{i=1}^{m}\frac{\xi_i^2 + \mu\epsilon_i^2}{(1-\theta)^2} \\ \text{s.t.} \quad & \boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - \max_{l \neq y_i} \boldsymbol{w}_l^\top \phi(\boldsymbol{x}_i) \geq 1 - \theta - \xi_i, \\ & \boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - \max_{l \neq y_i} \boldsymbol{w}_l^\top \phi(\boldsymbol{x}_i) \leq 1 + \theta + \epsilon_i, \ \forall i. \end{aligned} \tag{1}$$

where λ, μ and θ are the parameters for trading-off described previously.

4. Optimization

Due to the $\max$ operator in the second constraint, mcODM is a non-differentiable non-convex programming, which is quite difficult to solve directly.

In this section, we first relax mcODM into a series of convex quadratic programming (QP), which can be much easier to handle. Specifically, at each iteration, we recast the first constraint as $k-1$ linear inequality constraints:

$$\boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - \boldsymbol{w}_l^\top \phi(\boldsymbol{x}_i) \geq 1 - \theta - \xi_i, \ l \neq y_i,$$

and replace the second constraint with

$$\boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - M_i \leq 1 + \theta + \epsilon_i,$$

where $M_i = \max_{l \neq y_i} \bar{\boldsymbol{w}}_l^\top \phi(\boldsymbol{x}_i)$ and $\bar{\boldsymbol{w}}_l$ is the solution to the previous iteration. Then we can repeatedly solve the following convex QP problem until convergence:

$$\begin{aligned} \min_{\boldsymbol{w}_l,\xi_i,\epsilon_i} \quad & \frac{1}{2}\sum_{l=1}^{k}\|\boldsymbol{w}_l\|_{\mathbb{H}}^2 + \frac{\lambda}{m}\sum_{i=1}^{m}\frac{\xi_i^2 + \mu\epsilon_i^2}{(1-\theta)^2} \\ \text{s.t.} \quad & \boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - \boldsymbol{w}_l^\top \phi(\boldsymbol{x}_i) \geq 1 - \theta - \xi_i, \ \forall l \neq y_i, \\ & \boldsymbol{w}_{y_i}^\top \phi(\boldsymbol{x}_i) - M_i \leq 1 + \theta + \epsilon_i, \ \forall i. \end{aligned} \tag{2}$$

Introduce the lagrange multipliers $\zeta_i^l \geq 0, l \neq y_i$ for the first $k-1$ constraints and $\beta_i \geq 0$ for the last constraint respectively, the Lagrangian function of Eq. 2 leads to

$$\begin{aligned} & L(\boldsymbol{w}_l, \xi_i, \epsilon_i, \zeta_i^l, \beta_i) \\ = & \frac{1}{2}\sum_{l=1}^{k}\|\boldsymbol{w}_l\|_{\mathbb{H}}^2 + \frac{\lambda}{m}\sum_{i=1}^{m}\frac{\xi_i^2+\mu\epsilon_i^2}{(1-\theta)^2} \\ & - \sum_{i=1}^{m}\sum_{l\neq y_i}\zeta_i^l(\boldsymbol{w}_{y_i}^\top\phi(\boldsymbol{x}_i) - \boldsymbol{w}_l^\top\phi(\boldsymbol{x}_i) - 1 + \theta + \xi_i) \\ & + \sum_{i=1}^{m}\beta_i(\boldsymbol{w}_{y_i}^\top\phi(\boldsymbol{x}_i) - M_i - 1 - \theta - \epsilon_i), \end{aligned}$$

By setting the partial derivations of variables $\{\boldsymbol{w}_l, \xi_i, \epsilon_i\}$ to zero, we have

$$\begin{aligned} \boldsymbol{w}_l &= \sum_{i=1}^{m}\left(\delta_{y_i,l}\sum_{s\neq y_i}\zeta_i^s - (1-\delta_{y_i,l})\zeta_i^l - \delta_{y_i,l}\beta_i\right)\phi(\boldsymbol{x}_i), \\ \xi_i &= \frac{m(1-\theta)^2}{2\lambda}\sum_{l\neq y_i}\zeta_i^l, \quad \epsilon_i = \frac{m(1-\theta)^2}{2\lambda\mu}\beta_i. \end{aligned} \tag{3}$$

where $\delta_{y_i,l}$ equals 1 when $y_i = l$ and 0 otherwise. We further simplify the expression of $\boldsymbol{w}_l$ as

$$\boldsymbol{w}_l = \sum_{i=1}^{m}(\alpha_i^l - \delta_{y_i,l}\beta_i)\phi(\boldsymbol{x}_i), \tag{4}$$

by defining $\alpha_i^l \equiv -\zeta_i^l$ for $\forall l \neq y_i$ and $\alpha_i^{y_i} \equiv \sum_{s\neq y_i}\zeta_i^s$ and substituting Eq. 4 and Eq. 3 into the Lagrangian function, then we have the following dual problem

$$\begin{aligned} \min_{\alpha_i^l, \alpha_i^{y_i}, \beta_i} \quad & \frac{1}{2}\sum_{l=1}^{k}\|\boldsymbol{w}_l\|_{\mathbb{H}}^2 + \frac{m(1-\theta)^2}{4\lambda}\sum_{i=1}^{m}(\alpha_i^{y_i})^2 \\ & + \frac{m(1-\theta)^2}{4\lambda\mu}\sum_{i=1}^{m}\beta_i^2 + (1-\theta)\sum_{i=1}^{m}\sum_{l\neq y_i}\alpha_i^l \\ & + (M_i + 1 + \theta)\sum_{i=1}^{m}\beta_i \\ \text{s.t.} \quad & \sum_{l=1}^{k}\alpha_i^l = 0, \ \forall i, \\ & \alpha_i^l \leq 0, \ \forall i, \forall l \neq y_i, \\ & \beta_i \geq 0, \ \forall i. \end{aligned} \tag{5}$$

The objective function in Eq. 5 involves $m(k+1)$ variables in total, so it is not easy to optimize with respect to all the variables simultaneously. Note that all the constraints can be partitioned into m disjoint sets, and the i-th set only involves $\alpha_i^1, \ldots, \alpha_i^k, \beta_i$, so the variables can be divided into m decoupled groups and an efficient block coordinate descent algorithm (Tseng, 2001) can be applied. Specifically, we sequentially select a group of $k+1$ variables $\alpha_i^1, \ldots, \alpha_i^k, \beta_i$ associated with instance $\boldsymbol{x}_i$ to minimize, while keeping other variables as constants, and repeat this procedure until convergence.

Algorithm 1 below details the kenrel mcODM.

Algorithm 1 Kenrel mcODM

1: **Input:** Data set S.
2: Initialize $\boldsymbol{\alpha}^\top = [\alpha_1^1, \ldots, \alpha_1^k, \ldots, \alpha_m^1, \ldots, \alpha_m^k]$ and $\boldsymbol{\beta}^\top = [\beta_1, \ldots, \beta_m]$ as zero vector.
3: **while** $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ not converge **do**
4: **for** $i = 1, \ldots, m$ **do**
5: $M_i \leftarrow \max_{l\neq y_i}\sum_{j=1}^{m}(\alpha_j^l - \delta_{y_j,l}\beta_j)\kappa(\boldsymbol{x}_j, \boldsymbol{x}_i)$.
6: **end for**
7: Solve Eq. 5 by block coordinate descent method.
8: **end while**
9: **Output:** $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$.

4.1. Solving the sub-problem

The sub-problem in step 7 of Algorithm 1 is also a convex QP with $k+1$ variables, which can be accomplished by some standard QP solvers. However, by exploiting its special structure, i.e., only a small quantity of cross terms are involved, we can derive an algorithm to solve this sub-problem just by sorting, which can be much faster than general QP solvers.

Note that all variables except $\alpha_i^1, \ldots, \alpha_i^k, \beta_i$ are fixed, so we have the following sub-problem:

$$\begin{aligned} \min_{\alpha_i^l, \alpha_i^{y_i}, \beta_i} \quad & \sum_{l\neq y_i}\frac{A}{2}(\alpha_i^l)^2 + \sum_{l\neq y_i}B_l\alpha_i^l + \frac{D}{2}(\alpha_i^{y_i})^2 - A\alpha_i^{y_i}\beta_i \\ & + B_{y_i}\alpha_i^{y_i} + \frac{E}{2}\beta_i^2 + F\beta_i \\ \text{s.t.} \quad & \sum_{l=1}^{k}\alpha_i^l = 0, \\ & \alpha_i^l \leq 0, \ \forall l \neq y_i, \\ & \beta_i \geq 0. \end{aligned} \tag{6}$$

where $A = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_i)$, $B_l = \sum_{j\neq i}\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)(\alpha_j^l - \delta_{y_j,l}\beta_j) + 1 - \theta$ for $\forall l \neq y_i$, $B_{y_i} = \sum_{j\neq i}\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)(\alpha_j^{y_i} - \delta_{y_j,y_i}\beta_j)$, $D = A + \frac{m(1-\theta)^2}{2\lambda}$, $E = A + \frac{m(1-\theta)^2}{2\lambda\mu}$ and $F \equiv M_i + 1 + \theta - B_{y_i}$.

The KKT conditions of Eq. 6 indicate that there are scalars ν, ρ_l and η such that

$$\sum_{l=1}^{k}\alpha_i^l = 0, \tag{7}$$

$$\alpha_i^l \leq 0, \ \forall l \neq y_i, \tag{8}$$

$$\beta_i \geq 0, \tag{9}$$
$$\rho_l \alpha_i^l = 0,\ \rho_l \geq 0,\ \forall l \neq y_i, \tag{10}$$
$$A\alpha_i^l + B_l - \nu + \rho_l = 0,\ \forall l \neq y_i, \tag{11}$$
$$\eta\beta_i = 0,\ \eta \geq 0, \tag{12}$$
$$-A\alpha_i^{y_i} + E\beta_i + F - \eta = 0, \tag{13}$$
$$D\alpha_i^{y_i} - A\beta_i + B_{y_i} - \nu = 0. \tag{14}$$

According to Eq. 8, Eq. 10 and Eq. 11 are equivalent to

$$A\alpha_i^l + B_l - \nu = 0, \quad \text{if } \alpha_i^l < 0,\ \forall l \neq y_i, \tag{15}$$
$$B_l - \nu \leq 0, \quad \text{if } \alpha_i^l = 0,\ \forall l \neq y_i. \tag{16}$$

In the same way, Eq. 12 and Eq. 13 are equivalent to

$$-A\alpha_i^{y_i} + E\beta_i + F = 0, \quad \text{if } \beta_i > 0, \tag{17}$$
$$-A\alpha_i^{y_i} + F \geq 0, \quad \text{if } \beta_i = 0. \tag{18}$$

Thus KKT conditions turn to Eq. 7 - Eq. 9 and Eq. 14 - Eq. 18. Note that

$$\alpha_i^l \equiv \min\left(0, \frac{\nu - B_l}{A}\right),\ \forall l \neq y_i, \tag{19}$$

satisfies KKT conditions Eq. 8 and Eq. 15 - Eq. 16 and

$$\beta_i \equiv \max\left(0, \frac{A\alpha_i^{y_i} - F}{E}\right), \tag{20}$$

satisfies KKT conditions Eq. 9 and Eq. 17 - Eq. 18. By substituting Eq. 20 into Eq. 14, we obtain

$$D\alpha_i^{y_i} + B_{y_i} - \nu = \max\left(0, \frac{A}{E}(A\alpha_i^{y_i} - F)\right). \tag{21}$$

Let's consider the following two cases in turn.

Case 1: $A\alpha_i^{y_i} \leq F$, according to Eq. 20 and Eq. 21, we have $\beta_i = 0$ and $\alpha_i^{y_i} = \frac{\nu - B_{y_i}}{D}$. Thus, $A\frac{\nu - B_{y_i}}{D} \leq F$, which implies that $\nu \leq B_{y_i} + \frac{DF}{A}$.

Case 2: $A\alpha_i^{y_i} > F$, according to Eq. 20 and Eq. 21, we have $\beta_i = \frac{A\alpha_i^{y_i} - F}{E} > 0$ and $\alpha_i^{y_i} = \frac{E\nu - AF - EB_{y_i}}{DE - A^2}$. Thus, $A\frac{E\nu - AF - EB_{y_i}}{DE - A^2} > F$, which implies that $\nu > B_{y_i} + \frac{DF}{A}$.

The remaining task is to find ν such that Eq. 7 holds. With Eq. 7 and Eq. 19, it can be shown that

$$\nu = \frac{\frac{AB_{y_i}}{D} + \sum_{l:\alpha_i^l<0} B_l}{\frac{A}{D} + |\{l|\alpha_i^l < 0\}|}, \quad \text{Case 1}, \tag{22}$$

$$\nu = \frac{\frac{AEB_{y_i} + A^2F}{DE - A^2} + \sum_{l:\alpha_i^l<0} B_l}{\frac{AE}{DE - A^2} + |\{l|\alpha_i^l < 0\}|}, \quad \text{Case 2}. \tag{23}$$

In both cases, the optimal ν takes the form of $(P + \sum_{l:\alpha_i^l<0} B_l)/(Q + |\{l|\alpha_i^l < 0\}|)$, where P and Q are some constants. (Fan et al., 2008) showed that it can be found by sorting $\{B_l\}$ for $\forall l \neq y_i$ in decreasing order and then sequentially adding them into an empty set Φ, until

$$\nu^* = \frac{P + \sum_{l\in\Phi} B_l}{Q + |\Phi|} \geq \max_{l\notin\Phi} B_l. \tag{24}$$

Note that the Hessian matrix of the objective function of Eq. 6 is positive definite, which guarantees the existence and uniqueness of the optimal solution, so only one of Eq. 22 and Eq. 23 can hold. We can first compute ν^* according to Eq. 24 for Case 1, and then check whether the constraint of ν is satisfied. If not, we further compute ν^* for Case 2. Algorithm 2 summarizes the pseudo-code for solving the sub-problem.

Algorithm 2 Solving the sub-problem

Input: Parameters A, $B = \{B_1, \ldots, B_k\}$, D, E, F.
Initialize $\hat{B} \leftarrow B$, then swap $\hat{B}_1$ and $\hat{B}_{y_i}$, and sort $\hat{B}\backslash\{\hat{B}_1\}$ in decreasing order.
$i \leftarrow 2$, $\nu \leftarrow AB_{y_i}/D$.
while $i \leq k$ and $\nu/(i - 2 + A/D) < \hat{B}_i$ **do**
  $\nu \leftarrow \nu + \hat{B}_i$.
  $i \leftarrow i + 1$.
end while
if $\nu \leq B_{y_i} + DF/A$ **then**
  $\alpha_i^l \leftarrow \min(0, (\nu - B_l)/A)$, $l \neq y_i$.
  $\alpha_i^{y_i} \leftarrow (\nu - B_{y_i})/D$.
  $\beta_i \leftarrow 0$.
else
  $i \leftarrow 2$, $\nu \leftarrow (AE\hat{B}_1 + A^2F)/(DE - A^2)$.
  while $i \leq k$ and $\nu/(i - 2 + AE/(DE - A^2)) < \hat{B}_i$ **do**
    $\nu \leftarrow \nu + \hat{B}_i$.
    $i \leftarrow i + 1$.
  end while
  $\alpha_i^l \leftarrow \min(0, (\nu - B_l)/A)$, $l \neq y_i$.
  $\alpha_i^{y_i} \leftarrow (E\nu - AF - EB_{y_i})/(DE - A^2)$.
  $\beta_i \leftarrow (A\alpha_i^{y_i} - F)/E$.
end if
Output: $\alpha_i^1, \ldots, \alpha_i^k, \beta_i$.

4.2. Speedup for linear kernel

In section 4.1, the proposed method can efficiently deal with kernel mcODM. However, the computation of M_i in step 5 of Algorithm 1 and the computation of parameters $\bar{B}_l$ in Algorithm 2 both involve the kernel matrix, whose inherent computational cost takes $O(m^2)$ time, so it might be computational prohibitive for large scale problems.

When linear kernel is used, these problems can be alleviated. According to Eq. 4, $\boldsymbol{w}$ is spanned by the training instance so it lies in a finite dimensional space under this

circumstance. By storing $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k$ explicitly, the computational cost of $M_i = \max_{l \neq y_i} \boldsymbol{w}_l^\top \boldsymbol{x}_i$ can be much less. Moreover, note that $\bar{B}_l = \sum_{j \neq i} \boldsymbol{x}_i^\top \boldsymbol{x}_j (\alpha_j^l - \delta_{y_j,l}\beta_j) = \sum_{j=1}^m \boldsymbol{x}_i^\top \boldsymbol{x}_j (\bar{\alpha}_j^l - \delta_{y_j,l}\bar{\beta}_j) - \boldsymbol{x}_i^\top \boldsymbol{x}_i (\bar{\alpha}_i^l - \delta_{y_i,l}\bar{\beta}_i) = \boldsymbol{w}_l^\top \boldsymbol{x}_i - A(\alpha_i^l - \delta_{y_i,l}\beta_i)$, so $\bar{B}_l$ can also be computed efficiently.

5. Analysis

In this section, we study the statistical property of mcODM. To present the generalization bound of mcODM, we need to introduce the following loss function Φ,

$$\Phi(z) = 1_{z \leq 0} + \frac{(z-1+\theta)^2}{(1-\theta)^2} 1_{0 < z \leq 1-\theta},$$
$$\gamma_{h,\theta}(\boldsymbol{x}, y) = \boldsymbol{w}_y^\top \phi(\boldsymbol{x}) - \max_{l \in \mathcal{Y}} \{\boldsymbol{w}_l^\top \phi(\boldsymbol{x}) - (1-\theta) 1_{l=y}\},$$

where $1_{(\cdot)}$ is the indicator function that returns 1 when the argument holds, and 0 otherwise. As can be seen, $\gamma_{h,\theta}(\boldsymbol{x}, y)$ is a lower bound of $\gamma_h(\boldsymbol{x}, y)$ and $\Phi(\gamma_{h,\theta}(\boldsymbol{x}, y)) = \Phi(\gamma_h(\boldsymbol{x}, y))$.

Theorem 1. *Let* $H = \{(\boldsymbol{x}, y) \in \mathcal{X} \times [k] \mapsto \boldsymbol{w}_y^\top \phi(\boldsymbol{x}) | \sum_{l=1}^k \|\boldsymbol{w}_l\|_{\mathbb{H}}^2 \leq \Lambda^2\}$ *be the hypothesis space of mcODM, where* $\phi : \mathcal{X} \mapsto \mathbb{H}$ *is a feature mapping induced by some positive definite kernel* κ*. Assume that* $S \subseteq \{\boldsymbol{x} : \kappa(\boldsymbol{x}, \boldsymbol{x}) \leq r^2\}$*, then for any* $\delta > 0$*, with probability at least* $1 - \delta$*, the following generalization bound holds for any* $h \in H$*,*

$$R(h) \leq \frac{1}{m} \sum_{i=1}^m \Phi(\gamma_h(\boldsymbol{x}_i, y_i)) + \frac{16r\Lambda}{1-\theta} \sqrt{\frac{2\pi k}{m}} + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2m}}.$$

Proof. Let $\tilde{H}_\theta$ be the family of hypotheses mapping $\mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ defined by $\tilde{H}_\theta = \{(\boldsymbol{x}, y) \mapsto \gamma_{h,\theta}(\boldsymbol{x}, y) : h \in H\}$, with McDiarmid inequality (McDiarmid, 1989), yields the following inequality with probability at least $1 - \delta$,

$$\mathbb{E}[\Phi(\gamma_{h,\theta}(\boldsymbol{x}, y))] \leq \frac{1}{m} \sum_{i=1}^m \Phi(\gamma_{h,\theta}(\boldsymbol{x}_i, y_i)) + 2\mathcal{R}_S(\Phi \circ \tilde{H}_\theta) + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2m}}, \forall h \in \tilde{H}_\theta.$$

Note that $\Phi(\gamma_{h,\theta}) = \Phi(\gamma_h)$, $R(h) = \mathbb{E}[1_{\gamma_h(\boldsymbol{x},y) \leq 0}] \leq \mathbb{E}[1_{\gamma_{h,\theta}(\boldsymbol{x},y) \leq 0}] \leq \mathbb{E}[\Phi(\gamma_{h,\theta}(\boldsymbol{x}, y))]$ and $\Phi(z)$ is $\frac{2}{1-\theta}$-Lipschitz function, by using Talagrand's lemma (Mohri et al., 2012), we have

$$R(h) \leq \frac{1}{m} \sum_{i=1}^m \Phi(\gamma_h(\boldsymbol{x}_i, y_i)) + \frac{4\mathcal{R}_S(\tilde{H}_\theta)}{1-\theta} + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2m}}.$$

According to Theorem 7 of (Lei et al., 2015), we have $\mathcal{R}_S(\tilde{H}_\theta) \leq 4r\Lambda\sqrt{2\pi k/m}$ and proves the stated result. □

Theorem 1 shows that we can get a tighter generalization bound for smaller $r\Lambda$ and smaller θ. Since $\gamma \leq 2r\Lambda$, so the former can be viewed as an upper bound of the margin. Besides, $1 - \theta$ is the lower bound of the zero loss band of mcODM. This verifies that better margin distribution (i.e., larger margin mean and smaller margin variance) can yield better generalization performance, which is also consistent with the work of (Gao & Zhou, 2013).

6. Empirical Study

In this section, we empirically evaluate the effectiveness of our method on a broad range of data sets. We first introduce the experimental settings and compared methods in Section 6.1, and then in Section 6.2, we compare our method with four versions of multi-class SVMs, i.e., mcSVM (Weston & Watkins, 1999; Crammer & Singer, 2001; 2002), one-versus-all SVM (ovaSVM), one-versus-one SVM (ovoSVM) (Ulrich, 1998) and error-correcting output code SVM (ecocSVM) (Dietterich & Bakiri, 1995). In addition, we also study the influence of the number of classes on generalization performance and margin distribution in Section 6.3. Finally, the computational cost is presented in Section 6.4.

6.1. Experimental Setup

We evaluate the effectiveness of our proposed methods on twenty two data sets. Table 1 summarizes the statistics of these data sets. The data set size ranges from 150 to more than 581,012, and the dimensionality ranges from 4 to more than 62,061. Moreover, the number of class ranges from 3 to 1,000, so these data sets cover a broad range of properties. All features are normalized into the interval $[0, 1]$. For each data set, eighty percent of the instances are randomly selected as training data, and the rest are used as testing data. For each data set, experiments are repeated for 10 times with random data partitions, and the average accuracies as well as the standard deviations are recorded.

mcODM is compared with four versions of multi-class SVMs, i.e., ovaSVM, ovoSVM, ecocSVM and mcSVM. These four methods can be roughly classified into two groups. The first group includes the first three methods by converting the multi-class classification problem into a set of binary classification problems. Specially, ovaSVM consists of learning k scoring functions $h_l : \mathcal{X} \mapsto \{-1, +1\}, l \in \mathcal{Y}$, each seeking to discriminate one class $l \in \mathcal{Y}$ from all the others, as can be seen it need train k SVM models. Alternatively, ovoSVM determines the scoring functions for all the combinations of class pairs, so it need train $k(k-1)/2$ SVM models. Finally, ecocSVM is a generalization of the former two methods. This technique assigns to each class $l \in \mathcal{Y}$ a code word with length c, which serves as a signature for this class. After training

Table 1. Characteristics of experimental data sets.

ID	Dataset	#Instance	#Feature	#Class	ID	Dataset	#Instance	#Feature	#Class
1	*iris*	150	4	3	12	*sector*	9,619	55,197	105
2	*wine*	178	13	3	13	*pendigits*	10,992	16	10
3	*glass*	214	9	6	14	*news20*	19,928	62,061	20
4	*svmguide2*	391	20	3	15	*letter*	20,000	16	26
5	*svmguide4*	612	10	6	16	*protein*	24,387	357	3
6	*vehicle*	846	18	4	17	*shuttle*	58,000	9	7
7	*vowel*	990	10	11	18	*connect-4*	67,557	126	3
8	*segment*	2,310	19	7	19	*mnist*	70,000	780	10
9	*dna*	3,186	180	3	20	*aloi*	108,000	128	1,000
10	*satimage*	6,435	36	6	21	*rcv1*	534,135	47,236	53
11	*usps*	9,298	256	10	22	*covtype*	581,012	54	7

c binary SVM models $h_1(\cdot), \ldots, h_c(\cdot)$, the class predicted for each testing instance is the one whose signatures is the closest to $[h_1(\boldsymbol{x}), \ldots, h_c(\boldsymbol{x})]$ in Hamming distance. The weakness of these methods is that they may produce unclassifiable regions and their computational costs are usually quite large in practice, which can be observed in the following experiments. On the other hand, mcSVM belongs to the second group. It directly determines all the scroing functions at once, so the time cost is usually less than the former methods. In addition, the unclassifiable regions are also resolved.

For all the methods, the regularization parameter λ for mcODM or C for binary SVM and mcSVM is selected by 5-fold cross validation from $[2^0, 2^2, \ldots, 2^{20}]$. For mcODM, the regularization parameters μ and θ are both selected from $[0.2, 0.4, 0.6, 0.8]$. For ecocSVM, the exhaustive codes strategy is employed, i.e., for each class, we construct a code of length $2^{k-1} - 1$ as the signature. All the selections for parameters are performed on training sets.

6.2. Results

Table 2 summarizes the detailed results on twenty two data sets. As can be seen, the overall performance of our method is superior or highly competitive to the other compared methods. Specifically, mcODM performs significantly better than mcSVM/ovaSVM/ovoSVM/ecocSVM on 17/19/18/17 over 22 data sets respectively, and achieves the best accuracy on 20 data sets. In addition, as can be seen, in comparing with other four methods which don't consider margin distribution, the win/tie/loss counts show that mcODM is always better or comparable, almost never worse than it.

6.3. Influence of the Number of Classes

In this section we study the influence of the number of classes on generalization performance and margin distribution, respectively.

6.3.1. Generalization Performance

Figure 1 plots the generalization performance of all the five methods on data set segment, and similar observation can be found for other data sets. As can be seen, when the number of classes is less than four, all methods perform quite well. However, as the fifth class is added, the generalization performance of other four methods decreases drastically. This might be attributable to the introduction of some noisy data, which SVM-style algorithms are very sensitive to since they optimize the minimum margin. On the other hand, our method considers the whole margin distribution, so it can be robust to noise and relatively more stable.

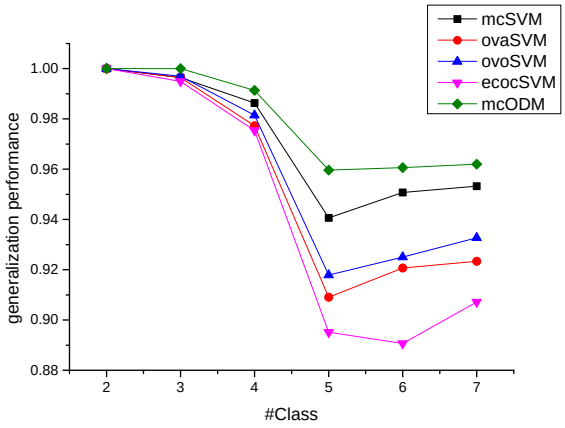

Figure 1. Generalization performance of all the five methods on data set segment with the increase of the number of classes.

6.3.2. Margin Distribution

Figure 2 plots the frequency histogram of margin distribution produced by mcSVM, ovaSVM and mcODM on data set segment as the number of classes increases from two to seven. As can be seen, when the number of classes is less than four, all methods can achieve good margin distribution, whereas with the increase of the number of classes, the other two methods begin to produce negative margins. At the same time, the distribution of our method becomes

Table 2. Accuracy (mean±std.) comparison on twenty two data sets. Linear kernel is used. The best accuracy on each data set is bolded. •/○ indicates the performance of mcODM is significantly better/worse than the compared methods (paired t-tests at 95% significance level). The average rank and top1 times are listed in the third and second row from the bottom. The win/tie/loss counts for mcODM are summarized in the last row. ovoSVM and ecocSVM did not return results on some data sets in 48 hours.

Dataset	mcSVM	ovaSVM	ovoSVM	ecocSVM	mcODM
iris	84.7±5.0•	82.0±5.0•	73.0±7.1•	73.3±4.2•	**87.3±3.4**
wine	96.5±2.6•	97.0±2.4	96.8±3.1	91.6±2.7•	**98.4±1.9**
glass	62.0±7.6•	57.8±2.1•	62.4±6.0•	60.4±4.0•	**68.2±7.3**
svmguide2	80.9±2.6•	81.5±2.3•	74.6±5.4•	79.0±3.6•	**84.1±1.8**
svmguide4	86.6±3.2•	77.2±3.1•	77.1±6.9•	74.2±4.3•	**87.8±3.5**
vehicle	80.5±2.1•	78.5±2.9•	73.5±3.5•	76.9±2.5•	**85.6±6.6**
vowel	65.4±2.1	44.1±2.6•	**71.1±4.9**	43.1±2.0•	65.6±3.9
segment	95.3±0.9•	92.3±0.7•	93.2±2.2•	90.7±0.8•	**96.7±1.3**
dna	95.1±0.6•	95.1±0.7•	93.0±1.0•	90.9±1.1•	**95.6±0.7**
satimage	81.8±0.5•	80.1±0.7•	77.4±4.2•	76.0±1.4•	**89.1±8.0**
usps	**95.1±0.3**	94.2±0.3•	94.5±0.6•	92.3±0.7•	95.1±0.6
sector	93.3±0.4•	93.1±0.5•	89.9±0.6•	N/A	**93.6±0.6**
pendigits	94.8±0.3•	91.8±0.5•	95.7±0.8•	87.6±0.7•	**96.7±0.9**
news20	83.3±0.3•	83.0±0.3•	69.4±0.4•	N/A	**83.6±0.3**
letter	76.9±0.4•	64.9±1.0•	75.7±1.2•	N/A	**82.7±1.6**
protein	68.7±0.3	68.7±0.4	68.1±0.3•	66.5±0.5•	**70.2±2.3**
shuttle	97.2±0.1•	89.9±2.8•	97.4±0.1•	84.2±0.0•	**99.6±0.0**
connect-4	75.7±0.1•	75.7±0.1•	75.7±0.2•	75.1±0.2•	**94.4±7.6**
mnist	92.5±0.2•	91.5±0.2•	90.9±0.5•	87.6±0.2•	**95.2±0.3**
aloi	90.1±0.2•	82.2±0.2•	N/A	N/A	**91.4±0.3**
rcv1	92.8±0.1	92.7±0.1	92.1±0.1•	N/A	**93.0±1.1**
covtype	71.7±0.1	71.2±0.1•	72.7±0.1	70.6±0.1•	**72.8±1.6**
Avg. Rank	2.45	3.18	3.45	4.82	1.09
mcODM: w/t/l	17/5/0	19/3/0	18/3/0	17/0/0	

"sharper", which prevents the instance with small margin, so our method can still perform relatively well as the number of classes increases, which is also consistent with the observation from Figure 1.

6.4. Time Cost

We compare the single iteration time cost of our method with mcSVM, ovaSVM, ovoSVM on all the data sets except aloi, on which ovoSVM could not return results in 48 hours. All the experiments are performed with MATLAB 2012b on a machine with 8×2.60 GHz CPUs and 32GB main memory. The average CPU time (in seconds) on each data set is shown in Figure 3. The binary SVM used in ovaSVM, ovoSVM and mcSVM are both implemented by the LIBLINEAR (Fan et al., 2008) package. It can be seen that for small data sets, the efficiency of all the methods are similar, however, for data sets with more than ten classes, e.g., sector and rcv1, mcSVM and mcODM, which learn all the scroing functions at once, are much faster than ovaSVM and ovoSVM, owing to the inefficiency of binary-decomposition as discussed in Section 6.1. Note that LIBLINEAR are very fast implementations of binary SVM and mcSVM, and this shows that our method is also computationally efficient.

7. Conclusions

Recent studies disclose that for binary class classification, maximizing the minimum margin does not necessarily lead to better generalization performances, and instead, it is crucial to optimize the margin distribution. However, it remains open to the influence of margin distribution for multi-class classification. We try to answer this question in this paper. After maximizing the margin mean and minimizing the margin variance simultaneously, the resultant optimization is a difficult non-differentiable non-convex programming. We propose mcODM to solve this problem efficiently. Extensive experiments on twenty two data sets validate the superiority of our method to four versions of multi-class SVMs. In the future it will be interesting to extend mcODM to more general learning settings, i.e., multi-label learning and structured learning.

Acknowledgements

This research was supported by the NSFC (61333014) and the Collaborative Innovation Center of Novel Software Technology and Industrialization. Authors want to thank reviewers for helpful comments, and thank Dr. Wei Gao for reading a draft.

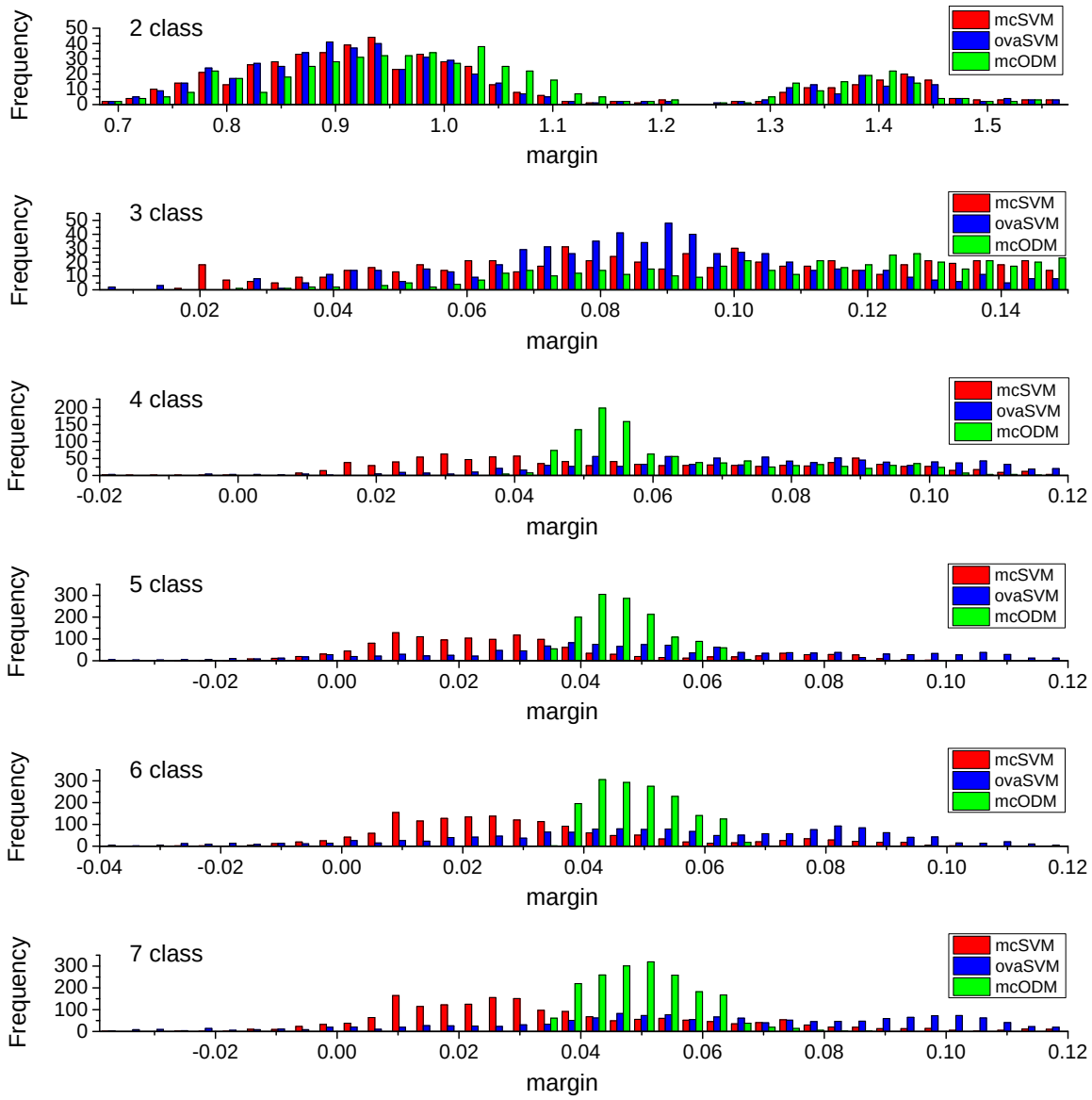

Figure 2. Frequency histogram of the margin distribution produced by mcSVM, ovaSVM and mcODM on data set segment as the number of classes increase from two to seven.

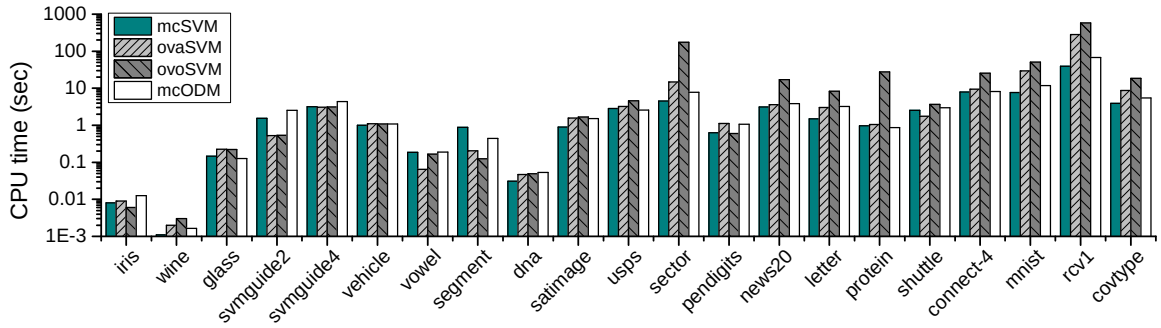

Figure 3. Single iteration time cost of mcSVM, ovaSVM, ovoSVM and mcODM on all the data sets except aloi.

References

Cortes, C. and Vapnik, V. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

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

Crammer, K. and Singer, Y. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2):201–233, 2002.

Dietterich, T. G. and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2(2):263–286, 1995.

Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., and Lin, C. J. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the 2nd European Conference on Computational Learning Theory*, pp. 23–37, Barcelona, Spain, 1995.

Gao, W. and Zhou, Z.-H. On the doubt about margin explanation of boosting. *Artificial Intelligence*, 203:1–18, 2013.

Lei, Y., rn Dogan, Binder, A., and Kloft, M. Multiclass svms: From tighter data-dependent generalization bounds to novel algorithms. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2026–2034. Curran Associates, Inc., 2015.

McDiarmid, C. On the method of bounded differences. *Surveys in Combinatorics*, 141(1):148–188, 1989.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. MIT Press, Cambridge, MA, 2012.

Reyzin, L. and Schapire, R. E. How boosting the margin can also boost classifier complexity. In *Proceedings of 23rd International Conference on Machine Learning*, pp. 753–760, Pittsburgh, PA, 2006.

Schapire, R. E., Freund, Y., Bartlett, P. L., and Lee, W. S. Boosting the margin: a new explanation for the effectives of voting methods. *Annuals of Statistics*, 26(5):1651–1686, 1998.

Tseng, P. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

Ulrich, H. G. Kreel. Pairwise classification and support vector machines. In Schlkopf, B., Burges, C. J. C., and Smola, A. J. (eds.), *Advances in Kernel Methods: Support Vector Machines*, pp. 255–268, Cambridge, MA, December 1998. MIT Press.

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

Wang, L. W., Sugiyama, M., Yang, C., Zhou, Z.-H., and Feng, J. A refined margin analysis for boosting algorithms via equilibrium margin. *Journal of Machine Learning Research*, 12:1835–1863, 2011.

Weston, J. and Watkins, C. Multi-class support vector machines. In *Proceedings of the European Symposium on Artificial Neural Networks*, Brussels, Belgium, 1999.

Zhang, T. and Zhou, Z.-H. Large margin distribution machine. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 313–322, New York, NY, 2014.

Zhang, T. and Zhou, Z.-H. Optimal margin distribution machine. *CoRR, abs/1604.03348*, 2016.

Zhou, Y.-H. and Zhou, Z.-H. Large margin distribution learning with cost interval and unlabeled data. *IEEE Transactions on Knowledge and Data Engineering*, 28 (7):1749–1763, 2016.

Zhou, Z.-H. *Ensemble Methods: Foundations and Algorithms*. CRC Press, Boca Raton, FL, 2012.

Zhou, Z.-H. Large margin distribution learning. In *Proceedings of the 6th IAPR International Workshop on Artificial Neural Networks in Pattern Recognition*, pp. 1–11, Montreal, Canada, 2014.