Maximum Margin Multi-Dimensional Classification

Bin-Bin Jia,[1,2,3] **Min-Ling Zhang**[1,3,4*]
[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China
[3]Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
[4]Collaborative Innovation Center of Wireless Communications Technology, China
{jiabb, zhangml}@seu.edu.cn

Abstract

Multi-dimensional classification (MDC) assumes *heterogenous* class spaces for each example, where class variables from different class spaces characterize semantics of the example along different dimensions. Due to the heterogeneity of class spaces, the major difficulty in designing margin-based MDC techniques lies in that the modeling outputs from different class spaces are not comparable to each other. In this paper, a first attempt towards maximum margin multi-dimensional classification is investigated. Following the one-vs-one decomposition within each class space, the resulting models are optimized by leveraging classification margin maximization on individual class variable and model relationship regularization across class variables. We derive convex formulation for the maximum margin MDC problem, which can be tackled with alternating optimization admitting QP or closed-form solution in either alternating step. Experimental studies over real-world MDC data sets clearly validate effectiveness of the proposed maximum margin MDC techniques.

Introduction

In multi-dimensional classification, each training example is represented by a single instance while associated with multiple class variables (Read, Bielza, and Larrañaga 2014; Ma and Chen 2018; Jia and Zhang 2019). Here, each *class variable* corresponds to one specific *class space* which characterizes the semantics of an object along one *dimension*. Many real-world problems can be naturally formalized under MDC frameworks (Rodríguez et al. 2012; Borchani et al. 2013; Sagarna et al. 2014; Serafino et al. 2015). For example, a news document can be characterized from the `topic` dimension (with possible classes *sports*, *politics*, *social*, *Sci&Tech*, etc.), from the `mood` dimension (with possible classes *good news*, *neutral news*, *bad news*), and from the `zone` dimension (with possible classes *domestic*, *intra-/inter-continental*, etc.).

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ be the d-dimensional input (feature) space and $\mathcal{Y} = C_1 \times C_2 \times \cdots \times C_q$ be the output space which corresponds to the Cartesian product of q class spaces. Here, each class space C_j $(1 \leq j \leq q)$ consists of K_j possible class labels, i.e., $C_j = \{c_1^j, c_2^j, \ldots, c_{K_j}^j\}$, among which only one is relevant to the example. Furthermore, let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq N\}$ be the MDC training set with N training examples, where $\boldsymbol{x}_i \in \mathcal{X}$ is a d-dimensional feature vector and $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ is the associated class vector, each of which is one possible value in the corresponding class space, i.e., $y_{ij} \in C_j$. Then, the task of multi-dimensional classification is to induce a predictive function $f : \mathcal{X} \mapsto \mathcal{Y}$ from $\mathcal{D}$ which can assign a proper class vector $f(\boldsymbol{x}) \in \mathcal{Y}$ for the unseen instance $\boldsymbol{x}$.

To accomplish the task of learning from MDC examples, the most intuitive strategy is to induce a number of independent multi-class classifiers, one per class space. However, this strategy completely ignores potential dependencies among class variables which would impact the generalization performance of induced predictive model. Therefore, most existing approaches try to model class dependencies in different ways, such as specifying chaining order over class variables (Zaragoza et al. 2011; Read, Martino, and Luengo 2014), assuming directed acyclic graph (DAG) structure over class variables (Bielza, Li, and Larrañaga 2011; Batal, Hong, and Hauskrecht 2013; Zhu, Liu, and Jiang 2016; Bolt and van der Gaag 2017; Gil-Begue, Larrañaga, and Bielza 2018; Benjumeda, Bielza, and Larrañaga 2018), and partitioning class variables into groups (Read, Bielza, and Larrañaga 2014), etc.

To derive margin-based techniques for multi-dimensional classification, the major difficulty lies in that the modeling outputs from different class spaces are not directly comparable. In this paper, we make a first attempt to adapt maximum margin technique for multi-dimensional classification, and propose a novel approach named M^{3}MDC, i.e., *MaxiMum Margin for Multi-Dimensional Classification*. Specifically, based on one-vs-one decomposition within each class space, the multi-dimensional classification models are optimized by maximizing classification margin on individual class variable and regularizing model relationship across class variables. The resulting convex formulation is solved with alternating optimization admitting QP or closed-form solution in either alternating step. Comparative studies against

[*]Corresponding author

other well-established MDC approaches clearly validate the effectiveness of the proposed approach.

The rest of this paper is organized as follows. Firstly, technical details of the proposed approach are introduced. Secondly, related works on MDC are briefly discussed. Thirdly, experimental results of comparative studies are reported. Finally, we conclude this paper.

The Maximum Margin MDC Approach

The common premise of margin-based approaches is that different modeling outputs are comparable. However, due to MDC's inherent property that each class variable corresponds to one *heterogenous* class space, the modeling outputs from different class spaces are not directly comparable. In this section, we present technical details of the M^{3}MDC approach which considers the margins between each pair of class labels in the same class space.

Following the same notations given in previous section, it is easy to know that there are totally $m = \sum_{j=1}^{q} \frac{K_j(K_j-1)}{2}$ pairs of class labels across all class spaces. To obtain margins between each pair of class labels, one-vs-one (OvO) decomposition is made accordingly. Without loss of generality, for the ith pair of class labels l_+^i and l_-^i, let $\mathcal{D}^i = \{(\boldsymbol{x}_j^i, y_j^i) \mid 1 \leq j \leq n_i\}$ be the corresponding OvO decomposition training set. Here, $\boldsymbol{x}_j^i \in \mathcal{X}$, y_j^i equals $+1$ when l_+^i is relevant and -1 when l_-^i is relevant, n_i is the number of training examples in $\mathcal{D}$ for which either l_+^i or l_-^i is relevant. Assuming that hyperplane $(\boldsymbol{w}_i, b_i)$ can perfectly classify examples in $\mathcal{D}^i$, the margin of $(\boldsymbol{w}_i, b_i)$ can be defined as $2/\|\boldsymbol{w}_i\|$ by appropriately normalizing $(\boldsymbol{w}_i, b_i)$ (Cortes and Vapnik 1995), where $\|\cdot\|$ denotes the vector norm. We can get the maximum margin hyperplane by maximizing $2/\|\boldsymbol{w}_i\|$ which is equivalent to minimizing $\|\boldsymbol{w}_i\|^2/2$. Considering all pairs of class labels, let $\mathbf{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m] \in \mathbb{R}^{d \times m}$ and $\boldsymbol{b} = (b_1, \ldots, b_m)^\top$, and for a more general case that training examples in each $\mathcal{D}^i$ can't be separated perfectly, slack variables $\boldsymbol{\xi} = (\xi_1^1, \ldots, \xi_{n_1}^1, \ldots, \xi_1^m, \ldots, \xi_{n_m}^m)^\top$ can be introduced to model the empirical risk. Then, we can get the following maximum margin formulation for MDC:

$$\begin{aligned} \min_{\mathbf{W}, \boldsymbol{b}, \boldsymbol{\xi}} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n_i} \xi_j^i + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^\top) \\ \text{s.t.} \quad & y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) \geq 1 - \xi_j^i, \\ & \xi_j^i \geq 0, \ i = 1, \ldots, m, \ j = 1, \ldots, n_i \end{aligned} \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes inner product of two vectors, $\mathrm{tr}(\cdot)$ denotes the trace of a square matrix, and λ_1 is a regularization parameter. The formulation in Eq.(1) just independently deals with each pair of class labels, i.e., dependencies among class spaces are ignored. Following the idea in (Zhang and Yeung 2014; Liu et al. 2016; Ma and Chen 2019), we model the relationships among all $\boldsymbol{w}_i$s in $\mathbf{W}$ with the column covariance matrix of $\mathbf{W}$. Thereafter, the above optimization problem turns out to be:

$$\begin{aligned} \min_{\mathbf{W}, \boldsymbol{b}, \boldsymbol{\xi}, \mathbf{C}} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n_i} \xi_j^i + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^\top) \\ & + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top) \\ \text{s.t.} \quad & \mathbf{C} \succeq 0, \mathrm{tr}(\mathbf{C}) \leq 1, \\ & \quad y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) > 1 - \xi_j^i, \\ & \quad \xi_j^i \geq 0, \ i = 1, \ldots, m, \ j = 1, \ldots, n_i \end{aligned} \tag{2}$$

Here $\mathbf{C} \succeq 0$ means that $\mathbf{C}$ is positive semi-definite which corresponds to a covariance matrix, and $\mathrm{tr}(\mathbf{C}) \leq 1$ is used to penalize its complexity. λ_2 is another regularization parameter.

Obviously, the first two terms in objective function are convex with respect to $\mathbf{W}$ and $\boldsymbol{b}$, and it has been proved in (Zhang and Yeung 2014) that the third term in the objective function is also a convex function with respect to $\mathbf{W}$, $\boldsymbol{b}$ and $\mathbf{C}$. So the optimization problem in Eq.(2) is jointly convex.

However, it is not easy to solve this optimization problem directly because of the non-linear and non-smooth constraint $\mathbf{C} \succeq 0$. Here, we use an alternating method to solve it efficiently. Specifically, the objective function with respect to $\mathbf{W}$ and $\boldsymbol{b}$ is firstly optimized when $\mathbf{C}$ is fixed, and then it is optimized with respect to $\mathbf{C}$ when $\mathbf{W}$ and $\boldsymbol{b}$ are fixed. These two steps are repeated until convergence. Technical details of the two alternating steps are introduced as follows.

Optimizing with respect to W and *b* when C is fixed. When $\mathbf{C}$ is fixed, we can reformulate the optimization problem in Eq.(2) as follows:

$$\begin{aligned} \min_{\mathbf{W}, \boldsymbol{b}, \boldsymbol{\xi}} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n_i} \xi_j^i + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^\top) \\ & + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top) \\ \text{s.t.} \quad & y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) > 1 - \xi_j^i, \\ & \quad \xi_j^i \geq 0, \ i = 1, \ldots, m, \ j = 1, \ldots, n_i \end{aligned} \tag{3}$$

The Lagrangian of the above problem is given by:

$$\begin{aligned} \mathcal{L}(\mathbf{W}, \boldsymbol{b}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = & \sum_{i=1}^{m} \sum_{j=1}^{n_i} \xi_j^i + \frac{\lambda_1}{2} \mathrm{tr}(\mathbf{W}\mathbf{W}^\top) \\ & + \frac{\lambda_2}{2} \mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top) \\ & - \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i [y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) - 1 + \xi_j^i] \\ & - \sum_{i=1}^{m} \sum_{j=1}^{n_i} \beta_j^i \xi_j^i \end{aligned} \tag{4}$$

where $\alpha_j^i, \beta_j^i \geq 0$. Then, the gradients of $\mathcal{L}$ are calculated with respected to $\mathbf{W}, b_i$ and ξ_j^i, and we can obtain the following equations by setting them to be 0:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = 0 \Rightarrow$$

$$\mathbf{W} = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i y_j^i \boldsymbol{x}_j^i \boldsymbol{e}_i^\top \mathbf{C}(\lambda_1 \mathbf{I}_m + \lambda_2 \mathbf{C})^{-1} \tag{5}$$

$$\frac{\partial \mathcal{L}}{\partial b_i} = 0 \Rightarrow \sum_{j=1}^{n_i} \alpha_j^i y_j^i = 0,\ (1 \leq i \leq m) \tag{6}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_j^i} = 0 \Rightarrow \alpha_j^i + \beta_j^i = 1 \tag{7}$$

where $\boldsymbol{e}_i$ is the ith column vector of identity matrix $\mathbf{I}_m$. Plugging Eq.(5)$\sim$Eq.(7) into Eq.(4), the dual problem, i.e., $\max_{\boldsymbol{\alpha}}\ \min_{\boldsymbol{W},\boldsymbol{b}}\ \mathcal{L}(\mathbf{W}, \mathbf{b})$, can be equivalently formulated as:

$$\begin{aligned} \min_{\boldsymbol{\alpha}}\ & \frac{1}{2} \sum_{i_1=1}^{m} \sum_{j_1=1}^{n_{i_1}} \sum_{i_2=1}^{m} \sum_{j_2=1}^{n_{i_2}} \alpha_{j_1}^{i_1} \alpha_{j_2}^{i_2} y_{j_1}^{i_1} y_{j_2}^{i_2} M_{i_1 i_2} \langle \boldsymbol{x}_{j_1}^{i_1}, \boldsymbol{x}_{j_2}^{i_2} \rangle \\ & - \sum_{i=1}^{m} \sum_{j=1}^{n_i} \alpha_j^i \\ & \text{s.t.}\ \sum_{j=1}^{n_i} \alpha_j^i y_j^i = 0\ (1 \leq i \leq m),\ 0 \leq \alpha_j^i \leq 1 \end{aligned} \tag{8}$$

where $\mathbf{M} = (\lambda_1 \mathbf{I}_m + \lambda_2 \mathbf{C})^{-\top} \mathbf{C}^\top$, and $M_{i_1 i_2}$ denotes the element in i_1th row and i_2th column of $\mathbf{M}$. $\boldsymbol{\alpha} = (\alpha_1^1, \ldots, \alpha_{n_1}^1, \ldots, \alpha_1^m, \ldots, \alpha_{n_m}^m)^\top \in \mathbb{R}^{\sum_{j=1}^m n_j \times 1}$. Eq.(8) is a standard quadratic programming (QP) problem with m equality constraints, but the number of α_j^is, i.e., $\sum_{j=1}^m n_j$, is usually too large making this QP problem difficult to be solved directly. Here, we decompose it into m sub-QP problems with only one equality constraint as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha}^i}\ & \frac{1}{2} \sum_{j_1=1}^{n_i} \sum_{j_2=1}^{n_i} \alpha_{j_1}^{i} \alpha_{j_2}^{i} y_{j_1}^{i} y_{j_2}^{i} M_{ii} \langle \boldsymbol{x}_{j_1}^{i}, \boldsymbol{x}_{j_2}^{i} \rangle \\ & - \sum_{j=1}^{n_i} (1 - S_j^i) \alpha_j^i \\ & \text{s.t.}\ \sum_{j=1}^{n_i} \alpha_j^i y_j^i = 0,\ 0 \leq \alpha_j^i \leq 1 \end{aligned} \tag{9}$$

where $1 \leq i \leq m$, $S_j^i = y_j^i \sum_{i_1 \neq i} \frac{1}{2}(M_{ii_1} + M_{i_1 i}) \sum_{j_1=1}^{n_{i_1}} \alpha_{j_1}^{i_1} y_{j_1}^{i_1} \langle \boldsymbol{x}_j^i, \boldsymbol{x}_{j_1}^{i_1} \rangle$, and $\boldsymbol{\alpha}^i = (\alpha_1^i, \ldots, \alpha_{n_i}^i)^\top \in \mathbb{R}^{n_i \times 1}$. To solve the problem in Eq.(8), we can initialize $\boldsymbol{\alpha} = \mathbf{0}$, and then repeatedly solve the m sub-QP problems in Eq. (9) until all α_j^is meet Karush-Kuhn-Tucker (KKT) conditions.

Here, the values of $\mathbf{W}$ and $\boldsymbol{b}$ need to be obtained for validating KKT conditions. For $\mathbf{W}$, it just needs to plug $\boldsymbol{\alpha}$ into Eq.(5). But for $\boldsymbol{b}$, the situation is somewhat complicated. When there are α_j^is in $(0, 1)$, we have $y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) = 1$, it's easy to know that $b_i = y_j^i - \langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle$. However, when there aren't α_j^is in $(0, 1)$, i.e., either $\alpha_j^i = 0$ or $\alpha_j^i = 1$, we need to solve the inequalities. In this case, when $\alpha_j^i = 0$, b_i should meet $y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) \geq 1$, while when $\alpha_j^i = 1$, b_i should meet $y_j^i(\langle \boldsymbol{w}_i, \boldsymbol{x}_j^i \rangle + b_i) \leq 1$. Then we can get many upper and lower limits of b_i , in which we select the moderate one for b_i.

Optimizing with respect to $\mathbf{C}$ when $\mathbf{W}$ and $\boldsymbol{b}$ are fixed. When $\mathbf{W}$ and $\boldsymbol{b}$ are fixed, the optimization problem in Eq.(2) for finding $\mathbf{C}$ becomes:

$$\min_{\mathbf{C}}\ \mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top),\ \text{s.t.}\ \mathbf{C} \succeq 0, \mathrm{tr}(\mathbf{C}) \leq 1 \tag{10}$$

As per the property of $\mathrm{tr}(\mathbf{XYZ}) = \mathrm{tr}(\mathbf{YZX})$ and the constraint $\mathrm{tr}(\mathbf{C}) \leq 1$, we can lower-bound the objective in Eq.(10) as:

$$\begin{aligned} \mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top) &= \mathrm{tr}(\mathbf{C}^{-1}\mathbf{W}^\top\mathbf{W}) \\ &\geq \mathrm{tr}(\mathbf{C}^{-1}\mathbf{W}^\top\mathbf{W})\mathrm{tr}(\mathbf{C}) \\ &= \mathrm{tr}(\mathbf{C}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{C}^{-\frac{1}{2}})\mathrm{tr}(\mathbf{C}^{\frac{1}{2}}\mathbf{C}^{\frac{1}{2}}) \\ &\geq (\mathrm{tr}(\mathbf{C}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{C}^{\frac{1}{2}}))^2 = (\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2 \end{aligned} \tag{11}$$

where $\mathbf{A} = \mathbf{W}^\top\mathbf{W} = \sum_{i_1=1}^m \sum_{j_1=1}^{n_{i_1}} \sum_{i_2=1}^m \sum_{j_2=1}^{n_{i_2}} \alpha_{j_1}^{i_1} \alpha_{j_2}^{i_2} y_{j_1}^{i_1} y_{j_2}^{i_2} \mathbf{M}\boldsymbol{e}_{i_1}\boldsymbol{e}_{i_2}^\top\mathbf{M}^\top \langle \boldsymbol{x}_{j_1}^{i_1}, \boldsymbol{x}_{j_2}^{i_2} \rangle$. The last inequality in Eq.(11) holds based on the property that both $\mathbf{A}$ and $\mathbf{C}$ are symmetric as well as the following Lemma:

Lemma 1. *Given* $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{\ell_1 \times \ell_2}$, *then* $\mathrm{tr}(\mathbf{U}^\top\mathbf{U})\mathrm{tr}(\mathbf{V}^\top\mathbf{V}) \geq \big(\mathrm{tr}(\mathbf{U}^\top\mathbf{V})\big)^2$ *holds. The minimum can be got when* $\mathbf{U} = \mu \cdot \mathbf{V}$ *where* μ *is a constant.*

Proof. It is easy to know,

$$\mathrm{tr}(\mathbf{U}^\top\mathbf{U}) = \sum_{i=1}^{\ell_2} \sum_{j=1}^{\ell_1} U_{ij}^2 = \langle vec\mathbf{U}, vec\mathbf{U} \rangle = \|vec\mathbf{U}\|^2$$

$$\mathrm{tr}(\mathbf{V}^\top\mathbf{V}) = \sum_{i=1}^{\ell_2} \sum_{j=1}^{\ell_1} V_{ij}^2 = \langle vec\mathbf{V}, vec\mathbf{V} \rangle = \|vec\mathbf{V}\|^2$$

$$\mathrm{tr}(\mathbf{U}^\top\mathbf{V}) = \sum_{i=1}^{\ell_2} \sum_{j=1}^{\ell_1} U_{ij} V_{ij} = \langle vec\mathbf{U}, vec\mathbf{V} \rangle$$

Here, $vec\mathbf{U}, vec\mathbf{V}$ denote the results of vectorization for $\mathbf{U}, \mathbf{V}$. As per the property of inner product $\|vec\mathbf{U}\| \cdot \|vec\mathbf{V}\| \geq |\langle vec\mathbf{U}, vec\mathbf{V} \rangle|$, and take the square over both sides of this inequality, then we have $\|vec\mathbf{U}\|^2 \cdot \|vec\mathbf{V}\|^2 \geq (\langle vec\mathbf{U}, vec\mathbf{V} \rangle)^2$. The equality relationship holds only when $vec\mathbf{U} = \mu \cdot vec\mathbf{V}$, i.e., $\mathbf{U} = \mu \cdot \mathbf{V}$. □

According to Eq.(11), $\mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top)$ attains its minimum value $(\mathrm{tr}(\mathbf{A}^{\frac{1}{2}}))^2$ when $\mathrm{tr}(\mathbf{C}) = 1$ and $\mathbf{A}^{\frac{1}{2}}\mathbf{C}^{-\frac{1}{2}} = \mu\mathbf{C}^{\frac{1}{2}}$ for some constant μ. Therefore, the closed-form solution of $\mathbf{C}$ can be obtained (Zhang and Yeung 2014) as follows:

$$\mathbf{C} = \frac{(\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}}}{\mathrm{tr}((\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}})} \tag{12}$$

As the above two alternating optimizing steps converge, we can get the optimal values of $\mathbf{W}$, $\boldsymbol{b}$ and $\mathbf{C}$. Then, predictions

Table 1: The pseudo-code of M^{3}MDC.

```
Inputs:
$\mathcal{D}$:               MDC training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \le i \le N\}$
$\lambda_1, \lambda_2$:      regularization parameters
$\boldsymbol{x}_*$:          unseen instance
Outputs:
$\boldsymbol{y}_*$:          predicted class vector for $\boldsymbol{x}_*$
Process:
 1: Transform $\mathcal{D}$ to a total of $m = \sum_{j=1}^{q}(K_j(K_j-1))/2$
    binary classification data sets via OvO decomposition
    w.r.t. each class space;
 2: Initialize $\mathbf{C} = \frac{1}{m}\mathbf{I}_m$ and $\boldsymbol{\alpha} = \mathbf{0}$;
 3: repeat
 4:    while not all $\boldsymbol{\alpha}$ meet KKT conditions do
 5:       for $i = 1$ to $m$ do
 6:          Solve sub-QP problem in Eq.(9);
 7:       end for
 8:    end while
 9:    Calculate $\mathbf{C}$ according to Eq.(12);
10: until convergence
11: Calculate $m$ binary predictions $\boldsymbol{y}_*^b$ for $\boldsymbol{x}_*$ according to
    Eq.(13);
12: Return $\boldsymbol{y}_*$ via OvO decoding rule based on $\boldsymbol{y}_*^b$.
```

for unseen instances can be made accordingly. Specifically, for test instance $\boldsymbol{x}_*$, we can get its binary prediction vector $\boldsymbol{y}_*^b$ with a total of m elements as follows:

$$\begin{aligned} \boldsymbol{y}_*^b &= \mathrm{sign}(\mathbf{W}^\top \boldsymbol{x}_* + \boldsymbol{b}) \qquad (13) \\ &= \mathrm{sign}(\sum_{i=1}^{m}\sum_{j=1}^{n_i} \alpha_j^i y_j^i \mathbf{M}\boldsymbol{e}_i \langle \boldsymbol{x}_j^i, \boldsymbol{x}_* \rangle + \boldsymbol{b}) \end{aligned}$$

where $\mathrm{sign}(\cdot)$ is the (element-wise) signed function. The first $\frac{K_1(K_1-1)}{2}$ elements in $\boldsymbol{y}_*^b$ belong to the first class space, the $\frac{K_1(K_1-1)}{2} + 1 \sim \frac{K_2(K_2-1)}{2}$ elements belong to the second class space, and so on. Finally, we can make prediction of each class space for $\boldsymbol{x}_*$ via OvO decoding rule based on these binary predictions.

In summary, Table 1 presents the complete procedure of the proposed M^{3}MDC approach. Firstly, we employ OvO decomposition for the original MDC problem per class space (Step 1). After that, an alternating optimizing process is used to solve the problem in Eq.(2) (Steps 2-10). Finally, the class vector for unseen instance is predicted based on its m binary predictions (Steps 11-12).

Computational complexity. Let $\mathcal{F}_{QP}(r)$ denote the time complexity to solve Eq.(9) with r variables, and $\mathcal{F}_{SR}(s)$ denote the time complexity to solve matrix square root operation in Eq.(12) with $s \times s$ elements.[1] M^{3}MDC has computational complexity $\mathcal{O}(T_1 \cdot T_2 \cdot m \cdot \mathcal{F}_{QP}(N) + T_1 \cdot \mathcal{F}_{SR}(m))$ for training phase, where T_1 denotes the number of iterations of the whole alternating optimizing process and T_2 denotes the number of iterations of m sub-QP problems in Eq.(9). Moreover, $\mathcal{F}_{QP}(N)$ is actually the maximum complexity of each Eq.(9) because the number of examples in each OvO decomposition is always less than N.

Related Work

Intuitively, MDC corresponds to a set of traditional multi-class classification (MCC), one per class space. However, it is better to solve the set of MCC together rather than one by one independently, because dependencies among class variables usually exist due to the fact that all these MCC problems share the same input space. Therefore, most existing MDC approaches try to model class dependencies in different ways, such as capturing pairwise interactions between class variables (Arias et al. 2016), specifying chaining order over class variables (Zaragoza et al. 2011; Read, Martino, and Luengo 2014), assuming directed acyclic graph (DAG) structure over class variables (Bielza, Li, and Larrañaga 2011; Batal, Hong, and Hauskrecht 2013; Zhu, Liu, and Jiang 2016; Bolt and van der Gaag 2017; Gil-Begue, Larrañaga, and Bielza 2018; Benjumeda, Bielza, and Larrañaga 2018), and partitioning class variables into groups (Read, Bielza, and Larrañaga 2014), etc.

Furthermore, MDC can also be regarded as a generalized version of multi-label classification (MLC) (Zhang and Zhou 2014; Gibaja and Ventura 2015) by not restricting binary-valued class variable in each class space. However, the key difference between MDC and MLC is whether the class space is *heterogenous* or *homogeneous*. Generally, MDC assumes *heterogenous* class spaces which characterize objects' semantics along different dimensions, while MLC assumes *homogeneous* class space which characterizes the relevancy of specific concepts along one dimension. In other words, the relationship between a pair of class labels from different class spaces in MDC is different from the relationship between a pair of class labels in MLC. Therefore, it is unreasonable and will get suboptimal solutions to directly align class labels from different class spaces when trying to design MDC approaches.

Maximum margin techniques have been widely used to solve MCC and MLC problems. For MCC with single-label assignment, one can derive margin-based classification models by transforming the MCC problem into a number of binary classification problems via one-vs-one, one-vs-all, and many-vs-many decomposition, or directly maximizing multi-class margins. For MLC with multi-label assignment, one can also derive margin-based classification models via binary decomposition, or by maximizing margins between relevant-irrelevant label pairs (Elisseeff and Weston 2002), or relevant-relevant label pairs with different importance degrees (Xu, Li, and Zhou 2019), or output coding margins (Liu and Tsang 2015; Liu et al. 2019), etc.

It is worth noting that we adopt the same strategy in (Zhang and Yeung 2014; Liu et al. 2016; Ma and Chen 2019) by employing the regularization term $\mathrm{tr}(\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top)$ to help induce a set of learners jointly. Nonetheless, M^{3}MDC differs from those related works which aims to solve the

[1] `MOSEK` optimization software (https://www.mosek.com/) is used to solve Eq.(9), and built-in function `sqrtm` in Matlab is used to solve Eq.(12).

MDC problem based on maximum margin formulation. Furthermore, the empirical loss utilized by M^{3}MDC follows from the one-vs-one decomposition w.r.t. each class space.

Experiments

Experimental Setup

Benchmark data sets. A total of ten benchmark data sets are employed for performance evaluation. Table 2 summarizes the characteristics of all MDC data sets, including *number of examples* (#Exam.), *number of class spaces* (#Dim.), *number of class labels per class space* (#Labels/Dim.),[2] and *number of features* (#Features).

Comparing approaches. The performance of M^{3}MDC is compared with four well-established MDC approaches (Read, Bielza, and Larrañaga 2014) including Binary Relevance (BR), Ensembles of Classifier Chains (ECC), Ensembles of Class Powerset (ECP), and Ensembles of Super Class classifiers (ESC). BR solves MDC problem by training a number of independent multi-class classifiers, one per class space, while ECC, ECP, ESC model dependencies among class spaces by specifying a chaining order over class spaces, conducting powerset transformation, and grouping the MDC class variables into super-classes respectively.

For ensemble approaches ECC, ECP and ESC, a random cut of 67% examples from the original MDC training set is used to generate the base MDC model and the number of base classifiers is set to be 10. Furthermore, predictions of base MDC models are combined via majority voting. Support vector machine (SVM) is used to instantiate BR, ECC, ECP, ESC as base classifier. Specifically, LIBSVM (Chang and Lin 2011) with linear kernel is used.[3] As shown in Table 1, the two regularization parameters for M^{3}MDC are set to be $\lambda_1 = 0.1, \lambda_2 = 0.001$ respectively.

Evaluation metrics. In this paper, a total of three metrics, i.e., *Hamming Score*, *Exact Match* and *Sub-Exact Match*, are utilized to measure the generalization abilities of MDC approaches. Specifically, let $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq p\}$ denote the test set, where $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top$ is the ground-truth class vector associated with $\boldsymbol{x}_i$. To evaluate the performance of the MDC predictive function f, let $\hat{\boldsymbol{y}}_i = f(\boldsymbol{x}_i) = [\hat{y}_{i1}, \hat{y}_{i2}, \ldots, \hat{y}_{iq}]^\top$ denote the predicted class vector of $\boldsymbol{x}_i$, and then we can get the number of class spaces which f predicts correctly, i.e., $r^{(i)} = \sum_{j=1}^{q} [\![y_{ij} = \hat{y}_{ij}]\!]$. Here, the predicate $[\![\pi]\!]$ returns 1 if π holds and 0 otherwise. Concrete metric definitions can be given as follows:

- *Hamming Score*:

$$\mathrm{HScore}_{\mathcal{S}}(f) = \frac{1}{p} \sum_{i=1}^{p} \frac{1}{q} \cdot r^{(i)}$$

Table 2: Characteristics of the experimental data sets.

Data Set	#Exam.	#Dim.	#Labels/Dim.	#Features[†]
Edm	154	2	3	$16n$
Flare1	323	3	3,4,2	$10x$
Cal500	502	10	2	$68n$
Music	591	6	2	$71n$
Song	785	3	3	$98n$
WQplants	1060	7	4	$16n$
WQanimals	1060	7	4	$16n$
WaterQuality	1060	14	4	$16n$
Yeast	2417	14	2	$103n$
Voice	3136	2	4,2	$19n$

[†] n and x denote numeric and nominal features respectively.

- *Exact Match*:

$$\mathrm{EMatch}_{\mathcal{S}}(f) = \frac{1}{p} \sum_{i=1}^{p} [\![r^{(i)} = q]\!]$$

- *Sub-Exact Match*:

$$\mathrm{SEMatch}_{\mathcal{S}}(f) = \frac{1}{p} \sum_{i=1}^{p} [\![r^{(i)} \geq q - 1]\!]$$

In a nutshell, *Hamming Score* is the average accuracy over all class spaces, while *Exact Match* is the accuracy when considering all class spaces as a single one by conducting powerset transformation. *Sub-Exact Match* is a relaxed version of *Exact Match* where at most one incorrect prediction can be made over all class spaces for each test example. For all three metrics, the *larger* the values the better the performance. Ten-fold cross-validation is performed on the benchmark data sets, where the mean metric value as well as standard deviation are recorded for each comparing approach.

Experimental Results

Table 3 reports the detailed experimental results of five comparing approaches in terms of each evaluation metric, where the best performance among all comparing approaches is shown in boldface. Moreover, *Wilcoxon signed-ranks test* (Demšar 2006) is used as the statistical test to show whether M^{3}MDC performs significantly better than BR, ECC, ECP, ESC respectively. Table 4 summarizes the statistical test results and the p-values for the corresponding tests are also shown in the brackets. Here, the significance level is set to be 0.05.

Based on the reported experimental results, the following observations can be made:

- Across all the 30 cases (10 data sets × 3 evaluation metrics), M^{3}MDC ranks first in 21 cases, ranks second in 3 cases, and never ranks last.
- In terms of *Hamming Score*, M^{3}MDC is statistically better than BR, ECC, ECP, ESC.
- ECP can be regarded as an approach which works by maximizing *Exact Match* via class powerset transformation. It is worth noting that M^{3}MDC still ranks first in 5 out of 10 cases in term of this metric and can achieve comparable performance against ECP.

[2] If all class spaces have the same number of class labels, then only this number is recorded; Otherwise, the number of class labels in each class space is recorded in turn.

[3] Due to the margin-based nature of M^{3}MDC, we employ LIBSVM as the base classifier for fair comparison between M^{3}MDC and the comparing approaches.

Table 3: Predictive performance of each comparing approach (mean±std. deviation) on experimental data sets. Moreover, the best performance among all comparing approaches is shown in boldface.

(a) Hamming Score

Data Set	Edm	Flare1	Cal500	Music	Song	WQpla.	WQani.	WQ	Yeast	Voice
M^{3}MDC	**.728±.083**	**.923±.033**	**.630±.010**	.811±.022	**.795±.029**	**.660±.013**	**.632±.014**	**.647±.012**	**.802±.006**	**.971±.009**
BR	.689±.070	.922±.034	.628±.011	.808±.023	.793±.023	.657±.016	.630±.014	.644±.013	.801±.006	.964±.007
ECC	.695±.065	.922±.034	.625±.015	**.814±.025**	.790±.024	.654±.016	.630±.014	.643±.013	.797±.007	.961±.008
ECP	.721±.082	.921±.036	.616±.015	.799±.032	.786±.029	.647±.015	.629±.013	.628±.015	.795±.007	.955±.013
ESC	.701±.079	**.923±.033**	.616±.019	.809±.022	.790±.030	.651±.016	.630±.014	.641±.013	.800±.006	.961±.008

(b) Exact Match

Data Set	Edm	Flare1	Cal500	Music	Song	WQpla.	WQani.	WQ	Yeast	Voice
M^{3}MDC	.501±.139	**.821±.073**	.016±.016	.281±.074	**.488±.065**	**.102±.035**	.059±.022	**.008±.008**	.157±.018	**.942±.017**
BR	.442±.125	**.821±.073**	.016±.016	.272±.075	.479±.059	.097±.033	.058±.022	.007±.008	.151±.017	.929±.014
ECC	.454±.123	.817±.078	.020±.016	**.346±.079**	.481±.057	.093±.037	.061±.023	.006±.008	.207±.014	.923±.016
ECP	**.559±.136**	.817±.078	**.026±.028**	.343±.076	.484±.054	.093±.028	**.065±.018**	.001±.003	**.252±.012**	.912±.025
ESC	.513±.122	**.821±.073**	.014±.013	.330±.069	.480±.067	.094±.038	.062±.021	.006±.008	.236±.019	.924±.016

(c) Sub-Exact Match

Data Set	Edm	Flare1	Cal500	Music	Song	WQpla.	WQani.	WQ	Yeast	Voice
M^{3}MDC	**.955±.053**	**.951±.036**	.082±.046	**.687±.067**	.901±.042	**.287±.051**	**.237±.028**	**.051±.025**	.273±.028	**.999±.001**
BR	.935±.061	.947±.039	.074±.037	.674±.067	**.903±.033**	**.287±.055**	.229±.034	**.051±.024**	.269±.029	**.999±.002**
ECC	.935±.069	**.951±.036**	.080±.031	.676±.064	.891±.036	.283±.049	.229±.032	.050±.023	.288±.023	.998±.002
ECP	.883±.074	.947±.039	.078±.036	.640±.064	.878±.040	.281±.049	.230±.032	.035±.018	.304±.020	.998±.003
ESC	.890±.076	**.951±.036**	**.086±.038**	.669±.062	.893±.038	.284±.050	.232±.033	.046±.022	**.309±.028**	.998±.002

Table 4: Wilcoxon signed-ranks test for M^{3}MDC against BR,ECC,ECP,ESC in terms of each evaluation metric (significance level $\alpha = 0.05$; p-values shown in the brackets).

Evaluation Metric	M^{3}MDC **vs** BR	M^{3}MDC **vs** ECC	M^{3}MDC **vs** ECP	M^{3}MDC **vs** ESC
Hamming Score	**win** [1.95e-3]	**win** [9.77e-3]	**win** [1.95e-3]	**win** [3.91e-3]
Exact Match	**win** [7.81e-3]	**tie** [7.70e-1]	**tie** [4.32e-1]	**tie** [7.54e-1]
Sub-Exact Match	**win** [2.34e-2]	**tie** [9.77e-2]	**win** [4.88e-2]	**tie** [1.95e-1]

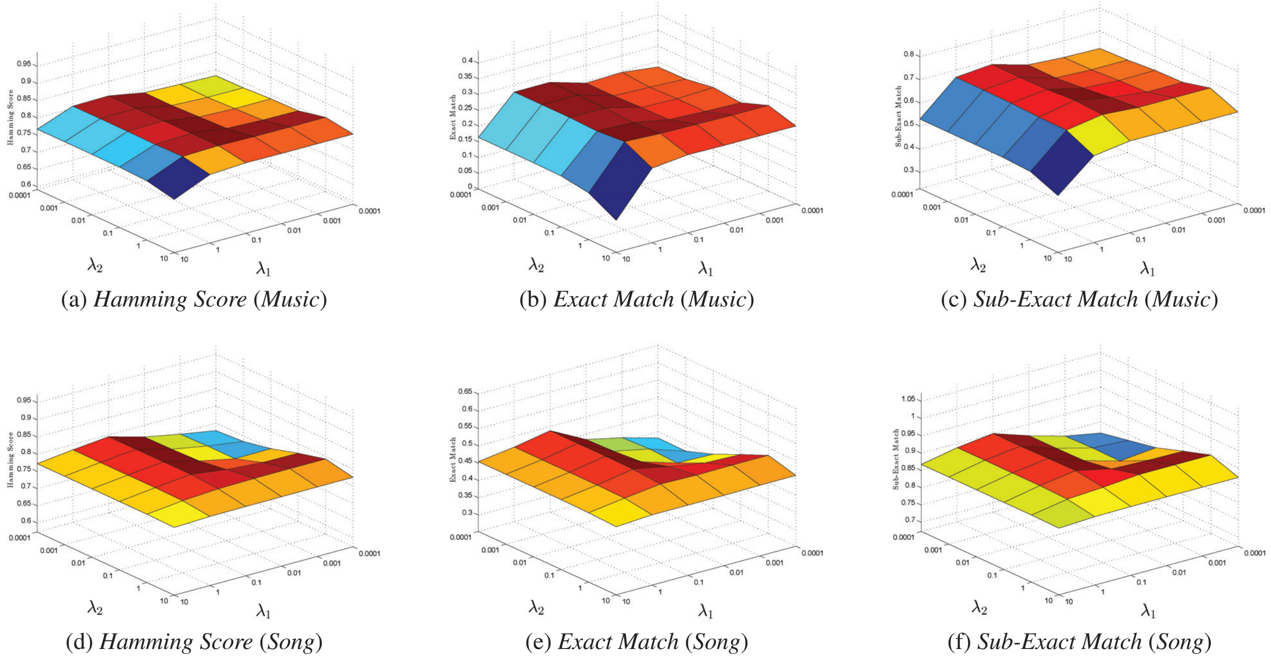
- It is impressive to notice that M^{3}MDC is statistically better than BR in terms of all evaluation metrics, which clearly validates the effectiveness of M^{3}MDC in modeling relationships among class spaces.

Further Analysis

Sensitivity analysis. As shown in Eq.(2), λ_1, λ_2 are used to make a tradeoff among empirical risk, structural risk and relationship regularizer. Figure 1 shows how the performance of M^{3}MDC changes w.r.t. λ_1, λ_2 on data sets *Music* and *Song* respectively. Similar results can be obtained on other data sets which are not reported here due to page limit. In terms of each evaluation metric, M^{3}MDC can achieve relatively better performance when $\lambda_1 = 0.1$ and $\lambda_2 \leq 1$. In this paper, we fix $\lambda_1 = 0.1, \lambda_2 = 0.001$ respectively, which are also the recommended default parameter settings for ease of use.

Correlation analysis. By normalizing matrix $\mathbf{C}$ in Eq.(2) with its diagonal elements, we can get correlation matrix $\mathbf{R}$ which represents the relationships among all pairs of class labels, i.e., $R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} \times C_{jj}}}$, where R_{ij} (C_{ij}) denotes element in ith row and jth column of $\mathbf{R}$ ($\mathbf{C}$). We depict the correlation matrix $\mathbf{R}$ on data sets *Song*, *WaterQuality* and *Yeast* in Figure 2. Here, $+1$ indicates absolutely positive correlation (i.e., red color) while -1 indicates absolutely negative correlation (i.e., blue color). As shown in Figure 2, there are indeed some red or blue squares (excluding diagonal ones), which indicate that dependencies among classes do exist. However, there are many squares in green which indicate independencies between classes. These observations show that class dependencies should indeed be taken into account but with great care when designing MDC approaches. M^{3}MDC can model class dependencies automatically as long as dependencies exist which is a desirable property when inducing predictive models.

Convergence analysis. The optimization problem in Eq.(3) is solved in an alternating way. Although the objective function is jointly convex, here we also analyze its convergent characteristics. Specifically, Frobenius norm of the difference between each pair of $\mathbf{W}$s in two adjacent iterations is recorded. Figure 3 illustrates how the Frobenius norm changes as the number of iterations increases on data sets *Song* and *Voice*. It is observed that the difference has



(a) *Hamming Score* (*Music*)

(b) *Exact Match* (*Music*)

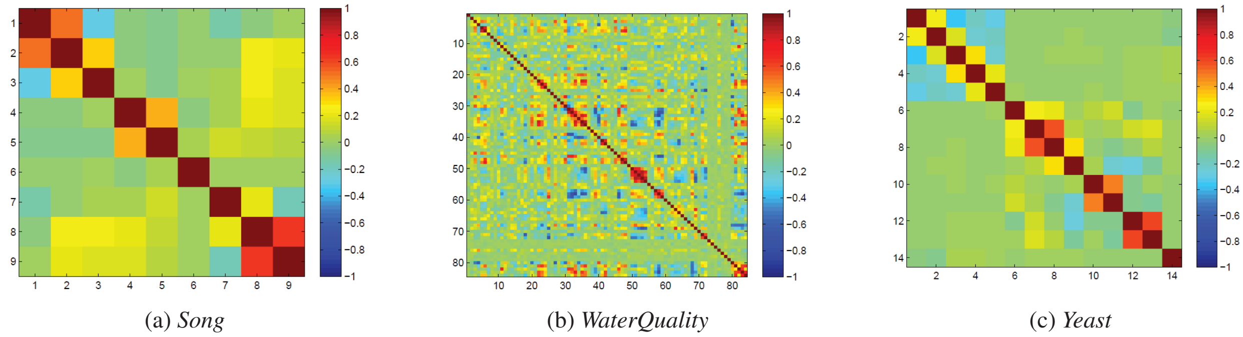
(c) *Sub-Exact Match* (*Music*)

(d) *Hamming Score* (*Song*)

(e) *Exact Match* (*Song*)

(f) *Sub-Exact Match* (*Song*)

Figure 1: Performance of M^{3}MDC changes as λ_1, λ_2 range in $\{10, 1, 0.1, 0.01, 0.001, 0.0001\}$

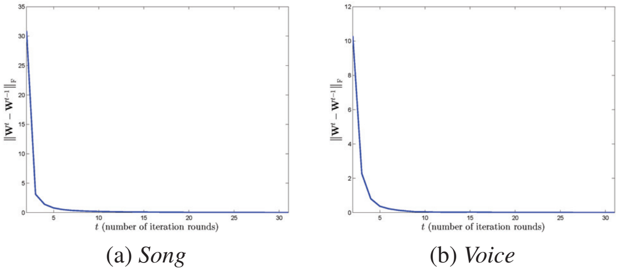


(a) *Song*

(b) *WaterQuality*

(c) *Yeast*

Figure 2: Correlation matrix on data sets *Song*, *WaterQuality*, and *Yeast*



(a) *Song*

(b) *Voice*

Figure 3: Convergence curves on data sets *Song* and *Voice*.

been very small when the number of iterations reaches 5, which means M^{3}MDC usually converges very quickly.

Conclusion

In this paper, the problem of margin-based multi-dimensional classification is investigated. Specifically, a novel approach named M^{3}MDC is proposed which considers the margin over MDC examples via OvO decomposition and models the dependencies among class spaces with covariance regularization. The resulting convex formulation is solved via alternating optimization admitting QP or closed-form solution in either alternating step. Experimental studies on benchmark data sets clearly validate the effectiveness of the derived M^{3}MDC approach.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Key R&D Program of China (2018YFB1004300), the National Science Foundation of

China (61573104), and partially supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Arias, J.; Gamez, J. A.; Nielsen, T. D.; and Puerta, J. M. 2016. A scalable pairwise class interaction framework for multidimensional classification. *International Journal of Approximate Reasoning* 68:194–210.

Batal, I.; Hong, C.; and Hauskrecht, M. 2013. An efficient probabilistic framework for multi-dimensional classification. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, 2417–2422.

Benjumeda, M.; Bielza, C.; and Larrañaga, P. 2018. Tractability of most probable explanations in multidimensional bayesian network classifiers. *International Journal of Approximate Reasoning* 93:74–87.

Bielza, C.; Li, G.; and Larrañaga, P. 2011. Multi-dimensional classification with bayesian networks. *International Journal of Approximate Reasoning* 52(6):705–727.

Bolt, J. H., and van der Gaag, L. C. 2017. Balanced sensitivity functions for tuning multi-dimensional bayesian network classifiers. *International Journal of Approximate Reasoning* 80:361–376.

Borchani, H.; Bielza, C.; Toro, C.; and Larrañaga, P. 2013. Predicting human immunodeficiency virus inhibitors using multi-dimensional bayesian network classifiers. *Artificial Intelligence in Medicine* 57(3):219–229.

Chang, C.-C., and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(3):Article 27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

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

Demšar, J. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7(Jan):1–30.

Elisseeff, A., and Weston, J. 2002. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems*, 681–687.

Gibaja, E., and Ventura, S. 2015. A tutorial on multilabel learning. *ACM Computing Surveys* 47(3):Article 52.

Gil-Begue, S.; Larrañaga, P.; and Bielza, C. 2018. Multi-dimensional bayesian network classifier trees. In *Lecture Notes in Computer Science 11314*. Switzerland: Springer. 354–363.

Jia, B.-B., and Zhang, M.-L. 2019. Multi-dimensional classification via kNN feature augmentation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 3975–3982.

Liu, W., and Tsang, I. W. 2015. Large margin metric learning for multi-label prediction. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2800–2806.

Liu, M.; Zhang, D.; Chen, S.; and Xue, H. 2016. Joint binary classifier learning for ecoc-based multi-class classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38(11):2335–2341.

Liu, W.; Xu, D.; Tsang, I. W.; and Zhang, W. 2019. Metric learning for multi-output tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(2):408–422.

Ma, Z., and Chen, S. 2018. Multi-dimensional classification via a metric approach. *Neurocomputing* 275:1121–1131.

Ma, Z., and Chen, S. 2019. A convex formulation for multiple ordinal output classification. *Pattern Recognition* 86:73–84.

Read, J.; Bielza, C.; and Larrañaga, P. 2014. Multi-dimensional classification with super-classes. *IEEE Transactions on Knowledge and Data Engineering* 26(7):1720–1733.

Read, J.; Martino, L.; and Luengo, D. 2014. Efficient monte carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition* 47(3):1535–1546.

Rodríguez, J. D.; Pérez, A.; Arteta, D.; Tejedor, D.; and Lozano, J. A. 2012. Using multidimensional bayesian network classifiers to assist the treatment of multiple sclerosis. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews* 42(6):1705–1715.

Sagarna, R.; Mendiburu, A.; Inza, I.; and Lozano, J. A. 2014. Assisting in search heuristics selection through multidimensional supervised classification: A case study on software testing. *Information Sciences* 258:122–139.

Serafino, F.; Pio, G.; Ceci, M.; and Malerba, D. 2015. Hierarchical multidimensional classification of web documents with multiwebclass. In *Lecture Notes in Computer Science 9356*. Berlin: Springer. 236–250.

Xu, M.; Li, Y.-F.; and Zhou, Z.-H. 2019. Robust multi-label learning with PRO loss. *IEEE Transactions on Knowledge and Data Engineering* , in press.

Zaragoza, J. H.; Sucar, L. E.; Morales, E. F.; Bielza, C.; and Larrañaga, P. 2011. Bayesian chain classifiers for multidimensional classification. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, volume 11, 2192–2197.

Zhang, Y., and Yeung, D.-Y. 2014. A regularization approach to learning task relationships in multi-task learning. *ACM Transactions on Knowledge Discovery from Data* 8(3):Article 12.

Zhang, M.-L., and Zhou, Z.-H. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26(8):1819–1837.

Zhu, M.; Liu, S.; and Jiang, J. 2016. A hybrid method for learning multi-dimensional bayesian network classifiers based on an optimization model. *Applied Intelligence* 44(1):123–148.