Decomposition-based Classifier Chains for Multi-Dimensional Classification

Bin-Bin Jia, Min-Ling Zhang, *Senior Member, IEEE*

***Abstract*—In multi-dimensional classification, the semantics of objects are characterized by multiple class variables from different dimensions. To model the dependencies among class variables, one natural strategy is to build a number of multi-class classifiers in a chaining structure, one per dimension, where the subsequent classifiers on the chain augment the feature space with all labeling information used by the preceding classifiers. However, it is shown that this strategy cannot compete with existing state-of-the-art approaches via comparative studies. One possible reason is that inaccurate predictions of preceding classifiers would degenerate the performance of subsequent ones. Besides, it is more difficult to learn a multi-class classifier than a binary one with the same accuracy, and better performance can be expected if the multi-dimensional classification problem can be solved by building multiple binary classifiers in a chaining structure. Based on these conjectures, this paper proposes an approach which builds a chain of binary classifiers to solve the multi-dimensional classification problem with the help of one-vs-one decomposition. To address the issue that different one-vs-one decomposed problems involve different training examples, the feature space is augmented with the binary predictions of preceding classifiers on the chain to train the subsequent ones. To alleviate the effect of the specified chaining order, the ensemble version of proposed approach is further investigated. Comparative studies over twenty benchmark data sets clearly show the superiority of the proposed approach against the state-of-the-art multi-dimensional classification baselines.**

***Index Terms*—Machine Learning, Multi-Dimensional Classification, Classifier Chains, Class Dependencies.**

I. Introduction

In the field of machine learning, there are more and more application scenarios where only one single output variable cannot qualify for characterizing the rich semantics of objects. As a result, multi-output learning [28], which characterizes the semantics of objects by multiple output variables, has attracted increasing attentions in recent years. Multi-dimensional classification (MDC) [15] can be regarded as a special case of multi-output learning where all output variables correspond to multi-class type. Specifically, each MDC example is represented by a single instance while associated with multiple class variables. Here, each class variable corresponds to one specific class space which characterizes the semantics of objects from one dimension. Many applications in various areas (e.g., bioinformatics [4], text classification [24], resource allocation [1], ecology [27], etc.) can be naturally formalized under the MDC framework. Fig.1 further shows an intuition of multi-dimensional classification on vehicle classification.

Input space (vehicles)	Output space		
	Color	*Brand*	*Type*
	Red	Audi	Car
	Black	Benz	SUV
	Red	YUTONG	Bus
⋮	⋮	⋮	⋮
	White	JAC	Truck
	?	?	?

Fig. 1. An intuition on multi-dimensional classification. Here, the semantics of vehicles are characterized from three different dimensions, including the *color* dimension (with possible classes *Red*, *Black*, *White*, etc.), the *brand* dimension (with possible classes *Audi*, *Benz*, *YUTONG*, *JAC*, etc.), and the *type* dimension (with possible classes *Car*, *SUV*, *Bus*, *Truck*, etc.).

To solve the MDC problem, an intuitive strategy is to decompose the MDC problem into a number of independent multi-class classification problems, one per dimension. However, possible dependencies among class spaces are ignored by this independent decomposition strategy which would degenerate the generalization performance. To consider the class dependencies, another intuitive strategy is to treat all class variables as a compound one, where all distinct class combinations comprise the set of class labels for the compound class variable. However, class combinations not appearing in training set cannot be returned for unseen instance by this powerset-like strategy which would also degenerate the generalization performance. The two intuitive strategies reveal the key challenge of learning from MDC data that either ignoring or overfitting the class dependencies should be avoided when inducing MDC models. In other words, considering class dependencies is important but should be done in a well-designed way. Typical existing works in modeling class dependencies include grouping class variables into super-classes according to conditional dependence information [21], learning a distance metric in the decomposed label space [18], assuming a directed acyclic graph (DAG) structure over class variables [13], etc.

There is also another strategy to consider class dependencies which solves the MDC problem by training a chain of multi-

Bin-Bin Jia is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China and the College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China. (e-mail: jiabb@seu.edu.cn).

Min-Ling Zhang (corresponding author) is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. (e-mail: zhangml@seu.edu.cn).

class classifiers, one per dimension.[1] In fact, this strategy can be regarded as a directly generalized version of the classifier chains (CC) model [23] which is initially proposed in the multi-label context. Specifically, the vanilla CC for multi-label classification (MLC) [33] trains a chain of binary classifiers, one per label, where the subsequent classifiers on the chain will augment the feature space with the labels which are used to train the preceding classifiers. CC can not only model the dependencies among labels via the chaining structure, but also maintain acceptable computational complexity similar to the independent decomposition strategy. In the past decade, CC has been shown as one of the most effective solutions in multi-label empirical studies [19].[2] However, in some recent MDC literatures [16], it is shown that the directly generalized CC for MDC can no longer compete with state-of-the-art MDC approaches. This phenomenon might be due to the different characteristics of output variables which are multi-class type in MDC and binary type in MLC. Generally, learning a multi-class classifier is more difficult than a binary one with the same accuracy, while inaccurate predictions of preceding classifiers on the chain would degenerate the performance of subsequent ones. Existing MDC works related to CC [20] mainly aim at finding better chaining orders for the vanilla CC model which actually treat MLC and MDC equally with little discrimination and do not consider the special characteristics of MDC's output space.

In this paper, an MDC approach based on CC techniques is designed which builds a chain of binary classifiers similar to the version of CC in MLC setting. Typical strategies for transforming multi-class classification problem into binary classification problems include one-vs-rest (OvR), one-vs-one (OvO) and error-correcting output-coding (ECOC) [8], among which OvO has been shown as the better alternative by some comparative studies [11]. Therefore, the proposed approach chooses OvO decomposition to assist itself in building a chain of binary classifiers. The main contributions of this paper are summarized as follows:

- This paper proposes a first attempt towards considering the special characteristics of class variables when building chaining-based MDC approaches. Different from existing strategy which only focuses on finding better chaining orders, the proposed strategy provides a new perspective to induce chaining-based MDC models.
- By considering the multi-class nature of class variables, this paper makes a first attempt to build a chain of *binary* classifiers with the help of OvO decomposition to solve the MDC problem. To address the dilemma that different OvO decomposed problems involve different training examples, a "train-and-predict" strategy is employed in training phase.[3] To alleviate the effect of the specified chaining order, the ensemble version of the proposed approach is further investigated.
- Experimental results reported in Subsection IV-B show the superiority of the proposed approach against state-of-the-art MDC approaches and then validate the effectiveness of the proposed strategy. Detailed algorithmic properties (e.g., binary decomposition strategy, "train-and-predict" strategy, etc.) are further analyzed with comparative studies in Subsection IV-C.

In the following of this paper, for the sake of brevity, the basic version of the proposed approach is abbreviated as DCC, i.e., *Decomposition-based Classifier Chains*, and the ensemble version as EDCC, i.e., *Ensembles of DCC*, respectively.

The rest of this paper is organized as follows. Firstly, Section II briefly discusses some related works about MDC. Then, Section III presents the technical details of both the generalized classifier chains model and our specially designed classifier chains for MDC. After that, Section IV conducts comprehensive comparative studies to validate the effectiveness of the proposed approach. Finally, Section V concludes this paper.

II. Related Work

MDC has a close relationship with the widely-studied multi-label classification (MLC) [12]. Specifically, both MDC and MLC can be regarded as special cases of multi-output learning [28], where the type of each output variable corresponds to multi-class in MDC and binary-class in MLC. Therefore, MDC is also usually regarded as a generalized version of MLC by no longer restricting each MLC's class variable (i.e., label) to be binary-valued. This is why a few MLC models which do not consider the special characteristics of each binary label (e.g., classifier chains [23]), can be easily generalized to solve the MDC problem, while the only adjustment is to replace the employed binary classifier with a multi-class classifier and vice versa.[4] However, as labels in MLC are binary-valued, most existing MLC approaches which have considered the special characteristics of each binary label cannot be directly applicable for solving the MDC problem [32]. Conceptually, the fundamental differences between MLC and MDC lie in whether the semantic space is homogeneous or heterogeneous. The class variables in MDC correspond to heterogeneous class spaces, which characterize the semantics of objects along different dimensions, while labels in MLC correspond to one homogeneous class space which characterizes the relevancy of multiple concepts along one dimension [15].

The key challenge of learning from MDC data is to properly model the dependencies among class spaces when training predictive models. To deal with such challenge, one intuitive strategy, which is usually termed as class powerset (CP),[5] is to convert all class variables as a compound one via powerset-like transformation, where each distinct class combination in training set is treated as a new class. However, CP cannot return class combinations not appearing in training set for unseen instance and its computational complexity might be high due to huge number of possible new classes. These deficiencies can be alleviated by partitioning the class variables

[1]The technical details of this strategy will be presented in Subsection III-B.

[2]The original paper of CC [22] has won the Test of Time Award at ECML PKDD 2019 (https://www.ecmlpkdd2019.org/programme/awards/).

[3]Please see Fig.3 for an intuition on this special training procedure.

[4]In fact, there are indeed several MDC works which only evaluate their approaches over multi-label data sets [2].

[5]The name CP is derived from the MLC approach label powerset (LP) [25], and appears for the first time in [21].

into groups [21] but cannot be fully addressed due to the combinatorial nature. The gMML approach [18] solves the MDC problem by alternatingly learning multiple regression models (one per class label) as well as a Mahalanobis distance metric. The mechanism of metric learning can make the regression models being trained in a joint manner, and thus consider the class dependencies. One potential problem of gMML is that the modeling outputs of class labels from different dimensions are directly aligned which might be less reasonable due to the heterogeneous assumption of class spaces in MDC. Class dependencies can also be explicitly modeled by learning a DAG structure over class spaces [26], where different DAG structures correspond to different approaches which form a family of MDC models called multi-dimensional Bayesian network classifiers (MBC) [3]. Recent works about MBC mainly focus on dealing with the high computational complexity of DAG structure learning [36].

Binary relevance (BR)[6] learns from MDC data by decomposing the original MDC problem into a number of independent multi-class classification problems, one per dimension. This approach is usually criticized by ignoring all the potential class dependencies. An improved strategy is to learn the classifier chains (CC) model which is initially proposed in multi-label context [23] but can be easily generalized to solve the MDC problem. Similar to BR, the CC model also only needs to train q multi-class classifiers, one per dimension, but class dependencies can be considered by CC via the specified chaining structure. However, it is shown that the generalized CC cannot compete with state-of-the-art MDC approaches in some MDC literatures [16]. This difference might be caused by the different types of classifiers on the chain which is binary-class in MLC and multi-class in MDC, where it is more difficult to learn a multi-class classifier than a binary one with the same accuracy. Existing MDC works related to CC mainly aim at finding better chaining orders (e.g., random order [20] or deterministic order [30]) for the vanilla CC model which actually treat MLC and MDC equally with little discrimination, while the proposed approach in this paper aims at constructing *binary* classifier chains in the OvO decomposed label space tailored for MDC. To the best of our knowledge, this is the first attempt to consider the special characteristics of MDC's output space when building chaining-based models for MDC.

III. Multi-Dimensional Classification with Classifier Chains

The idea of learning classifier chains model is initially proposed in the multi-label context [23], but it can be easily generalized to solve the MDC problem. In this section, the definition of MDC is firstly given with strict mathematical notations, then the generalized classifier chains model under the MDC setting is introduced, and finally the technical details of the proposed DCC approach are presented.

[6]The name BR is borrowed from the multi-label field [31], though class relevance (CR) might be more appropriate in the MDC context.

TABLE I
Notations in the MDC setting.

Notation	Descriptions
d	number of features in input space
q	number of class spaces (dimensions) in output space
K_j	number of class labels in the jth class space ($1 \leq j \leq q$)
m	number of training examples
$\mathcal{X}$	the d dimensional input (feature) space, i.e., $\mathcal{X} = \mathbb{R}^d$
C_j	the jth class space where $C_j = \{c_1^j, \ldots, c_{K_j}^j\}$ ($1 \leq j \leq q$)
c_a^j	the ath class label in C_j ($1 \leq a \leq K_j$)
$\mathcal{Y}$	the output space where $\mathcal{Y} = C_1 \times C_2 \times \cdots \times C_q$
$\mathcal{D}$	the MDC training set where $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\}$
$\boldsymbol{x}_i$	the ith feature vector where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^\top \in \mathcal{X}$
$\boldsymbol{y}_i$	the ith class vector where $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$
$\boldsymbol{x}_*$	the unseen instance where $\boldsymbol{x}_* \in \mathcal{X}$
f	the MDC predictive model: $\mathcal{X} \mapsto \mathcal{Y}$ where $f(\boldsymbol{x}_*) \in \mathcal{Y}$

A. Formal Definition

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ be the input (feature) space,[7] 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., $C_j = \{c_1^j, c_2^j, \ldots, c_{K_j}^j\}$. Given the MDC training set $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\}$ with m training examples, for each example $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}$, $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^\top \in \mathcal{X}$ is a d-dimensional feature vector and $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ is the class vector associated with $\boldsymbol{x}_i$, where each component y_{ij} takes one possible class label in C_j, i.e., $y_{ij} \in C_j$. The goal of multi-dimensional classification is to learn a predictive model $f : \mathcal{X} \mapsto \mathcal{Y}$ from $\mathcal{D}$ which can assign a class vector $f(\boldsymbol{x}_*) \in \mathcal{Y}$ for unseen instance $\boldsymbol{x}_*$. To facilitate understanding, the notations are further summarized in Table I.

B. Classifier Chains

Let $\Psi(q) = \{\psi(1), \psi(2), \ldots, \psi(q)\}$ be one permutation of $\{1, 2, \ldots, q\}$, which specifies an order over 1 to q. In training phase, CC trains a multi-class classifier $g_{\psi(j)}$ for the $\psi(j)$-th dimension ($1 \leq j \leq q$) over the following multi-class data set:

$$\mathcal{D}_{\text{cc}}^{\psi(j)} = \{(\boldsymbol{x}_i^{\psi(j)}, y_{i\psi(j)}) \mid 1 \leq i \leq m\} \quad \text{where } \boldsymbol{x}_i^{\psi(j)} = [\boldsymbol{x}_i; y_{i\psi(1)}; \ldots; y_{i\psi(j-1)}]. \tag{1}$$

i.e., $g_{\psi(j)} = \mathcal{M}(\mathcal{D}_{\text{cc}}^{\psi(j)})$, where $\mathcal{M}$ corresponds to the employed multi-class algorithm. In other words, each instance will be appended with the class labels of preceding dimensions on the chain when constructing $\mathcal{D}_{\text{cc}}^{\psi(j)}$. By doing this, CC trains q cascaded multi-class classifiers in the specified order $\Psi(q)$, and thus class dependencies are expected to be considered. To facilitate understanding, Fig.2 shows an intuition on the training procedure of CC.

In prediction phase, for unseen instance $\boldsymbol{x}_*$, its predicted class label $\hat{y}_{*\psi(j)}$ for the $\psi(j)$-th dimension ($1 \leq j \leq q$) is determined as follows:

$$\hat{y}_{i\psi(j)} = g_{\psi(j)}(\boldsymbol{x}_*^{\psi(j)}), \quad \text{where } \boldsymbol{x}_*^{\psi(j)} = [\boldsymbol{x}_*; \hat{y}_{i\psi(1)}; \ldots; \hat{y}_{i\psi(j-1)}]. \tag{2}$$

[7]Note that the feature type can also be discrete-valued, $\mathcal{X} = \mathbb{R}^d$ only aims at brief notations.

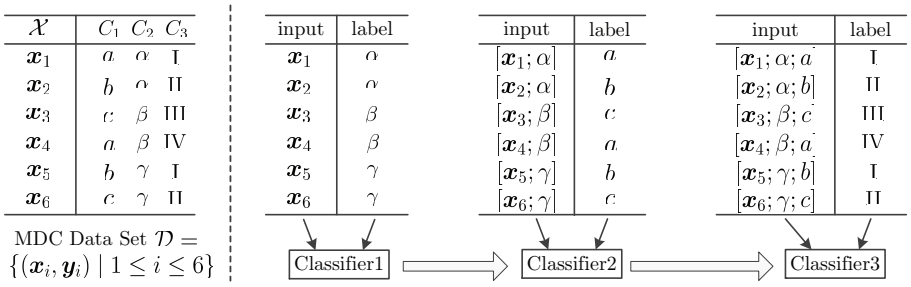


Fig. 2. An intuition on the training procedure of the CC approach. Specifically, an MDC data set $\mathcal{D}$ is shown to the left side of the dotted line, where the 6 examples (i.e., $m = 6$) are characterized from 3 dimensions (i.e, $q = 3$) with $C_1 = \{a, b, c\}$ (i.e., $K_1 = 3$), $C_2 = \{\alpha, \beta, \gamma\}$ (i.e., $K_2 = 3$) and $C_3 = \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ (i.e., $K_3 = 4$). The intuitive training procedure is shown to the right side of the dotted line with chaining order $C_2 \rightarrow C_1 \rightarrow C_3$.

Here, predictions are employed to be appended with the unseen instance just because its ground-truth class labels are unavailable.

Algorithm 1 The CC approach.

Input: The MDC training set $\mathcal{D}$, the order $\Psi(q)$, the employed multi-classifier $\mathcal{M}$, and the unseeen instance $\boldsymbol{x}_*$

Output: The predicted class vector $\boldsymbol{y}_*$ for $\boldsymbol{x}_*$

1: **for** $j = 1$ to q **do**
2: Construct the multi-class training set $\mathcal{D}_{cc}^{\psi(j)}$ according to Eq.(1);
3: Train multi-class classifier: $g_{\psi(j)} = \mathcal{M}(\mathcal{D}_{cc}^{\psi(j)})$;
4: **end for**
5: **for** $j = 1$ to q **do**
6: Determine the class label $\hat{y}_{*\psi(j)}$ according to Eq.(2);
7: **end for**
8: Return $\boldsymbol{y}_* = [\hat{y}_{*1}, \hat{y}_{*2}, \ldots, \hat{y}_{*q}]^\top$.

Algorithm 1 summarizes the complete procedure of CC. Specifically, steps 1-4 correspond to the training phase, and steps 5-8 correspond to the testing phase. Note that MDC assumes that the type of output variables is discrete-valued without ordinal relationship. If the employed $\mathcal{M}$ can only accept real-valued input features (e.g., support vector machine), class labels should be transformed into their one-hot form when they serve as features.

It is obvious that the specified order $\Psi(q)$ would affect the performance of CC. To alleviate the effect of the chaining order, it can construct *Ensembles of Classifier Chains* (ECC) with n different orders $\Psi_1(q), \Psi_2(q), \ldots, \Psi_n(q)$, and then combine the predictive results via majority voting. Moreover, instead of training all classifiers over the original training set $\mathcal{D}$, it would be beneficial to train each CC over a modified training set $\mathcal{D}_r$ $(1 \leq r \leq n)$ to increase the diversities of base learners. The n modified training sets can be obtained by randomly sampling the original training set $\mathcal{D}$ with replacement (i.e., Bagging [5]) or without replacement (e.g., selecting 67% examples [21]).

C. Decomposition-based Classifier Chains

Although empirical studies show CC (or its ensemble) achieves highly competitive performance in multi-label context [19], the generalized MDC version in Subsection III-B (or its ensemble) can no longer compete with state-of-the-art MDC approaches as shown in recent MDC studies [14], even though advanced techniques are utilized to help improve the quality of chaining orders. Regardless of the special distributions in different data sets, the biggest difference from multi-label CC to multi-dimensional CC is the different types of classifiers on the chain which is binary-class in MLC and multi-class in MDC. Based on this observation, it is speculated that if a chain of binary classifiers could be built to solve the MDC problem, the resulting model would achieve better generalization performance. Next, the technical details of the proposed DCC approach will be presented which consists of three parts, including binary decomposition, training phase, and testing phase.

1) Binary Decomposition: In order to solve the MDC problem via building a chain of binary classifiers, the MDC problem needs to be transformed into a number of binary classification problems. As OvO decomposition has been shown as the better alternative for transforming multi-class classification into binary classification [9], in this paper, the proposed approach simply chooses to decompose the MDC problem via OvO rule w.r.t. each dimension. Following the same notations given in previous sections, it is easy to know that the MDC problem can be transformed into a total of $T = \sum_{j=1}^{q} \binom{K_j}{2}$ binary classification problems. Without loss of generality, for the t-th decomposed problem $(1 \leq t \leq T)$, assume that it corresponds to the pair of class labels (p^t, n^t), where instances associated with label p^t (n^t) will be used as positive (negative) samples. For example, the first decomposed problem in Fig.3 corresponds to the pair of class labels (a, b), where $\boldsymbol{x}_1$ and $\boldsymbol{x}_4$ associated with label a are used as positive samples, and $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ associated with label b are used as negative samples. Here, let $\mathcal{I}_+^t$ and $\mathcal{I}_-^t$ be the index sets of examples which respectively own class labels p^t and n^t, the t-th decomposed problem corresponds to the following binary data set:

$$\mathcal{D}_{\mathrm{b}}^t = \{(\boldsymbol{x}_i, l_i^t) \mid i \in \mathcal{I}_+^t \cup \mathcal{I}_-^t\}, \text{where } l_i^t = \begin{cases} 1, & i \in \mathcal{I}_+^t \\ 0, & i \in \mathcal{I}_-^t \end{cases}$$

In other words, instances with label '1' ('0') in $\mathcal{D}_{\mathrm{b}}^t$ means that their original multi-class labels correspond to p^t (n^t).

2) Training Phase: To build a chain of binary classifiers, the significant difference between the decomposed T binary classification problems and an MLC problem with T labels

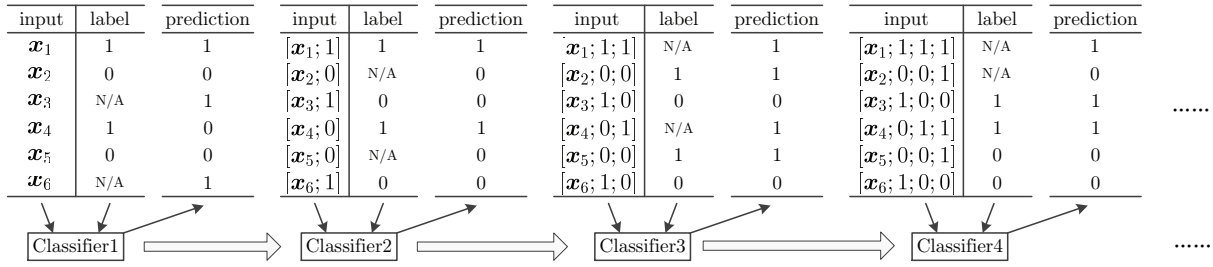


Fig. 3. An intuition on the training procedure of the proposed DCC approach. Specifically, with the same data set in Fig.2, here part of the intuitive training procedure is shown with chaining order $(a,b) \rightarrow (a,c) \rightarrow (b,c) \rightarrow (\beta,\gamma) \rightarrow \cdots$. Note that examples with label 'N/A' are not used to train the corresponding binary classifier. Besides, the prediction and its ground-truth are deliberately made different for some examples to demonstrate the possibility of wrong predictions (e.g., $\boldsymbol{x}_4$ in the first step).

is that each decomposed problem here corresponds to one different set of examples (i.e., $\mathcal{I}_+^t \cup \mathcal{I}_-^t$). As a result, it is impossible to construct a chain of data sets similar to Eq.(1) because examples involved in the current classifier might be uninvolved in the preceding classifiers on the chain. Therefore, the proposed approach tries to address this dilemma via a so-called "train-and-predict" strategy. Specifically, let $\Psi(T) = \{\psi(1), \psi(2), \ldots, \psi(T)\}$ be one permutation of $\{1, 2, \ldots, T\}$, which specifies an order over 1 to T. In training phase, DCC trains a binary classifier $h_{\psi(t)}$ for the $\psi(t)$-th decomposed OvO problem over the following data set:

$$\mathcal{D}_{\mathrm{dcc}}^{\psi(t)} = \{(\boldsymbol{x}_i^{\psi(t)}, l_i^{\psi(t)}) \mid i \in \mathcal{I}_+^{\psi(t)} \cup \mathcal{I}_-^{\psi(t)}\} \quad \text{where } \boldsymbol{x}_i^{\psi(t)} = [\boldsymbol{x}_i; \hat{l}_i^{\psi(1)}; \ldots; \hat{l}_i^{\psi(t-1)}]. \tag{3}$$

i.e., $h_{\psi(t)} = \mathcal{B}(\mathcal{D}_{\mathrm{dcc}}^{\psi(t)})$. Here, $\mathcal{B}$ corresponds to the employed binary classification algorithm, $l_i^{\psi(t)} = 1$ if $i \in \mathcal{I}_+^{\psi(t)}$ and $l_i^{\psi(t)} = 0$ if $i \in \mathcal{I}_-^{\psi(t)}$, and $\hat{l}_i^{\psi(t)} = h_{\psi(t)}(\boldsymbol{x}_i^{\psi(t)})$. The "train-and-predict" strategy refers to that the classifier $h_{\psi(t)}$ is firstly *trained* over $\mathcal{D}_{\mathrm{dcc}}^{\psi(t)}$ which only involves training examples whose indexes belong to the union of $\mathcal{I}_+^{\psi(t)}$ and $\mathcal{I}_-^{\psi(t)}$, and then the binary classes are *predicted* for all training examples with $h_{\psi(t)}$ to prepare for learning subsequent classifiers. To facilitate understanding, Fig.3 shows an intuition on the training procedure of DCC.

3) Testing Phase: When making prediction for unseen instance $\boldsymbol{x}_*$, its predicted binary label $\hat{l}_*^{\psi(t)}$ for $\psi(t)$-th decomposed OvO problem ($1 \leq t \leq T$) is determined as follows:

$$\hat{l}_*^{\psi(t)} = h_{\psi(t)}(\boldsymbol{x}_*^{\psi(t)}), \quad \text{where } \boldsymbol{x}_*^{\psi(t)} = [\boldsymbol{x}_*; \hat{l}_*^{\psi(1)}; \ldots; \hat{l}_*^{\psi(t-1)}]. \tag{4}$$

After the predicted binary label vector $\boldsymbol{l}_* = [\hat{l}_*^1, \hat{l}_*^2, \ldots, \hat{l}_*^T]^\top$ is obtained, the class vector $\boldsymbol{y}_* = [y_{*1}, \ldots, y_{*q}]^\top$ of $\boldsymbol{x}_*$ can be determined via OvO decoding rule.

Algorithm 2 summarizes the complete procedure of DCC. Specifically, step 1 corresponds to the binary decomposition, steps 2-6 correspond to the training phase, and steps 7-10 correspond to the testing phase. As a chaining-based classifier, the performance of DCC would also be affected by the specified order $\Psi(T)$ similar to CC. Thus, the effect of the chaining order can also be alleviated by constructing the *Ensembles of DCC* (EDCC).

Algorithm 2 The DCC approach.

Input: The MDC training set $\mathcal{D}$, the order $\Psi(T)$, the employed binary classifier $\mathcal{B}$, and the unseeen instance $\boldsymbol{x}_*$

Output: The predicted class vector $\boldsymbol{y}_*$ for $\boldsymbol{x}_*$

1: Decompose the MDC training set into $T = \sum_{j=1}^{q} \binom{K_j}{2}$ binary classification data sets via OvO rule w.r.t. each dimension;
2: **for** $t = 1$ to T **do**
3: Construct the binary classification training set $\mathcal{D}_{\mathrm{dcc}}^{\psi(t)}$ according to Eq.(3);
4: Train binary classifier: $h_{\psi(t)} = \mathcal{B}(\mathcal{D}_{\mathrm{dcc}}^{\psi(t)})$;
5: Obtain the predictions $l_i^{\psi(t)}$ of all training examples $\boldsymbol{x}_i$ ($1 \leq i \leq m$);
6: **end for**
7: **for** $t = 1$ to T **do**
8: Determine the class label $\hat{l}_*^{\psi(t)}$ according to Eq.(4);
9: **end for**
10: Determine $\boldsymbol{y}_*$ based on $\boldsymbol{l}_* = [\hat{l}_*^1, \hat{l}_*^2, \ldots, \hat{l}_*^T]^\top$ via OvO decoding rule.

Note that the hard-predicted labels (i.e., 0/1) are simply employed to augment the feature space for all examples (i.e., examples with both valid labels '0/1' and invalid label 'N/A') in training phase. Some potential advantages include: (1) Compared with specifically employing the ground-truth binary label for examples with valid labels '0/1', the distribution in the augmented feature space might be more consistent with independent and identically distributed (*i.i.d.*) assumption if predicted labels are employed for all examples (also including unseen instance in testing phase). (2) Generally, using either soft or hard predicted labels will bring noise to examples. However, hard labels might be more robust to noise than soft labels especially when labels are binary (cf. ablation study in Subsection IV-C6). (3) Another minor issue is that it is easier to obtain hard predicted labels than soft ones as all binary classifiers can return hard predicted labels. Besides, a good alternative solution to utilize examples with invalid label is to use an additional class to indicate 'N/A' which is deserved to be explored in the future.

It is also worth noting that the proposed DCC model has some similarities with deep stacking networks (DSN) [7]. Specifically, both of them need to train multiple learners in

a serial mode and will augment the feature space with predictions of preceding learners to train the subsequent learners. However, there are also many differences: (1) In DSN, all learners is trained under the same supervised information, while in DCC, different classifiers on the chain corresponds to different labeling information. As a result, the training order of learners will affect the performance of DCC while this problem does not exist in DSN. (2) In DSN, the number of learners is a hyper-parameter which should be set or tuned manually, while in DCC, the number of learners is a fixed number which corresponds to the total number of decomposed binary classification problems. (3) In DSN, the training process includes two steps: block training and fine-tuning, while in DCC, the training process only includes the block training step. Nonetheless, the idea of DSN can be borrowed to build a deep learning model to deal with the current task which can be further explored in the future. Preliminary idea is to fix the number of learners as T and replace the supervised information for each learner with the corresponding labeling information of each decomposed binary classification problem.

IV. Experiments

In this section, comparative studies are conducted to validate the effectiveness of the proposed approach. Specifically, the experimental setup is introduced in Subsection IV-A. Then, the experimental results between the proposed approach and the compared baselines are reported in Subsection IV-B. Finally, some properties of the proposed approach are further analyzed in Subsection IV-C.

A. Experimental Setup

In the next, the detailed information of benchmark data sets are firstly introduced in Subsection IV-A1, then the concrete definitions of evaluation metrics are given in Subsection IV-A2, and finally the compared approaches (including their parameter settings) are presented in Subsection IV-A3.

1) Benchmark Data Sets: In this paper, the comparative studies are conducted over a total of 20 benchmark data sets. Table II summarizes the detailed information of all the employed data sets, including the *number of examples* (#Exam.), the *number of dimensions* (#Dim.), the *number of class labels per dimension* (#Labels/Dim.),[8] and the *number of features* (#Features).

2) Evaluation Metrics: In this paper, the generalization performance of MDC approaches are evaluated by three metrics, i.e., *Hamming Score* (HS), *Exact Match* (EM) and *Sub-Exact Match* (SEM), which have been widely used in recent MDC studies [15]. Specifically, given the MDC test set $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq p\}$ and the MDC model $f : \mathcal{X} \mapsto \mathcal{Y}$ to be evaluated, let $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ and $\hat{\boldsymbol{y}}_i = f(\boldsymbol{x}_i) = [\hat{y}_{i1}, \hat{y}_{i2}, \ldots, \hat{y}_{iq}]^\top$ be the ground-truth and predicted class vector for test example $\boldsymbol{x}_i$, then the number of class labels for which f returns the correct predictions can be calculated via $r^{(i)} = \sum_{j=1}^{q} \mathbf{1}_{y_{ij}=\hat{y}_{ij}}$. Here, predicate $\mathbf{1}_\pi$ returns 1 if π holds and 0 otherwise. The three evaluation metrics can be formally defined as follows:

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

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

$$\mathrm{SEM}_{\mathcal{S}}(f) = \frac{1}{p}\sum_{i=1}^{p}\mathbf{1}_{r^{(i)}\geq q-1}$$

According to the above definitions, it is easy to know that the *larger* the metric values, the *better* the generalization performance. For each compared approach, *ten-fold cross validation* is conducted over each data set, where both the mean metric value as well as standard deviation are recorded for performance comparison.

3) Compared Approaches: As shown in [23], the ensemble strategy will boost the performance of CC generally. Therefore, this paper aims at comparing the performance of the ensemble version of DCC (i.e., EDCC) with eight state-of-the-art MDC approaches:[10]

- ECC, i.e., Ensembles of Classifier Chains introduced in Subsection III-B, is the basic classifier chains model for MDC. Following [21], ECC builds a total of 10 base CC classifiers with different *random* chaining orders and combines the predictions via majority voting.
- EBCC, i.e., Ensembles of Bayesian Classifier Chains [30], incorporates a simplified Bayesian framework to obtain q *deterministic* chaining orders. EBCC builds a total of q BCC classifiers with the obtained q chaining orders and combines the predictions via majority voting.
- BR, i.e., Binary Relevance, decomposes the MDC problem into q independent multi-class classification problems and solves them one by one. BR ignores all possible class

TABLE II
Characteristics of the benchmark data sets.

Data Set	#Exam.	#Dim.	#Labels/Dim.	#Features[9]
Oes97	334	16	3	263n
Jura	359	2	4,5	9n
Oes10	403	16	3	298n
Song	785	3	3	98n
WQplants	1060	7	4	16n
WQanimals	1060	7	4	16n
WaterQuality	1060	14	4	16n
BeLaE	1930	5	5	44n,1x
Voice	3136	2	4,2	19n
Scm20d	8966	16	4	61n
Rf1	8987	8	4,4,3,4,4,3,4,3	64n
Thyroid	9172	7	5,5,3,2,4,4,3	7n, 22x
Pain	9734	10	2,5,4,2,2,5,2,5,2,2	136n
Scm1d	9803	16	4	280n
CoIL2000	9822	5	6,10,10,4,2	81x
Flickr	12198	5	3,4,3,4,4	1536n
Disfa	13095	12	5,5,6,3,4,4,5,4,4,4,6,4	136n
Fera	14052	5	6	136n
Adult	18419	4	7,7,5,2	5n,5x
Default	28779	4	2,7,4,2	14n,6x

[8]Here, the numbers for all dimensions are recorded in turn. However, if all numbers are the same to each other, then only one of them is recorded.

[9]Here, n and x denote numeric and nominal features, respectively.

[10]The benefit brought by the ensemble strategy for DCC will be further analyzed in Subsection IV-C1.

TABLE III
EXPERIMENTAL RESULTS (MEAN ± STD.) WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE (BASE CLASSIFIER: LOGISTIC REGRESSION).

Data Set	*Hamming Score*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	**.746±.022**	.733±.023	.737±.027	.736±.028	.523±.049	.732±.024	.734±.029	.724±.023	.661±.032
Jura	**.618±.056**	.613±.073	.610±.070	.605±.067	.592±.071	.593±.068	.592±.071	.606±.072	.593±.067
Oes10	**.797±.018**	.793±.016	.795±.020	.796±.017	.687±.037	.779±.022	.792±.017	.775±.017	.711±.021
Song	**.793±.028**	.789±.026	.789±.023	.789±.024	.783±.025	.790±.024	.781±.024	.788±.027	.773±.027
WQplants	**.660±.015**	.653±.016	.655±.015	.658±.014	.649±.016	.653±.016	.658±.014	.655±.015	.652±.016
WQanimals	**.636±.012**	.631±.013	.629±.014	.631±.013	.628±.013	.631±.014	.630±.014	.630±.015	.630±.013
WQ	**.648±.012**	.642±.012	.644±.012	.644±.011	.625±.011	.642±.014	.644±.012	.643±.013	.640±.013
BeLaE	**.449±.016**	.423±.019	.424±.018	.427±.017	.383±.023	.420±.022	.409±.019	.417±.020	.437±.019
Voice	.950±.009	.900±.015	.903±.013	.900±.012	.898±.011	.897±.011	.898±.011	.842±.009	**.962±.008**
Scm20d	**.690±.005**	.629±.006	.622±.006	.649±.005	.618±.010	.637±.008	.664±.006	.600±.007	N/A
Rf1	**.904±.004**	.846±.003	.842±.004	.835±.004	.885±.004	.864±.005	.857±.004	.730±.007	N/A
Thyroid	**.965±.002**	**.965±.002**	**.965±.002**	**.965±.002**	.963±.003	.963±.002	.964±.002	.960±.002	.961±.003
Pain	**.954±.003**	.953±.003	.952±.004	.953±.003	.952±.004	.952±.004	.953±.003	.948±.004	N/A
Scm1d	**.833±.003**	.743±.005	.731±.004	.762±.004	.703±.009	.733±.005	.771±.005	.697±.007	N/A
CoIL2000	.938±.003	.925±.004	.924±.005	.924±.005	.936±.005	**.952±.003**	.938±.004	.894±.004	N/A
Flickr	.800±.004	.800±.005	**.801±.004**	**.801±.005**	.768±.003	.795±.003	.799±.005	.779±.004	N/A
Disfa	**.907±.002**	.894±.002	.896±.002	.896±.002	.886±.003	.890±.003	.895±.002	.884±.003	N/A
Fera	**.653±.006**	.618±.008	.621±.007	.626±.007	.601±.008	.598±.009	.622±.007	.589±.007	N/A
Adult	**.723±.003**	.712±.004	.708±.003	.721±.004	.709±.004	.710±.005	.721±.003	.705±.004	.699±.004
Default	.671±.004	.670±.003	.669±.003	.669±.003	.669±.004	**.672±.004**	.671±.003	.666±.004	N/A
Data Set	*Exact Match*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	.057±.048	.057±.048	.054±.044	**.060±.045**	.054±.046	.054±.044	.057±.048	.042±.038	.051±.043
Jura	**.387±.104**	.379±.118	.371±.112	.359±.107	.357±.108	.351±.096	.357±.108	.368±.119	.348±.108
Oes10	**.099±.045**	.094±.047	.097±.045	.084±.033	.082±.049	**.099±.037**	.092±.035	.079±.040	.084±.049
Song	**.494±.054**	.485±.061	.485±.058	.481±.060	.479±.050	.485±.060	.469±.053	.484±.059	.453±.056
WQplants	.094±.036	.092±.036	.094±.034	.092±.033	.093±.031	.093±.036	**.095±.032**	.092±.035	.092±.031
WQanimals	.064±.017	.062±.018	.058±.018	.058±.017	**.065±.018**	.064±.019	.063±.024	.062±.023	.058±.022
WQ	**.007±.008**	.006±.008	.004±.007	.005±.008	.000±.000	.005±.008	.006±.008	.006±.008	**.007±.008**
BeLaE	**.033±.013**	.030±.011	.031±.010	.021±.008	.026±.014	.027±.009	.023±.011	.022±.009	.031±.010
Voice	.902±.018	.809±.027	.814±.023	.809±.023	.807±.021	.803±.019	.807±.021	.699±.017	**.926±.016**
Scm20d	.083±.008	.085±.007	.086±.006	.058±.008	**.132±.010**	.095±.009	.076±.009	.052±.007	N/A
Rf1	.485±.017	.325±.014	.296±.016	.288±.015	**.509±.011**	.425±.017	.351±.014	.138±.011	N/A
Thyroid	.770±.015	**.771±.014**	.770±.014	.769±.014	.762±.014	.761±.013	.764±.013	.741±.015	.748±.015
Pain	**.760±.015**	.759±.015	.756±.016	.755±.015	**.760±.017**	.758±.017	.756±.016	.750±.018	N/A
Scm1d	**.197±.011**	.147±.013	.143±.011	.121±.009	.180±.015	.170±.016	.146±.010	.102±.009	N/A
CoIL2000	.747±.016	.692±.016	.686±.019	.686±.018	.767±.016	**.821±.010**	.743±.017	.576±.015	N/A
Flickr	.331±.016	.331±.013	**.338±.012**	.333±.013	.295±.012	.331±.009	.333±.016	.287±.009	N/A
Disfa	**.417±.009**	.393±.011	.397±.009	.396±.009	.403±.012	.393±.012	.395±.010	.379±.011	N/A
Fera	.215±.013	.203±.013	.204±.011	.203±.013	**.219±.012**	.213±.012	.208±.014	.196±.013	N/A
Adult	.281±.009	.310±.008	.310±.007	.275±.008	**.317±.010**	.312±.011	.297±.008	.230±.009	.216±.010
Default	.185±.009	.186±.006	.184±.007	.181±.007	**.194±.008**	.187±.007	.187±.008	.177±.007	N/A
Data Set	*Sub-Exact Match*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	**.123±.076**	.114±.063	.120±.068	**.123±.065**	.084±.062	.111±.060	.111±.063	.099±.055	.093±.062
Jura	**.850±.044**	.847±.042	**.850±.051**	**.850±.049**	.827±.053	.836±.059	.827±.053	.844±.049	.838±.049
Oes10	.201±.064	**.211±.060**	**.211±.073**	.198±.055	.169±.042	.201±.056	.201±.050	.176±.038	.146±.052
Song	.885±.044	.885±.035	.885±.032	**.888±.035**	.875±.035	.887±.041	.876±.029	.883±.041	.869±.040
WQplants	.286±.046	.278±.052	.281±.044	.286±.044	.285±.052	.282±.049	**.292±.050**	.286±.053	.287±.048
WQanimals	**.234±.032**	.231±.026	.227±.032	.229±.030	.232±.032	.231±.029	.231±.032	.227±.033	.231±.030
WQ	.048±.025	.048±.024	.045±.019	.047±.019	.034±.017	.048±.019	.045±.020	**.049±.024**	.045±.023
BeLaE	**.158±.023**	.135±.016	.146±.018	.134±.025	.117±.019	.128±.024	.127±.019	.130±.020	.151±.023
Voice	.997±.003	.990±.007	.991±.006	.991±.006	.989±.006	.991±.007	.989±.006	.985±.011	**.998±.002**
Scm20d	.160±.007	.147±.007	.145±.007	.115±.009	**.209±.011**	.162±.010	.147±.009	.100±.009	N/A
Rf1	**.804±.009**	.661±.012	.663±.014	.624±.011	.758±.013	.703±.014	.694±.014	.375±.014	N/A
Thyroid	**.983±.004**	.982±.004	**.983±.004**	**.983±.004**	.982±.005	.982±.004	.982±.004	.982±.005	.982±.004
Pain	**.866±.010**	.859±.011	.856±.011	.862±.011	.858±.010	.858±.011	.862±.010	.846±.010	N/A
Scm1d	**.372±.016**	.251±.014	.241±.011	.234±.015	.284±.017	.275±.016	.275±.014	.198±.015	N/A
CoIL2000	.947±.005	.938±.008	.937±.007	.937±.008	.934±.008	**.953±.005**	.950±.006	.903±.010	N/A
Flickr	.737±.012	.736±.011	**.740±.009**	.736±.011	.671±.010	.725±.012	.734±.009	.689±.016	N/A
Disfa	**.677±.009**	.629±.010	.636±.012	.632±.012	.605±.010	.614±.011	.628±.012	.590±.009	N/A
Fera	**.460±.012**	.414±.011	.416±.011	.417±.010	.408±.011	.403±.012	.422±.010	.378±.013	N/A
Adult	**.687±.007**	.647±.006	.642±.007	.685±.009	.637±.007	.644±.007	.676±.008	.669±.008	.658±.008
Default	**.604±.008**	.600±.006	.600±.006	.601±.006	.594±.008	**.604±.008**	.603±.006	.593±.008	N/A

dependencies which might degenerate its generalization abilities.

- CP, i.e., Class Powerset, solves the MDC problem as a single multi-class classification problem by regarding the q output variables as a compound one. CP considers all possible class dependencies in training set. However, not all class combinations will appear in the limited training examples which might lead to overfitting problems.
- ESC, i.e., Ensembles of Super Class classifier [21], partitions the q class variables into several groups where each group will be regarded as a compound class variable. The resulting problem only considers the dependencies among class variables in the same group, and other class dependencies can be further considered by the subsequent solution.
- SEEM, i.e., Stacked dEpendency Exploitation for MD-C [16], works in a two-level style, where a classifier is trained for each pairwise class spaces to model the

TABLE IV
EXPERIMENTAL RESULTS (MEAN ± STD.) WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE (BASE CLASSIFIER: SVM).

Data Set	*Hamming Score*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	**.743±.024**	.733±.022	.725±.030	.694±.026	.485±.064	.713±.026	.689±.035	.724±.023	.661±.032
Jura	.607±.075	.609±.070	.588±.084	.586±.080	**.614±.083**	.588±.074	**.614±.083**	.606±.072	.593±.067
Oes10	**.800±.016**	.797±.019	.796±.017	.773±.018	.519±.058	.776±.022	.771±.023	.775±.017	.711±.021
Song	.789±.027	**.792±.025**	.789±.022	.791±.022	.749±.033	.790±.026	.774±.019	.788±.027	.773±.027
WQplants	.658±.013	.654±.013	.654±.013	.656±.016	.646±.014	.656±.014	**.659±.014**	.655±.015	.652±.016
WQanimals	**.635±.012**	.634±.011	.634±.013	.633±.011	.621±.012	.631±.012	**.635±.014**	.630±.015	.630±.013
WQ	.646±.012	.643±.011	.644±.012	.645±.011	.591±.018	.641±.013	**.647±.011**	.643±.013	.640±.013
BeLaE	**.439±.017**	.397±.019	.379±.023	.395±.012	.327±.018	.390±.018	.388±.019	.417±.020	.437±.019
Voice	.946±.007	.890±.014	.891±.013	.884±.014	.898±.012	.882±.012	.898±.012	.842±.009	**.962±.008**
Scm20d	.689±.007	.622±.007	.609±.007	.627±.008	**.721±.010**	.640±.044	.646±.010	.600±.007	N/A
Rf1	**.912±.003**	.833±.004	.832±.004	.800±.005	.898±.004	.889±.006	.837±.004	.730±.007	N/A
Thyroid	**.965±.002**	**.965±.003**	**.965±.002**	.964±.003	.963±.003	.963±.003	.963±.003	.960±.002	.961±.003
Pain	.954±.003	.952±.004	.952±.004	.952±.003	**.956±.003**	.955±.003	.952±.004	.948±.004	N/A
Scm1d	**.837±.002**	.729±.006	.716±.006	.731±.004	.823±.006	.800±.033	.754±.003	.697±.007	N/A
CoIL2000	.950±.003	.933±.004	.930±.005	.930±.004	.931±.005	**.954±.006**	.944±.003	.894±.004	N/A
Flickr	.795±.004	**.796±.003**	**.796±.004**	**.796±.004**	.730±.004	.785±.007	.795±.005	.779±.004	N/A
Disfa	**.907±.002**	.896±.002	.896±.003	.896±.002	.905±.002	.903±.004	.895±.003	.884±.003	N/A
Fera	.645±.007	.615±.007	.615±.006	.616±.008	.605±.027	**.672±.009**	.630±.004	.589±.007	N/A
Adult	**.712±.004**	.693±.004	.671±.007	.686±.007	.539±.044	.639±.007	.655±.009	.705±.004	.699±.004
Default	.665±.003	**.670±.003**	.667±.004	.667±.003	.467±.024	.620±.014	.585±.013	.666±.004	N/A
Data Set	*Exact Match*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	**.054±.044**	.051±.043	.048±.041	.018±.015	**.054±.046**	.045±.041	.042±.032	.042±.038	.051±.043
Jura	.365±.117	.365±.116	.331±.120	.318±.120	**.401±.110**	.331±.105	**.401±.110**	.368±.119	.348±.108
Oes10	.092±.047	.089±.042	**.097±.035**	.050±.031	.084±.047	.089±.043	.082±.039	.079±.040	.084±.049
Song	.475±.058	**.488±.058**	.478±.053	.480±.054	.437±.064	.486±.059	.461±.051	.484±.059	.453±.056
WQplants	**.096±.035**	.091±.032	.090±.033	.092±.035	.089±.029	.092±.034	.094±.025	.092±.035	.092±.031
WQanimals	**.067±.020**	.062±.016	.056±.023	.063±.021	.052±.019	.058±.016	.064±.015	.062±.023	.058±.022
WQ	**.008±.007**	.006±.008	.006±.008	.007±.008	.003±.006	.007±.010	.005±.007	.006±.008	.007±.008
BeLaE	.026±.011	.025±.007	.017±.009	.013±.011	.009±.007	.013±.006	.017±.008	.022±.009	**.031±.010**
Voice	.893±.014	.789±.027	.792±.023	.777±.026	.809±.022	.774±.025	.809±.022	.699±.017	**.926±.016**
Scm20d	.096±.008	.081±.010	.082±.007	.043±.003	**.154±.007**	.039±.019	.069±.008	.052±.007	N/A
Rf1	.505±.011	.305±.013	.299±.014	.216±.011	**.587±.015**	.526±.033	.319±.016	.138±.011	N/A
Thyroid	**.775±.013**	.773±.015	.771±.014	.766±.016	.760±.015	.762±.015	.762±.016	.741±.015	.748±.015
Pain	.762±.015	.759±.016	.761±.016	.757±.016	**.772±.015**	.765±.016	.754±.017	.750±.018	N/A
Scm1d	.205±.012	.140±.014	.139±.012	.099±.007	**.216±.014**	.154±.034	.131±.009	.102±.009	N/A
CoIL2000	.793±.010	.724±.017	.713±.017	.713±.015	.762±.015	**.812±.024**	.767±.013	.576±.015	N/A
Flickr	.323±.014	.327±.013	**.328±.015**	.323±.013	.252±.013	.315±.018	.326±.013	.287±.009	N/A
Disfa	.424±.010	.399±.010	.400±.011	.396±.010	**.463±.015**	.420±.013	.395±.012	.379±.011	N/A
Fera	.215±.013	.203±.011	.201±.011	.200±.010	.253±.017	**.271±.013**	.219±.011	.196±.013	N/A
Adult	**.260±.008**	.228±.014	.225±.013	.181±.017	.129±.015	.190±.005	.180±.013	.230±.009	.216±.010
Default	.177±.006	**.187±.007**	.181±.007	.180±.007	.035±.014	.141±.012	.104±.012	.177±.007	N/A
Data Set	*Sub-Exact Match*								
	EDCC	ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
Oes97	**.117±.086**	.114±.060	.096±.069	.063±.041	.081±.058	.108±.076	.090±.056	.099±.055	.093±.062
Jura	.850±.053	.852±.039	.844±.060	**.855±.045**	.827±.070	.844±.054	.827±.070	.844±.049	.838±.049
Oes10	**.213±.066**	.208±.063	.198±.056	.161±.072	.124±.053	.209±.061	.179±.051	.176±.038	.146±.052
Song	**.896±.049**	.890±.031	.892±.032	**.896±.035**	.825±.047	.885±.040	.868±.024	.883±.041	.869±.040
WQplants	**.296±.049**	.291±.046	.291±.048	.292±.045	.281±.046	.294±.047	.292±.052	.286±.053	.287±.048
WQanimals	**.241±.029**	.238±.029	.238±.036	.228±.037	.221±.023	.234±.040	.237±.046	.227±.033	.231±.030
WQ	**.056±.028**	.048±.024	.043±.026	.045±.028	.036±.023	.038±.018	.046±.017	.049±.024	.045±.023
BeLaE	.149±.036	.112±.027	.095±.025	.097±.018	.061±.013	.105±.026	.109±.022	.130±.020	**.151±.023**
Voice	**.998±.003**	.991±.007	.991±.007	.991±.006	.987±.006	.990±.007	.987±.006	.985±.011	**.998±.002**
Scm20d	.170±.008	.140±.010	.136±.009	.092±.009	**.294±.011**	.085±.038	.131±.013	.100±.009	N/A
Rf1	**.833±.008**	.628±.013	.616±.009	.535±.013	.824±.010	.794±.014	.647±.012	.375±.014	N/A
Thyroid	.981±.003	.980±.004	.981±.004	**.982±.004**	.981±.005	**.982±.004**	.981±.005	**.982±.005**	**.982±.004**
Pain	.866±.008	.856±.011	.856±.011	.856±.010	**.871±.009**	.866±.011	.857±.010	.846±.010	N/A
Scm1d	.388±.012	.238±.014	.234±.012	.202±.013	**.406±.014**	.305±.069	.250±.010	.198±.015	N/A
CoIL2000	**.961±.006**	.945±.008	.942±.010	.942±.009	.920±.011	**.961±.008**	.955±.006	.903±.010	N/A
Flickr	.725±.011	.729±.010	**.730±.008**	.729±.013	.597±.016	.704±.014	.729±.014	.689±.016	N/A
Disfa	.671±.010	.636±.010	.635±.014	.631±.014	**.679±.007**	.668±.013	.631±.013	.590±.009	N/A
Fera	.450±.009	.412±.012	.409±.009	.408±.010	.438±.029	**.503±.015**	.433±.010	.378±.013	N/A
Adult	.666±.011	.641±.013	.591±.022	.650±.011	.379±.048	.521±.019	.593±.017	**.669±.008**	.658±.008
Default	.591±.008	**.601±.008**	.594±.008	.596±.009	.255±.043	.508±.028	.445±.025	.593±.008	N/A

second-order dependencies in the first level, and then predictions from the first level w.r.t. each dimension are further aggregated to obtain the final prediction for unseen instance in the second level.[11]

- gMML solves the MDC problem by alternatingly learning multiple regression models (one per class label) as well as a Mahalanobis distance metric [18], where the learned metric can gradually help the regression models work better. Note that gMML works without any additional base classifiers, which is different from the proposed EDCC and the aforementioned six compared approaches.
- M3MDC, i.e., MaxiMum Margin Multi-Dimensional Classification [17],[12] adapts the popular and powerful maximum margin techniques for solving MDC problem.

[11]For fair comparison between EDCC and SEEM, the predictions are also combined via majority voting in the second level for SEEM [16].

[12]Ref.[17] is an extended version of preliminary work [14].

TABLE V
WILCOXON SIGNED-RANKS TEST FOR EDCC AGAINST EACH COMPARED APPROACH WHERE THE p-VALUES AT 0.05 SIGNIFICANCE LEVEL ARE ALSO SHOWN IN THE BRACKETS.

Evaluation Metric		EDCC against							
		ECC	EBCC	BR	CP	ESC	SEEM	gMML	M3MDC
LR	HS	**win**[1.03e-04]	**win**[1.40e-04]	**win**[1.20e-04]	**win**[8.86e-05]	**win**[5.17e-04]	**win**[1.40e-04]	**win**[8.86e-05]	**win**[1.37e-02]
	EM	**win**[7.90e-03]	**win**[1.12e-02]	**win**[2.53e-04]	**tie**[4.01e-01]	**tie**[8.36e-02]	**win**[4.26e-03]	**win**[8.84e-05]	**win**[3.71e-02]
	SEM	**win**[8.37e-04]	**win**[1.32e-03]	**win**[5.39e-04]	**win**[7.80e-04]	**win**[2.14e-03]	**win**[1.02e-03]	**win**[1.82e-04]	**win**[8.79e-03]
SVM	HS	**win**[2.82e-03]	**win**[3.38e-04]	**win**[4.49e-04]	**win**[8.92e-04]	**win**[1.94e-03]	**win**[8.92e-04]	**win**[1.03e-04]	**win**[3.22e-02]
	EM	**win**[6.42e-03]	**win**[1.71e-03]	**win**[3.38e-04]	**tie**[3.70e-01]	**tie**[5.69e-02]	**win**[2.49e-03]	**win**[3.90e-04]	**tie**[8.30e-02]
	SEM	**win**[1.51e-03]	**win**[2.93e-04]	**win**[8.92e-04]	**win**[5.11e-03]	**win**[1.32e-03]	**win**[1.40e-04]	**win**[2.19e-04]	**win**[1.27e-02]

Specifically, M3MDC maximizes the margins between each pair of class labels in the same class space via OvO decomposition and considers the class dependencies with covariance regularization. Note that M3MDC also works without any additional base classifiers similar to gMML.

In this paper, both logistic regression (LR) with cross-entropy loss and support vector machine (SVM) with hinge loss are employed to implement the base binary/multi-class classifier for all approaches (except gMML and M3MDC which do not necessitate a base classifier). Specifically, the efficient LIBLINEAR package [10] is employed to implement the two kinds of classifiers with parameter setting "L2-regularized logistic regression (primal)" for LR and "L2-regularized L1-loss support vector classification (dual)" for SVM.[13] For ESC [21], gMML [18] and M3MDC [17], the recommended parameter setting in the corresponding literatures are used. Similar to ECC, the proposed EDCC approach also builds a total of 10 base DCC models with different random chaining orders and combines the predictive results via majority voting.

It is worth noting that deep learning-based models are not included in compared approaches due to the following two reasons: (1) To the best of our knowledge, there are no existing deep learning-based MDC approaches. (2) The existing deep MLC models [32] have considered the special characteristics of MLC's homogeneous class space which makes them unusable for MDC problem who has multiple heterogeneous class spaces.

B. Experimental Results

Tables III and IV report the detailed experimental results of all the compared approaches with base classifier *LR* and *SVM*, respectively. The best performance w.r.t. each data set is highlighted via being shown in boldface. Note that both gMML and M3MDC do not necessitate a base classifier, and their experimental results are identical in the two tables.[14]

Moreover, statistical test is also conducted to judge about the significance of the experimental results. Specifically, *Wilcoxon signed-ranks test* [6] (at 0.05 significance level) is employed to serve as the statistical tool to show whether EDCC achieves statistically superior performance against ECC, EBCC, BR, CP, ESC, SEEM, gMML and M3MDC over the twenty benchmark data sets. Table V summarizes the corresponding test results where the p-values are also shown in the brackets.[15]

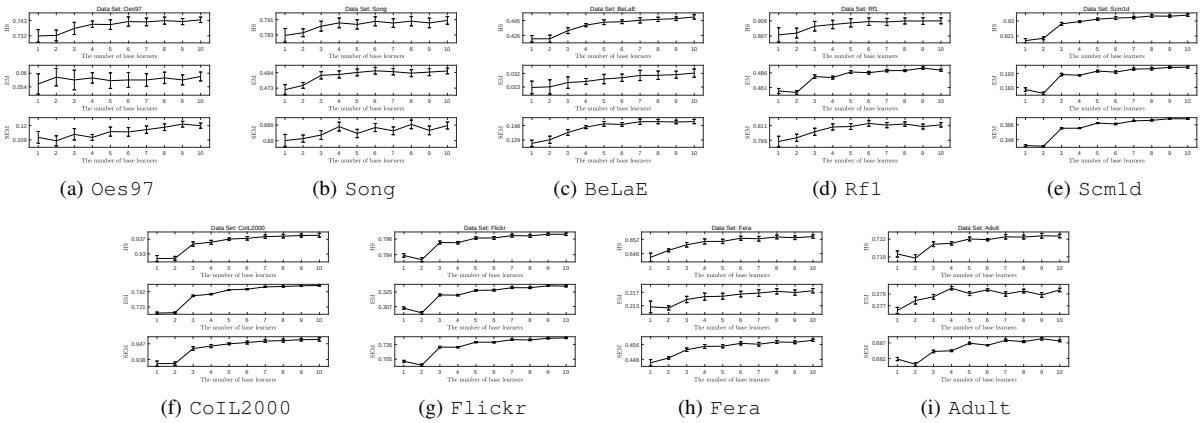
According to the reported results, the following observations can be made:

- Among all the 120 cases (20 data sets $\times$ 3 evaluation metrics $\times$ 2 base classifiers), EDCC ranks first in 60 cases, ranks second in 33 cases, ranks third in 11 cases, and never ranks last.
- Both ECC and EBCC do not consider the special characteristics of output space in MDC. As shown in Table V, EDCC achieves superior performance against ECC and EBCC in terms of all the three evaluation metrics with either base classifier. These experimental results can be regarded as a support to the main motivation of EDCC that it is necessary to consider multi-class nature of class variables when inducing chaining-based MDC models.
- BR does not consider any class dependencies which would impact its generalization performance. It is shown that EDCC achieves superior performance against BR in terms of all the three evaluation metrics with either base classifier. These experimental results validate the dependencies modeling strategy of EDCC.
- CP works by conducting powerset-like transformation in output space which can be regarded as optimizing *Exact Match*. ESC works by partitioning the class variables into groups to alleviate CP's overfitting on training set. Note that EDCC is not tailored to any specific evaluation metrics, while compared with CP and ESC, it is shown that EDCC also achieves comparable performance in terms of *Exact Match* and superior performance in terms of *Hamming Score* and *Sub-Exact Match*. These experimental results further validate the effectiveness of the special consideration strategy in EDCC.
- The gMML approach solves the MDC problem by alternatingly learning regression models (one per class label) as well as a Mahalanobis distance metric in the OvR decomposed label space, while EDCC solves the MDC problem by learning a chain of binary classifiers in the OvO decomposed label space. EDCC achieves superior performance against gMML in terms of all the three evaluation metrics, which suggests that the OvO decomposed label space might be more reasonable for MDC.
- The M3MDC approach also transforms the MDC problem into a number of binary classification problems via

[13]More details about the parameter setting can be found in the README file of LIBLINEAR [10].

[14]Due to high computational complexity, the experimental results of M3MDC are not available over 9 out of 20 data sets.

[15]The test results between EDCC and M3MDC are obtained based on the eleven data sets whose experimental results are available for M3MDC.



(a) Oes97 (b) Song (c) BeLaE (d) Rf1 (e) Scm1d

(f) CoIL2000 (g) Flickr (h) Fera (i) Adult

Fig. 4. Performance of EDCC (with base classifier LR) changes as the ensemble size (i.e, the number of base learners) increases from 1 to 10.

OvO decomposition w.r.t. each class space. M3MDC and EDCC solve the resulting binary classification problems in a joint manner via covariance regularization and learning a chain of classifiers, respectively. The superiority of EDCC over M3MDC suggests that chaining-based technique is a better alternative than covariance regularization in considering the relationships among a number of related binary classification problems.

- By comparing the experimental results in Table III and Table IV, it is shown that different base classifiers would affect the performance of EDCC. However, as shown in Table V, similar statistical test results are obtained with two different base classifiers. In other words, the superiority of EDCC against other baselines is not affected by the base classifier.

C. Further Analysis

In this subsection, some properties of the proposed approach will be further analyzed. Firstly, the improvement from DCC to EDCC (i.e., benefit brought by ensemble strategy) is presented. Next, the impact of different decomposition strategies and different ensemble strategies are investigated, respectively. Then, the computational complexity is analyzed. After that, the performance of EDCC with some recent multi-label/multi-task methods is compared. Finally, ablation study for the proposed approach is further conducted.

1) Improvement from DCC to EDCC: To alleviate the effect of the chaining order, it is recommended to construct ensembles of DCC (i.e., EDCC) whose performance has been reported and analyzed in Subsection IV-B. Here, the improvement from DCC to EDCC (i.e., benefit brought by ensemble strategy) is further analyzed. Specifically, the curve is illustrated on how the performance of EDCC changes in terms of each metric as the ensemble size (i.e., the number of base learners) increases from 1 to 10. To alleviate the effect of model's randomness, the experiments are repeated ten times for each ensemble size and the curve is drawn according to the mean values. Moreover, a vertical error bar at each data point is also drawn to show the corresponding standard deviation.

Fig.4 shows the corresponding curves over nine data sets, whose characteristics are very diversified, e.g., the number of data samples ranges from 334 to 18419, the number of features ranges from 10 to 1536, and the number of dimensions ranges from 3 to 16. It is shown that ensemble strategy will bring performance improvement for DCC. Specifically, when the ensemble size is greater than 5, though performance improvement still exists, the rate of performance improvement has become relatively slow. Therefore, the ensemble size of EDCC can be set to 5 for efficiency issue though greater ensemble size will mean slightly better generalization performance. In previous subsections, the ensemble size is fixed as 10 for EDCC just because this paper follows the common setting of the two ensemble-based approaches ECC [23] and ESC [21] and aims to conduct fair comparison.

2) Impact of Different Decomposition Strategies: To transform the MDC problem into a number of binary classification problems, DCC decomposes the MDC training set into $T = \sum_{j=1}^{q} \binom{K_j}{2}$ binary classification data sets via OvO rule w.r.t. each dimension. This subsection aims at analyzing the impact of different decomposition strategies, where the performance of EDCC is further compared with its two variants denoted as DeOvR and DeECOC. Specifically, DeOvR and DeECOC respectively replace the OvO decomposition with OvR and ECOC [8] for transforming multi-class classification into binary classification. The detailed experimental results are reported in Table VI and the *Wilcoxon signed-ranks test* [6] results are summarized in Table VII. It is shown that EDCC achieves statistically superior performance against DeOvR and DeECOC in terms of all evaluation metrics with either base classifier. In fact, some empirical studies [11] have shown that OvO usually serves as the better alternative than OvR and ECOC in decomposing multi-class classification into binary classification. These experimental results further support this conclusion and show the superiority of technical choice obtained by DCC.

3) Impact of Different Ensemble Strategies: For EDCC, the standard Bagging algorithm[16] is almost used to make ensembles of 10 base DCC classifiers in previous sections. Specifically, each DCC classifier is trained over a modified training set $\mathcal{D}_r$ $(1 \leq r \leq 10)$ which is generated by uniformly sampling the original training set $\mathcal{D}$ with replacement, and the

[16]See Figure 3.1 (pp.49) in [35] for more details about Bagging.

TABLE VI
EXPERIMENTAL RESULTS (MEAN ± STD.) OF EDCC AND TWO DECOMPOSITION VARIANTS WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE.

(a) Base classifier: Logistic regression

Data Set	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
	EDCC	DeOvR	DeECOC	EDCC	DeOvR	DeECOC	EDCC	DeOvR	DeECOC
Oes97	**.746±.022**	.741±.021	.737±.019	**.057±.048**	**.057±.048**	**.057±.048**	.123±.076	.123±.065	**.129±.060**
Jura	**.618±.056**	.582±.053	.606±.064	**.387±.104**	.337±.087	.362±.108	**.850±.044**	.827±.049	**.850±.037**
Oes10	.797±.018	**.803±.015**	.802±.014	**.099±.045**	.094±.038	.094±.042	.201±.064	.213±.065	**.218±.071**
Song	**.793±.028**	.786±.026	.789±.028	**.494±.054**	.484±.057	.489±.063	**.885±.044**	.880±.041	.882±.035
WQplants	**.660±.015**	.653±.016	.655±.016	**.094±.036**	**.094±.034**	.092±.033	**.286±.046**	.278±.044	.280±.053
WQanimals	**.636±.012**	.631±.015	.634±.015	**.064±.017**	**.064±.020**	.060±.017	.234±.032	.230±.032	**.238±.032**
WQ	**.648±.012**	.643±.014	.646±.012	**.007±.008**	.006±.008	.005±.008	**.048±.025**	.047±.023	.045±.018
BeLaE	**.449±.016**	.371±.018	.432±.015	**.033±.013**	.013±.004	.029±.010	**.158±.023**	.085±.020	.135±.019
Voice	**.950±.009**	.933±.010	.943±.008	**.902±.018**	.872±.017	.890±.016	**.997±.003**	.995±.005	.996±.004
Scm20d	**.690±.005**	.673±.004	.677±.006	**.083±.008**	.073±.010	.079±.009	**.160±.007**	.142±.010	.149±.009
Rf1	**.904±.004**	.888±.003	.897±.003	**.485±.017**	.420±.017	.454±.019	**.804±.009**	.776±.007	.799±.008
Thyroid	**.965±.002**	**.965±.002**	**.965±.002**	.770±.015	**.775±.015**	**.775±.014**	**.983±.004**	.982±.004	.982±.004
Pain	**.954±.003**	**.954±.003**	**.954±.003**	**.760±.015**	.758±.016	.759±.015	**.866±.010**	.864±.010	**.866±.008**
Scm1d	**.833±.003**	.815±.004	.823±.003	**.197±.011**	.181±.011	.192±.013	**.372±.016**	.332±.010	.352±.013
CoIL2000	**.938±.003**	.930±.005	.932±.003	**.747±.016**	.712±.019	.722±.013	**.947±.005**	.941±.009	.943±.006
Flickr	**.800±.004**	**.800±.004**	.798±.005	**.331±.016**	**.331±.013**	.326±.015	**.737±.012**	.734±.008	.732±.009
Disfa	**.907±.002**	.899±.002	.904±.002	**.417±.009**	.396±.010	.409±.008	**.677±.009**	.641±.010	.666±.012
Fera	**.653±.006**	.602±.007	.652±.008	**.215±.013**	.196±.013	**.215±.014**	**.460±.012**	.391±.010	.454±.011
Adult	**.723±.003**	.690±.006	.722±.004	**.281±.009**	.218±.010	.280±.008	**.687±.007**	.643±.012	.686±.010
Default	**.671±.004**	.651±.004	.667±.004	**.185±.009**	.156±.007	.181±.006	**.604±.008**	.564±.009	.593±.009

(b) Base classifier: Support vector machine

Data Set	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
	EDCC	DeOvR	DeECOC	EDCC	DeOvR	DeECOC	EDCC	DeOvR	DeECOC
Oes97	**.743±.024**	.732±.025	.736±.020	**.054±.044**	.030±.032	.042±.040	**.117±.086**	.096±.049	.099±.071
Jura	**.607±.075**	.597±.063	.602±.066	**.365±.117**	.356±.100	.348±.096	.850±.053	.838±.041	**.855±.057**
Oes10	.800±.016	.796±.016	**.801±.013**	.092±.047	**.102±.054**	.094±.052	**.213±.066**	.208±.073	.201±.063
Song	**.789±.027**	.786±.023	.788±.023	.475±.058	**.479±.045**	.478±.057	**.896±.049**	.883±.035	.888±.035
WQplants	**.658±.013**	.651±.013	.654±.016	**.096±.035**	**.096±.034**	.094±.034	**.296±.049**	.284±.049	.281±.044
WQanimals	**.635±.012**	.629±.013	.633±.014	**.067±.020**	.059±.022	.060±.023	**.241±.029**	.225±.035	.238±.027
WQ	**.646±.012**	.642±.012	.642±.013	**.008±.007**	.006±.008	.006±.008	**.056±.028**	.047±.023	.048±.022
BeLaE	**.439±.017**	.169±.019	.434±.017	.026±.011	.002±.003	**.027±.011**	**.149±.036**	.015±.009	.133±.018
Voice	**.946±.007**	.935±.009	.938±.009	**.893±.014**	.875±.016	.879±.016	**.998±.003**	.996±.004	.997±.003
Scm20d	**.689±.007**	.652±.008	.668±.007	**.096±.008**	.081±.008	.091±.009	**.170±.008**	.143±.008	.162±.010
Rf1	**.912±.003**	.887±.003	.902±.004	**.505±.011**	.413±.016	.467±.018	**.833±.008**	.773±.012	.810±.011
Thyroid	**.965±.002**	**.965±.002**	**.965±.002**	**.775±.013**	.771±.013	.773±.014	.981±.003	**.982±.004**	.981±.004
Pain	**.954±.003**	.952±.004	.953±.004	**.762±.015**	.758±.016	.761±.016	**.866±.008**	.857±.010	.861±.011
Scm1d	**.837±.002**	.804±.004	.828±.004	**.205±.012**	.179±.014	.198±.014	**.388±.012**	.325±.015	.363±.015
CoIL2000	.950±.003	.942±.003	**.960±.004**	.793±.010	.762±.016	**.833±.014**	.961±.006	.954±.006	**.969±.007**
Flickr	**.795±.004**	**.795±.005**	**.795±.004**	**.323±.014**	.322±.015	.322±.016	.725±.011	**.731±.013**	.725±.008
Disfa	**.907±.002**	.895±.003	.903±.002	**.424±.010**	.391±.012	.408±.010	**.671±.010**	.622±.012	.659±.007
Fera	**.645±.007**	.555±.009	.639±.009	**.215±.013**	.186±.012	.208±.012	**.450±.009**	.336±.013	.438±.017
Adult	**.712±.004**	.664±.005	.708±.004	**.260±.008**	.153±.009	.232±.007	.666±.011	.610±.012	**.678±.009**
Default	**.665±.003**	.653±.003	**.665±.003**	**.177±.006**	.156±.006	**.177±.006**	.591±.008	.571±.007	**.592±.009**

TABLE VII
WILCOXON SIGNED-RANKS TEST FOR EDCC AGAINST ITS TWO DECOMPOSITION VARIANTS.

Evaluation Metric		EDCC against DeOvR	EDCC against DeECOC
LR	HS	**win**[5.17e-04]	**win**[7.80e-04]
	EM	**win**[7.13e-04]	**win**[1.11e-03]
	SEM	**win**[4.49e-04]	**win**[1.69e-02]
SVM	HS	**win**[1.20e-04]	**win**[3.59e-03]
	EM	**win**[8.37e-04]	**win**[1.52e-02]
	SEM	**win**[1.89e-04]	**win**[7.19e-03]

predictions of base learners are aggregated via majority voting. The only minor difference with Bagging is that a random chaining order is also generated for each DCC classifier.

This subsection aims at analyzing the impact of different ensemble strategies, where the performance of EDCC is further compared with its two variants denoted as EnVar1 and EnVar2 respectively. Specifically, EnVar1 replaces the *sampling with replacement* step in EDCC with *sampling without replacement*, and EnVar2 replaces the *majority voting* step with *training a meta-learner* (i.e., stacking [35]). The detailed experimental results are reported in Table VIII and the *Wilcoxon signed-ranks test* [6] results are summarized in Table IX. It is shown that the performance of EDCC is indeed affected by the ensemble strategy and advanced ensemble techniques can be explored and integrated into EDCC in the future.

4) Computational Complexity: Table X summarizes the time complexity of EDCC and all compared approaches. Here, $\mathcal{F}(n_1, n_2, n_3)$ denotes the training complexity of some classification algorithm $\mathcal{L}$ over a n_3-class data set with n_1 training examples in n_2-dimensional feature space. $K = \max_{1 \leq j \leq N} K_j$ corresponds to the maximum number of class labels in each class space. For ESC [21], θ corresponds to the number of super-classes and K_θ corresponds to the maximum number of class labels in each super-class. For gMML [18], k corresponds to the number of nearest neighbors considered. For M3MDC [17], T_1 and T_2 correspond to the number of outer and inner iterations respectively. Note that EDCC trains binary classifiers while all compared approaches (except gMML and M3MDC) train multi-class classifiers.

Table XI reports the detailed time costs of EDCC and all compared approaches over each data set. It is shown that BR and ESC respectively spend the least and most time in

TABLE VIII
EXPERIMENTAL RESULTS (MEAN ± STD.) OF EDCC AND TWO ENSEMBLE VARIANTS WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE.

(a) Base classifier: Logistic regression

Data Set	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
	EDCC	EnVar1	EnVar2	EDCC	EnVar1	EnVar2	EDCC	EnVar1	EnVar2
Oes97	**.746±.022**	.742±.017	.741±.026	.057±.048	**.060±.051**	**.060±.053**	**.123±.076**	.114±.063	.111±.079
Jura	.618±.056	.609±.066	**.630±.044**	**.387±.104**	.362±.105	.384±.085	.850±.044	.855±.050	**.875±.054**
Oes10	.797±.018	.794±.018	**.804±.015**	**.099±.045**	**.099±.044**	**.099±.044**	.201±.064	.198±.067	**.218±.065**
Song	**.793±.028**	.789±.025	.790±.033	**.494±.054**	.479±.059	.493±.067	.885±.044	**.891±.040**	.879±.046
WQplants	**.660±.015**	.659±.015	.658±.016	.094±.036	.094±.034	**.097±.034**	.286±.046	.291±.043	**.296±.042**
WQanimals	**.636±.012**	.634±.013	.632±.016	.064±.017	**.065±.018**	.058±.018	**.234±.032**	**.234±.039**	.232±.043
WQ	**.648±.012**	.646±.012	.639±.012	**.007±.008**	.005±.008	.004±.008	**.048±.025**	**.048±.022**	.043±.020
BeLaE	.449±.016	**.452±.014**	.442±.017	**.033±.013**	.029±.013	.030±.011	**.158±.023**	**.158±.017**	.147±.029
Voice	.950±.009	.945±.008	**.955±.009**	.902±.018	.892±.016	**.913±.018**	.997±.003	.997±.003	**.998±.003**
Scm20d	.690±.005	.688±.006	**.699±.006**	**.083±.008**	**.083±.007**	**.083±.008**	.160±.007	.157±.009	**.162±.007**
Rf1	.904±.004	.899±.004	**.928±.003**	.485±.017	.473±.017	**.567±.014**	.804±.009	.789±.010	**.882±.008**
Thyroid	.965±.002	.964±.002	**.966±.002**	.770±.015	.765±.014	**.777±.015**	**.983±.004**	.982±.004	.982±.004
Pain	.954±.003	.954±.003	**.955±.003**	.760±.015	.758±.016	**.761±.012**	.866±.010	.864±.010	**.870±.008**
Scm1d	.833±.003	.831±.002	**.835±.003**	.197±.011	.198±.010	**.199±.007**	.372±.016	.371±.014	**.377±.012**
CoIL2000	.938±.003	.936±.004	**.939±.004**	.747±.016	.741±.016	**.753±.017**	.947±.005	.946±.006	**.949±.007**
Flickr	**.800±.004**	**.800±.005**	.798±.004	**.331±.016**	**.331±.015**	.328±.014	**.737±.012**	.734±.010	.734±.010
Disfa	.907±.002	.904±.002	**.910±.002**	.417±.009	.413±.009	**.426±.009**	.677±.009	.667±.011	**.686±.008**
Fera	.653±.006	.648±.006	**.658±.006**	.215±.013	.213±.012	**.224±.014**	.460±.012	.450±.010	**.468±.012**
Adult	**.723±.003**	**.723±.003**	**.723±.003**	**.281±.009**	.280±.007	.280±.007	.687±.007	**.688±.006**	**.688±.007**
Default	**.671±.004**	**.671±.004**	**.671±.004**	**.185±.009**	.184±.008	**.185±.008**	**.604±.008**	**.604±.007**	.603±.006

(b) Base classifier: Support vector machine

Data Set	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
	EDCC	EnVar1	EnVar2	EDCC	EnVar1	EnVar2	EDCC	EnVar1	EnVar2
Oes97	**.743±.024**	.737±.026	.736±.027	.054±.044	**.060±.049**	.039±.042	**.117±.086**	.114±.069	.099±.059
Jura	.607±.075	.600±.075	**.632±.062**	.365±.117	.359±.128	**.407±.120**	.850±.053	.841±.059	**.858±.058**
Oes10	.800±.016	.799±.017	**.801±.016**	.092±.047	.097±.043	**.102±.061**	**.213±.066**	**.213±.062**	.204±.068
Song	.789±.027	**.792±.022**	.768±.022	.475±.058	**.484±.055**	.452±.049	**.896±.049**	.894±.037	.860±.035
WQplants	**.658±.013**	.655±.015	.650±.013	**.096±.035**	.094±.036	.089±.029	**.296±.049**	.287±.047	.275±.037
WQanimals	**.635±.012**	.631±.013	.620±.013	**.067±.020**	.061±.021	.054±.027	**.241±.029**	.232±.035	.219±.026
WQ	**.646±.012**	.644±.013	.635±.013	**.008±.007**	.007±.008	**.008±.010**	**.056±.028**	.050±.026	.041±.020
BeLaE	.439±.017	**.450±.019**	.427±.012	.026±.011	**.028±.013**	.022±.009	.149±.036	**.158±.022**	.132±.025
Voice	.946±.007	.942±.009	**.948±.009**	.893±.014	.885±.017	**.899±.018**	**.998±.003**	**.998±.003**	.997±.003
Scm20d	**.689±.007**	.684±.006	.686±.007	**.096±.008**	.094±.008	.066±.008	**.170±.008**	**.170±.005**	.135±.010
Rf1	.912±.003	.908±.003	**.933±.002**	.505±.011	.491±.014	**.575±.010**	.833±.008	.822±.010	**.905±.011**
Thyroid	.965±.002	.965±.002	**.966±.002**	.775±.013	.772±.014	**.781±.011**	.981±.003	**.982±.004**	.979±.003
Pain	.954±.003	.953±.003	**.956±.004**	.762±.015	.761±.017	**.767±.017**	.866±.008	.863±.009	**.872±.009**
Scm1d	**.837±.002**	.834±.002	.826±.003	.205±.012	**.207±.012**	.178±.007	**.388±.012**	.379±.011	.350±.015
CoIL2000	.950±.003	.949±.003	**.951±.003**	.793±.010	.791±.013	**.799±.013**	**.961±.006**	.958±.006	.959±.007
Flickr	.795±.004	**.796±.006**	.789±.004	.323±.014	**.327±.016**	.315±.011	.725±.011	**.727±.017**	.717±.011
Disfa	.907±.002	.904±.002	**.911±.002**	.424±.010	.410±.010	**.440±.011**	.671±.010	.662±.009	**.691±.013**
Fera	.645±.007	.636±.007	**.646±.008**	.215±.013	.210±.013	**.217±.014**	.450±.009	.438±.010	**.453±.014**
Adult	**.712±.004**	.711±.004	.699±.008	**.260±.008**	.259±.006	.238±.021	**.666±.011**	.663±.011	.651±.019
Default	**.665±.003**	**.665±.003**	.650±.019	**.177±.006**	**.177±.006**	.155±.031	**.591±.008**	**.591±.009**	.564±.035

TABLE IX
WILCOXON SIGNED-RANKS TEST FOR EDCC AGAINST ITS TWO ENSEMBLE VARIANTS.

Evaluation Metric		EDCC against EnVar1	EDCC against EnVar2
LR	HS	**win**[7.80e-04]	**tie**[3.51e-01]
	EM	**win**[3.29e-03]	**tie**[1.61e-01]
	SEM	**tie**[1.13e-01]	**tie**[2.32e-01]
SVM	HS	**win**[1.00e-02]	**tie**[1.56e-01]
	EM	**tie**[2.79e-01]	**tie**[5.02e-01]
	SEM	**win**[5.68e-03]	**win**[3.33e-02]

TABLE X
THE TIME COMPLEXITY OF DCC AND ALL COMPARED APPROACHES.

Algo.	Time complexity
EDCC	$\mathcal{O}(q \cdot K^2 \cdot \mathcal{F}(m, d + q \cdot K^2, 2))$
ECC	$\mathcal{O}\left(q \cdot \mathcal{F}(m, d + q, K)\right)$
EBCC	$\mathcal{O}\left(q \cdot \mathcal{F}(m, d + q, K)\right)$
BR	$\mathcal{O}\left(q \cdot \mathcal{F}(m, d, K)\right)$
CP	$\mathcal{O}\left(\mathcal{F}(m, d, K^q)\right)$
ESC	$\mathcal{O}\left(q \cdot \mathcal{F}(m, d, K)\right) + \theta \cdot \mathcal{F}(m, d + \theta, K_\theta))$
SEEM	$\mathcal{O}(q^2 \cdot \mathcal{F}(m, d, K^2))$
gMML	$\mathcal{O}(d^3 + (\sum_j K_j)^3 + md^3 + md(\sum_j K_j) + mk)$
M3MDC	$\mathcal{O}(T_1 \cdot T_2 \cdot (\sum_{j=1}^{q} \binom{K_j}{2}) \cdot m^3 + T_1 \cdot (\sum_{j=1}^{q} \binom{K_j}{2})^3)$

most cases among EDCC, ECC, EBCC, BR, CP, ESC and SEEM whose implementations necessitate a base classifier. Generally, EDCC is more efficient than ESC in most cases and than CP with base classifier *logistic regression* in some cases where the number of dimensions is large (e.g., `Scm1d`, `Scm20d`, `Disfa` and `Fera`). Besides, both the proposed approach and M3MDC can be regarded as solving a set of binary classification problems obtained by OvO decomposition in a joint manner. As shown in Table XI, the time costs of EDCC are much smaller than M3MDC which suggests that chaining-based strategy is an efficient alternative to covariance regularization used by M3MDC.

5) Alternative Experimental Results: The task of MDC is similar to some related learning frameworks including multi-label classification (MLC) and multi-task learning (MTL), where their approaches can also be adapted to solve the MDC problems. In this subsection, some comparative studies are further conducted between the proposed approach and recently proposed multi-label/multi-task approaches. Specifically, for MLC, the WRAP approach is employed which solves the MLC

TABLE XI

THE TIME COSTS (UNIT: SECOND) OF EDCC AND ALL COMPARED APPROACHES OVER EACH DATA SET. NOTE THAT FOR THE LAST 11 DATA SET (FROM SCM20D TO DEFAULT), THE EXPERIMENTS OF M3MDC ARE EXECUTED ON A COMPUTING SERVER DUE TO ITS HIGH COMPUTATIONAL COMPLEXITY, AND ONLY THE EXPERIMENTAL RESULTS OVER THYROID AND ADULT ARE RETURNED WITHIN ABOUT 20 DAYS AND 10 DAYS (ABBREVIATED AS '20d' AND '10d' RESPECTIVELY) WHILE EXPERIMENTAL RESULTS OVER OTHER NINE DATA SETS ARE UNAVAILABLE WITHIN REASONABLE TIME.

(a) Base classifier: Logistic regression

Algo.	Oes97	Jura	Oes10	Song	WQp	WQa	WQ	BeLaE	Voice	Scm20	Rf1	Thy	Pain	Scm1d	CoIL	Flickr	Disfa	Fera	Adult	Default
EDCC	22	1	30	5	9	9	27	29	4	675	236	94	549	2714	1182	4745	1820	1192	247	208
ECC	25	1	34	4	4	4	11	16	3	464	200	38	315	2149	317	4254	935	610	51	100
EBCC	35	1	48	2	3	3	12	8	1	641	149	26	305	3339	157	2012	1086	301	21	40
BR	2	1	3	1	1	1	1	2	1	34	17	3	28	227	30	392	81	54	4	9
CP	12	1	18	1	5	6	9	42	1	2084	165	4	254	10056	636	4205	2576	2560	65	78
ESC	42	9	55	13	16	18	32	29	11	4440	579	48	1184	32258	1034	9302	5815	14269	351	111
SEEM	37	1	50	1	3	3	12	14	1	987	178	18	417	5459	312	2474	1790	588	32	51
gMML	3	2	4	3	6	6	9	11	11	105	44	44	48	118	69	86	115	83	96	110
M3MDC	1236	69	1615	194	2811	3275	11790	18932	3883	N/A	N/A	20d	N/A	N/A	N/A	N/A	N/A	N/A	10d	N/A

(b) Base classifier: Support vector machine

Algo.	Oes97	Jura	Oes10	Song	WQp	WQa	WQ	BeLaE	Voice	Scm20	Rf1	Thy	Pain	Scm1d	CoIL	Flickr	Disfa	Fera	Adult	Default
EDCC	37	1	49	7	23	25	113	57	3	1154	240	114	1127	3097	1313	5021	3953	3180	394	254
ECC	22	1	28	7	5	5	15	39	3	530	222	43	738	2675	491	4912	2524	2141	38	99
EBCC	28	1	34	3	3	3	13	18	1	795	173	30	747	5561	281	2489	2868	1055	15	40
BR	2	1	2	1	1	1	1	6	1	37	19	4	73	204	35	435	212	180	3	9
CP	2	1	3	1	2	2	3	14	1	386	41	4	193	2447	139	900	1209	1346	17	17
ESC	87	9	104	18	25	27	67	92	10	2884	315	54	1434	20111	482	6978	8729	7901	145	153
SEEM	16	1	19	2	2	2	8	29	1	914	148	20	926	5433	218	1981	3337	1105	12	25
gMML	3	2	4	3	6	6	9	11	11	105	44	44	48	118	69	86	115	83	96	110
M3MDC	1236	69	1615	194	2811	3275	11790	18932	3883	N/A	N/A	20d	N/A	N/A	N/A	N/A	N/A	N/A	10d	N/A

TABLE XII

EXPERIMENTAL RESULTS (MEAN ± STD.) OF EDCC (BASE CLASSIFIER: LOGISTIC REGRESSION), THE MULTI-LABEL APPROACH WRAP AND THE MULTI-TASK APPROACH FMC WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE.

Data	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
Set	EDCC	WRAP	FMC	EDCC	WRAP	FMC	EDCC	WRAP	FMC
Oes97	**.746±.022**	.626±.034	.563±.027	**.057±.048**	.048±.043	.000±.000	**.123±.076**	.093±.062	.003±.009
Jura	**.618±.056**	.476±.059	.602±.067	.387±.104	.159±.077	**.390±.090**	**.850±.044**	.794±.062	.814±.068
Oes10	**.797±.018**	.676±.024	.617±.018	**.099±.045**	.077±.045	.000±.000	**.201±.064**	.127±.048	.015±.017
Song	**.793±.028**	.767±.030	.752±.034	**.494±.054**	.438±.063	.430±.066	**.885±.044**	.866±.038	.831±.046
WQplants	**.660±.015**	.647±.015	.648±.015	**.094±.036**	**.094±.029**	**.094±.032**	**.286±.046**	.282±.048	.282±.048
WQanimals	**.636±.012**	.628±.013	.627±.013	**.064±.017**	.056±.024	.056±.024	**.234±.032**	.226±.031	.225±.031
WQ	**.648±.012**	.638±.012	.637±.013	**.007±.008**	.006±.008	.006±.008	**.048±.025**	.045±.023	.045±.023
BeLaE	**.449±.016**	.378±.024	.363±.024	**.033±.013**	.023±.007	.022±.009	**.158±.023**	.111±.020	.099±.012
Voice	**.950±.009**	.774±.011	.803±.020	**.902±.018**	.613±.023	.633±.032	**.997±.003**	.935±.016	.972±.016
Scm20d	**.690±.005**	.570±.009	.491±.008	**.083±.008**	.060±.008	.000±.001	**.160±.007**	.104±.013	.003±.002
Rf1	**.904±.004**	.696±.006	.613±.008	**.485±.017**	.098±.010	.034±.008	**.804±.009**	.305±.015	.194±.013
Thyroid	**.965±.002**	.960±.003	.960±.003	**.770±.015**	.738±.017	.738±.017	**.983±.004**	.982±.005	.982±.005
Pain	**.954±.003**	.948±.004	.949±.004	**.760±.015**	.751±.017	.745±.017	**.866±.010**	.847±.010	.850±.011
Scm1d	**.833±.003**	.653±.010	.539±.006	**.197±.011**	.093±.011	.001±.002	**.372±.016**	.157±.011	.007±.003
CoIL2000	**.938±.003**	.860±.004	.754±.007	**.747±.016**	.459±.014	.224±.021	**.947±.005**	.859±.010	.637±.016
Flickr	**.800±.004**	.732±.005	.643±.005	**.331±.016**	.205±.012	.086±.007	**.737±.012**	.581±.018	.392±.014
Disfa	**.907±.002**	.878±.003	.876±.003	**.417±.009**	.374±.012	.377±.014	**.677±.009**	.574±.010	.570±.009
Fera	**.653±.006**	.579±.007	.545±.010	**.215±.013**	.193±.013	.175±.025	**.460±.012**	.366±.013	.330±.014
Adult	**.723±.003**	.671±.004	.570±.008	**.281±.009**	.153±.006	.068±.013	**.687±.007**	.616±.008	.481±.015
Default	**.671±.004**	.653±.005	.622±.026	**.185±.009**	.161±.008	.134±.019	**.604±.008**	.570±.009	.513±.045

problem via generating label-specific features and inducing classification model in a joint manner [29]. For MTL, the FMC approach is employed which solves the MTL problem via incorporating a flexible manifold constraint and a robust loss [34]. The parameters of both WRAP and FMC are tuned according to respective literatures. Moreover, to make the two approaches applicable for MDC, one-hot conversion is conducted w.r.t. each class space which results in a binary label space with $\sum_{j=1}^{q} K_j$ labels. For unseen instance, the class label with the maximum modeling output in each class space is returned as the final prediction. The detailed experimental results are reported in Table XII and the *Wilcoxon signed-ranks test* [6] results are summarized in Table XIII. It is shown that EDCC achieves statistically superior performance against WRAP and FMC in terms of all evaluation metrics. These experimental results not only further validate the effectiveness of the proposed approach in this paper, but also demonstrate that it is better to specially design MDC approaches rather than directly adapting some approaches from other fields.

TABLE XIII

WILCOXON SIGNED-RANKS TEST FOR EDCC AGAINST WRAP AND FMC.

Evaluation	EDCC against	
Metric	WRAP	FMC
HS	**win**[8.86e-05]	**win**[8.86e-05]
EM	**win**[1.32e-04]	**win**[1.82e-04]
SEM	**win**[8.86e-05]	**win**[8.86e-05]

6) Ablation Study: In training phase, DCC employs the predicted labels to augment the feature space for both valid samples (with ground-truth labels '0/1') and invalid samples (without ground-truth labels). To analyze the rationality of this "train-and-predict" strategy, the performance of DCC is further

TABLE XIV
EXPERIMENTAL RESULTS (MEAN ± STD.) OF EDCC AND ITS TWO ABLATED VARIANTS WHERE THE BEST PERFORMANCE IS SHOWN IN BOLDFACE (BASE CLASSIFIER: LOGISTIC REGRESSION).

Data	*Hamming Score*			*Exact Match*			*Sub-Exact Match*		
Set	EDCC	AbVar1	AbVar2	EDCC	AbVar1	AbVar2	EDCC	AbVar1	AbVar2
Oes97	**.746±.022**	.737±.024	.736±.026	.057±.048	**.060±.051**	.057±.048	**.123±.076**	.105±.059	.111±.056
Jura	**.618±.056**	.609±.060	.598±.068	**.387±.104**	.371±.098	.348±.109	**.850±.044**	.847±.051	.847±.049
Oes10	**.797±.018**	.795±.013	.792±.015	**.099±.045**	.094±.042	.094±.038	**.201±.064**	**.201±.068**	.196±.050
Song	**.793±.028**	.790±.028	.788±.023	**.494±.054**	**.494±.058**	.478±.049	.885±.044	.877±.038	**.889±.037**
WQplants	**.660±.015**	.657±.017	.639±.017	.094±.036	**.095±.039**	.085±.033	.286±.046	**.288±.048**	.248±.040
WQanimals	**.636±.012**	.632±.012	.630±.016	.064±.017	**.067±.022**	.055±.019	**.234±.032**	.228±.025	.233±.035
WQ	**.648±.012**	.644±.012	.623±.009	**.007±.008**	.005±.008	.003±.005	.048±.025	**.050±.022**	.025±.011
BeLaE	**.449±.016**	.447±.019	.422±.019	.033±.013	**.036±.014**	.011±.006	**.158±.023**	.153±.028	.113±.018
Voice	.950±.009	.949±.009	**.951±.010**	.902±.018	.902±.017	**.904±.019**	**.997±.003**	**.997±.004**	**.997±.003**
Scm20d	**.690±.005**	.671±.007	.670±.006	.083±.008	**.099±.009**	.068±.008	.160±.007	**.172±.010**	.134±.006
Rf1	**.904±.004**	.892±.002	.890±.004	**.485±.017**	.469±.012	.446±.015	**.804±.009**	.777±.005	.765±.014
Thyroid	**.965±.002**	**.965±.002**	**.965±.002**	**.770±.015**	**.770±.015**	**.770±.015**	**.983±.004**	**.983±.003**	**.983±.004**
Pain	**.954±.003**	.953±.003	.953±.003	**.760±.015**	.759±.016	.758±.016	**.866±.010**	.860±.010	.860±.012
Scm1d	**.833±.003**	.819±.002	.828±.003	.197±.011	**.200±.012**	.180±.010	**.372±.016**	.358±.014	.352±.015
CoIL2000	**.938±.003**	**.938±.004**	.937±.003	.747±.016	**.748±.016**	.745±.014	.947±.005	**.948±.006**	.947±.007
Flickr	**.800±.004**	**.800±.004**	.791±.004	.331±.016	**.333±.015**	.312±.015	.737±.012	**.738±.010**	.715±.010
Disfa	**.907±.002**	.903±.002	.897±.002	**.417±.009**	.416±.010	.395±.011	**.677±.009**	.663±.009	.646±.010
Fera	**.653±.006**	.643±.007	.623±.005	**.215±.013**	.214±.013	.194±.012	**.460±.012**	.446±.011	.408±.011
Adult	**.723±.003**	.717±.003	.707±.005	.281±.009	**.305±.010**	.257±.012	**.687±.007**	.662±.007	.662±.008
Default	**.671±.004**	**.671±.003**	.583±.003	.185±.009	**.186±.007**	.060±.004	**.604±.008**	**.604±.006**	.439±.010

TABLE XV
WILCOXON SIGNED-RANKS TEST FOR EDCC AGAINST ITS TWO ABLATED VARIANTS (BASE CLASSIFIER: LOGISTIC REGRESSION).

Evaluation	EDCC against	
Metric	AbVar1	AbVar2
HS	**win**[3.90e-04]	**win**[1.40e-04]
EM	**tie**[4.21e-01]	**win**[1.82e-04]
SEM	**win**[1.76e-02]	**win**[4.63e-04]

compared with its two ablated variants denoted as AbVar1 and AbVar2 respectively in this subsection. Specifically, AbVar1 employs ground-truth labels '0/1' instead of predicted binary labels for valid samples, and AbVar2 employs soft predictions instead of hard prediction for invalid samples. Here, logistic regression which supports the soft prediction is employed as the base classifier. The detailed experimental results are reported in Table XIV and the *Wilcoxon signed-ranks test* [6] results are summarized in Table XV.

For AbVar1, it is shown that EDCC achieves statistically superior performance against it in terms of *Hamming Score* and *Sub-Exact Match*, and comparable performance against it in terms of *Exact Match*. These experimental results can be regarded as a support to the conjecture that the augmented feature space might be more consistent with *i.i.d.* assumption if predicted labels are used for both valid samples and invalid samples. For AbVar2, it is shown that EDCC achieves statistically superior performance against it in terms of all three evaluation metrics. These experimental results can be regarded as a support to the conjecture that hard labels might be more robust to noise than soft labels.

V. CONCLUSION

The classifier chains model is initially designed for multi-label classification, while the idea can be directly generalized to solve the MDC problem. In this paper, the DCC approach is proposed which builds a chain of binary classifiers with the help of OvO decomposition. The main motivation of DCC is that the characteristics of MDC's output space are different from MLC where the type of class variables corresponds to multi-class in MDC and binary-class in MLC. To the best of our knowledge, DCC serves as the first chaining-based MDC model which considers the unique characteristics of MDC. Moreover, the ensemble version of DCC is also constructed to alleviate the effect of chaining orders. Experiments clearly validate the effectiveness of the proposed approach.

In addition to some aforementioned issues in previous sections (e.g., using an additional class to indicate 'N/A', exploring deep learning model, introducing advanced ensemble techniques), there are still some other issues deserved to be further explored in the future. For example, the computational complexity is slightly higher than some compared approaches, it is better to design special base classifiers which can facilitate the training process of DCC. For another example, different random chaining orders are simply used to construct the ensemble model in EDCC, and it would be beneficial if some better chaining orders can be identified by exploiting the class dependencies.

REFERENCES

[1] A. H. Al Muktadir, T. Miyazawa, P. Martinez-Julia, H. Harai, and V. P. Kafle, "Multi-target classification based automatic virtual resource allocation scheme," *IEICE Transactions on Information and Systems*, vol. 102, no. 5, pp. 898–909, 2019.

[2] I. Batal, C. Hong, and M. Hauskrecht, "An efficient probabilistic framework for multi-dimensional classification," in *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, San Francisco, CA, USA, 2013, pp. 2417–2422.

[3] C. Bielza, G. Li, and P. Larrañaga, "Multi-dimensional classification with Bayesian networks," *International Journal of Approximate Reasoning*, vol. 52, no. 6, pp. 705–727, 2011.

[4] H. Borchani, C. Bielza, C. Toro, and P. Larrañaga, "Predicting human immunodeficiency virus inhibitors using multi-dimensional Bayesian network classifiers," *Artificial Intelligence in Medicine*, vol. 57, no. 3, pp. 219–229, 2013.

[5] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[6] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[7] L. Deng, X. He, and J. Gao, "Deep stacking networks for information retrieval," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, 2013, pp. 3153–3157.

[8] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1995.

[9] K.-B. Duan and S. S. Keerthi, "Which is the best multiclass SVM method? An empirical study," in *Proceedings of the International Workshop on Multiple Classifier Systems*, Seaside, CA, USA, 2005, pp. 278–285.

[10] R. Fan, K. Chang, C. Hsieh, X. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008, software available at https://www.csie.ntu.edu.tw/~cjlin/liblinear/.

[11] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes," *Pattern Recognition*, vol. 44, no. 8, pp. 1761–1776, 2011.

[12] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Computing Surveys*, vol. 47, no. 3, 2015, Article 52.

[13] S. Gil-Begue, C. Bielza, and P. Larrañaga, "Multi-dimensional Bayesian network classifiers: A survey," *Artificial Intelligence Review*, vol. 54, pp. 519–559, 2021.

[14] B.-B. Jia and M.-L. Zhang, "Maximum margin multi-dimensional classification," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, New York, NY, USA, 2020, pp. 4312–4319.

[15] B.-B. Jia and M.-L. Zhang, "Multi-dimensional classification via *k*NN feature augmentation," *Pattern Recognition*, vol. 106, 2020, Article 107423.

[16] B.-B. Jia and M.-L. Zhang, "Multi-dimensional classification via stacked dependency exploitation," *Science China Information Sciences*, vol. 63, no. 12, 2020, Article 222102.

[17] B.-B. Jia and M.-L. Zhang, "Maximum margin multi-dimensional classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2021, doi: 10.1109/TNNLS.2021.3084373.

[18] Z. Ma and S. Chen, "Multi-dimensional classification via a metric approach," *Neurocomputing*, vol. 275, pp. 1121–1131, 2018.

[19] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recognition*, vol. 45, no. 9, pp. 3084–3104, 2012.

[20] J. Read, L. Martino, and D. Luengo, "Efficient monte carlo methods for multi-dimensional learning with classifier chains," *Pattern Recognition*, vol. 47, no. 3, pp. 1535–1546, 2014.

[21] J. Read, C. Bielza, and P. Larrañaga, "Multi-dimensional classification with super-classes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 7, pp. 1720–1733, 2014.

[22] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Bled, Slovenia, 2009, pp. 254–269.

[23] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, no. 3, pp. 333–359, 2011.

[24] F. Serafino, G. Pio, M. Ceci, and D. Malerba, "Hierarchical multi-dimensional classification of web documents with multiwebclass," in *Proceedings of the 18th International Conference on Discovery Science*, Banff, AB, Canada, 2015, pp. 236–250.

[25] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random *k*-labelsets for multi-label classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079–1089, 2011.

[26] L. C. van der Gaag and P. R. de Waal, "Multi-dimensional Bayesian network classifiers," in *Proceedings of the 3rd European Workshop on Probabilistic Graphical Models*, Prague, Czech Republic, 2006, pp. 107–114.

[27] S. P. Verma, O. A. Uscanga-Junco, and L. Díaz-González, "A statistically coherent robust multidimensional classification scheme for water," *Science of the Total Environment*, vol. 750, 2021, Article 141704.

[28] D. Xu, Y. Shi, I. W. Tsang, Y. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 37, no. 7, pp. 2409–2429, 2020.

[29] Z.-B. Yu and M.-L. Zhang, "Multi-label classification with label-specific feature generation: A wrapped approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021, in press.

[30] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, and P. Larrañaga, "Bayesian chain classifiers for multidimensional classification," in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Barcelona, Spain, 2011, pp. 2192–2197.

[31] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, and X. Geng, "Binary relevance for multi-label learning: An overview," *Frontiers of Computer Science*, vol. 12, no. 2, pp. 191–202, 2018.

[32] M.-L. Zhang and Z.-H. Zhou, "Multi-label neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, 2006.

[33] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1819–1837, 2014.

[34] R. Zhang, H. Zhang, and X. Li, "Robust multi-task learning with flexible manifold constraint," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021, in press.

[35] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: Chapman & Hall/CRC, 2012.

[36] M. Zhu, S. Liu, and J. Jiang, "A hybrid method for learning multi-dimensional Bayesian network classifiers based on an optimization model," *Applied Intelligence*, vol. 44, no. 1, pp. 123–148, 2016.