Multi-Dimensional Classification via kNN Feature Augmentation

Bin-Bin Jia[a,b,c], Min-Ling Zhang[a,c,d,*]

[a] *School of Computer Science and Engineering, Southeast University, Nanjing 210096, China*
[b] *College of Electrical and Information Engineering, Lanzhou University of Technology, Lanzhou 730050, China*
[c] *Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China*
[d] *Collaborative Innovation Center of Wireless Communications Technology, China*

Abstract

In multi-dimensional classification (MDC), each training example is represented by a single instance (feature vector) while associated with multiple class variables, each of which specifies its class membership w.r.t. one specific class space. Most existing MDC approaches try to model dependencies among class variables in output space when inducing predictive functions, while the potential usefulness of manipulating feature space hasn't been investigated. As a first attempt towards feature manipulation in input space for MDC, a simple yet effective approach named KRAM is proposed which enriches the original feature space with augmented features based on kNN techniques. Specifically, simple counting statistics on the class membership of neighboring MDC examples as well as distance information between MDC examples and their k nearest neighbors are used to generate augmented feature vector. In this way, discriminative information from class space is expected to be brought into the feature space which would be helpful to the following MDC predictive model induction. To validate the effectiveness of the proposed feature augmentation techniques, comprehensive comparative studies are conducted over fifteen benchmark data sets. Compared to the original feature space, it is clearly shown that the kNN-

*Corresponding author
Email address: zhangml@seu.edu.cn (Min-Ling Zhang)

augmented features generated by the proposed KRAM approach can significantly improve generalization abilities of existing MDC approaches.
Keywords: machine learning, multi-dimensional classification, feature augmentation, class dependencies

1. Introduction

Multi-dimensional classification (MDC) aims to build learning models for real-world objects with a wealth of semantics, which assumes several class spaces to characterize the semantics of objects along different dimensions. Here, each example in MDC training set is represented by a single instance while associated with a number of class variables, and all these class variables respectively specify their class membership with regard to one specific class space. Specifically, there are many scenarios where we need to learn from MDC examples. For example, in image classification, the semantics of a natural scene image can be classified along the `landscape` dimension (with possible class labels *lake*, *grassland*, *mountain*, etc.), along the `time` dimension (with possible class labels *morning*, *afternoon*, *evening*, etc.), and along the `weather` dimension (with possible class labels *sunny*, *rainy*, *snowy*, etc.). For another example, in music classification, the semantics of a piece of song can be classified along the `language` dimension (with possible class labels *Chinese*, *English*, *Spanish*, etc.), along the `genre` dimension (with possible class labels *classical*, *popular*, *rock*, etc.), and along the `instrument` dimension (with possible class labels *guitar*, *violin*, *piano*, etc.). More applications of MDC techniques also include text classification [1, 2], bioinformatics [3, 4], web mining [5], etc.

Formally speaking, let $\mathcal{X} = \mathbb{R}^d$ be the d-dimensional input (feature) space and $\mathcal{Y} = C_1 \times C_2 \times \cdots \times C_q$ be the output space which corresponds to the Cartesian product of q heterogeneous class spaces. Here, each class space C_j $(1 \leq j \leq q)$ consists of K_j possible class labels, i.e., $C_j = \{c_1^j, c_2^j, \ldots, c_{K_j}^j\}$. Furthermore, let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\}$ be the MDC training set with m training examples, where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^\top \in \mathcal{X}$ is a d-dimensional

feature vector and $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ is the associated class vector, each of which is one possible value in j-th class space, i.e., $y_{ij} \in C_j$. Then, the learning task of multi-dimensional classification is to induce a mapping function $f : \mathcal{X} \mapsto \mathcal{Y}$ from $\mathcal{D}$ which can predict a proper class vector $f(\boldsymbol{x}) \in \mathcal{Y}$ for the unseen instance $\boldsymbol{x}$.

To accomplish MDC learning tasks, there are two intuitive solutions. The first one is to train multiple multi-class classifiers independently, one per class space. The second one is to train a single multi-class classifier by treating each distinct class combination in training set as a new class label, i.e., powerset method. These two intuitive strategies are simple and sometimes effective, but both of them also have obvious disadvantages. As all class spaces share the same feature space, there must be some relationships among them. However, due to the limited training examples, not all possible combination of class variables will appear in training set. The first intuitive strategy above ignores all dependencies among class spaces which would impact the generalization abilities of the induced MDC model, while the second strategy cannot make prediction for class combinations not appearing in the training set and also usually has prohibitively large number of combinations. In other words, both ignoring and overfitting class dependencies should be avoided. Therefore, most existing MDC approaches try to model dependencies among class variables from different dimensions in the output space in different ways, such as capturing dependencies between each pair of class spaces [6], specifying chaining order over class spaces [7, 8], assuming directed acyclic graph (DAG) structure over class spaces [9, 10], and partitioning class spaces into groups [11], etc.

Different from most existing works which directly model dependencies among different class variables in the output space, in this paper, we show the potential usefulness of manipulating input (feature) space for inducing MDC predictive models. Accordingly, a simple yet effective MDC approach named KRAM, i.e., *kNN featuRe Augmentation for Multi-dimensional classification*, is proposed.

Specifically, the main contributions of this paper correspond to:[1]

- We propose a first attempt towards manipulating feature space for MDC, where the proposed KRAM approach works by enriching the original features of MDC examples with their kNN-augmented features. In this way, discriminative information from output space can be brought into the input (feature) space to facilitate MDC predictive model induction.

- We design two versions of kNN-augmented features, i.e., discrete version and continuous version. For the discrete version, it is based on standard kNN techniques which employs simple counting statistics according to the class membership of k nearest neighbors in training set. For the continous version, it is based on weighted kNN techniques which combines the counting statistics with extra bias terms by considering the distance between current instance and its nearest neighbors.

- We conduct comprehensive comparative studies over fifteen benchmark data sets to validate the effectiveness of KRAM in improving the generalization performance of MDC approaches. Accordingly, properties of the proposed kNN augmented features have been analyzed based on the empirical results.

The rest of the paper is structured as follows. Section 2 briefly discusses existing works related to multi-dimensional classification. Section 3 presents technical details of the proposed KRAM approach. Section 4 reports comprehensive experimental results of comparative studies over a wide range of benchmark data sets. Finally, Section 5 concludes this paper.

[1]This paper is an extension of our preliminary work [12]. The main differences include: (1) The introduction and conclusion parts have been updated with further discussions on the proposed approaches; (2) Another strategy for generating kNN-augmented features (i.e. the continuous version) has been developed based on weighted kNN techniques; (3) More comprehensive comparative studies have been conducted in terms of newly added benchmark data sets, enriched parameter sensitivity analyses, and one recently proposed compared approach [13].

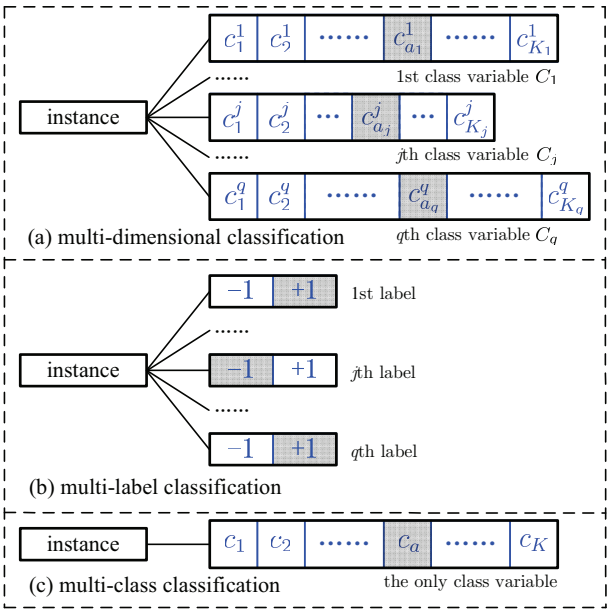


Figure 1: An intuitive comparison among multi-dimensional classification, multi-label classification, and multi-class classification.

2. Related Work

The most related learning frameworks to multi-dimensional classification include the traditional multi-class classification (MCC) and the popular multi-label classification (MLC) [14, 15, 16]. As shown in Figure 1, the MDC problem corresponds to several coupled MCC problems while the MLC problem corresponds to several coupled binary classification problems, i.e., both MDC and MLC can be regarded as one possible instantiation of multi-output learning [17] where each object is associated with multiple output variables. However, it is worth emphasizing that we should not simply consider that the difference between these two learning frameworks is whether the type of output variables is multi-class or binary class. Specifically, the key difference between MDC and MLC is whether semantic spaces are *heterogenous* or *homogeneous*, where each class variable in MDC corresponds to one specific class space, while each label in MLC specifies whether one concept is relevant or not in the only class space.

Therefore, it is unreasonable and will get suboptimal solutions to directly align class labels from different dimensions when trying to design MDC approaches. Anyway, the MLC problem can be regarded as a degenerated case of MDC by restricting all class variables to be binary-valued ones [11, 18, 19]. Additionally, recently proposed learning frameworks, such as duel set multi-label learning [20] and multiple ordinal output classification [21], can also be regarded as special cases of MDC when the number of dimensions equals two or class labels in each class space have ordinal relationship.

Intuitively, MDC can be solved by decomposing the original problem into a number of MCC problems, i.e., training a multi-class classifier independently according to each class variable. However, this natural strategy neglects the dependencies among class variables which may exist in real-world MDC tasks, and thus leads to suboptimal MDC model. Therefore, when trying to induce better MDC models, one of the core ways is to model dependencies among class spaces.

Dependencies between each pair of class spaces could be modeled by a set of base classifiers, and the final multi-dimensional inference is accomplished by combining predictive outputs from base classifiers via Markov random field [6]. Similar to classifier chains in MLC [22], the MDC problem can be converted into a chain of MCC problems, where the outputs of preceding multi-class classifiers in the chain are treated as extra input features when building subsequent ones. Obviously, its effectiveness is largely affected by the chaining order over class spaces which can be specified in deterministic manner [7] or random manner [8].

In addition, lots of existing works explicitly model dependencies among class variables with different families of directed acyclic graph (DAG) structures [9, 10], which form a family of probabilistic graphical models for MDC called multi-dimensional Bayesian network classifiers. Recent works focus on more efficient structure learning strategies which is still challenging [23, 24, 25]. Class powerset (CP) approach models class dependencies by converting the original MDC problem into only one MCC problem, where each distinct class combination existing in training set is regarded as a new class label in the new

MCC problem. Considering the possible huge number of class labels (with at most $\prod_{j=1}^{q} K_j$ class labels after powerset transformation), it will be helpful to group MDC class spaces into super-classes so as to facilitate the subsequent model induction for MDC [11].

3. The KRAM Approach

It has been widely acknowledged that modeling class dependencies in output space plays a crucial role when attempting to induce better MDC models, the importance of manipulating input (feature) space, however, hasn't been well investigated for MDC studies. In this section, we will present the technical details of our proposed KRAM approach, which aims at improving the generalization performance of existing MDC approaches via augmenting MDC examples' original features with the help of kNN techniques.

Following the same notations as used in previous sections, let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\}$ be the MDC training set where $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ is the corresponding class vector associated with $\boldsymbol{x}_i$. For each instance $\boldsymbol{x}$, let $\mathcal{N}(\boldsymbol{x}) = \{i_r \mid 1 \leq r \leq k\}$ be the set of indices for $\boldsymbol{x}$'s k nearest neighbors which are identified in $\mathcal{D}$. Here, Euclidean distance is utilized to measure the similarities between two instances, and for convenience, we assume that distance between $\boldsymbol{x}$ and $\boldsymbol{x}_{i_r}$ is in ascending order, which means that the smaller the value of r, the closer the distance between $\boldsymbol{x}$ and $\boldsymbol{x}_{i_r}$. Then, an indicating vector $\boldsymbol{v}_{ja}^{\boldsymbol{x}} = [v_{ja}^{\boldsymbol{x}}(1), v_{ja}^{\boldsymbol{x}}(2), \ldots, v_{ja}^{\boldsymbol{x}}(k)]^\top \in \{0,1\}^k$ is defined as follows:

$$v_{ja}^{\boldsymbol{x}}(r) = [\![y_{i_r j} = c_a^j]\!] \quad (1 \leq r \leq k,\ i_r \in \mathcal{N}(\boldsymbol{x})) \tag{1}$$

Here, $1 \leq a \leq K_j, 1 \leq j \leq q$. $\boldsymbol{y}_{i_r} = [y_{i_r 1}, y_{i_r 2}, \ldots, y_{i_r q}]^\top$ is the corresponding class vector associated with $\boldsymbol{x}_{i_r}$. The predicate $[\![\pi]\!]$ returns 1 if π holds and 0 otherwise. Therefore, $v_{ja}^{\boldsymbol{x}}(r)$ records whether the $\boldsymbol{x}$'s r-th nearest neighbor $\boldsymbol{x}_{i_r}$ has class label c_a^j in the j-th class space or not.

Based on $\boldsymbol{v}_{ja}^{\boldsymbol{x}}$, the following discrete version of statistics $\boldsymbol{\delta}_j^{\boldsymbol{x}} = [\delta_{j1}^{\boldsymbol{x}}, \delta_{j2}^{\boldsymbol{x}}, \ldots, \delta_{jK_j}^{\boldsymbol{x}}]^\top$ can be defined w.r.t. the j-th class space:

$$\delta_{ja}^{\boldsymbol{x}} = \langle \mathbf{1}_k, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle \quad (1 \leq a \leq K_j) \tag{2}$$

Here, $\mathbf{1}_k$ is a column vector of all ones with length k, and $\langle \cdot, \cdot \rangle$ returns inner product of two vectors. Therefore, $\delta_{ja}^{\boldsymbol{x}}$ records the number of examples in $\boldsymbol{x}$'s k nearest neighbors whose class label equals c_a^j in the j-th class space. According to the definition in Eq.(2), it is easy to know that $\sum_{a=1}^{K_j} \delta_{ja}^{\boldsymbol{x}} = k$ holds.

After traversing each class space one by one, a total of q different counting statistics $\boldsymbol{\delta}_j^{\boldsymbol{x}}$ $(1 \leq j \leq q)$, each of which contains K_j elements, can be generated. By concatenating all these q counting statistics, we can define the following augmented feature vector $\boldsymbol{\Delta}_{\boldsymbol{x}}$ for $\boldsymbol{x}$:

$$\boldsymbol{\Delta}_{\boldsymbol{x}} = \left[\boldsymbol{\delta}_1^{\boldsymbol{x}}, \boldsymbol{\delta}_2^{\boldsymbol{x}}, \ldots, \boldsymbol{\delta}_q^{\boldsymbol{x}}\right] \tag{3}$$

Based on the above feature vector, the original MDC training set $\mathcal{D}$ can be transformed into:

$$\widetilde{\mathcal{D}} = \{(\tilde{\boldsymbol{x}}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\} \tag{4}$$

Here, $\tilde{\boldsymbol{x}}_i = [\boldsymbol{x}_i, \boldsymbol{\Delta}_{\boldsymbol{x}_i}] \in \widetilde{\mathcal{X}}$ which means $\tilde{\boldsymbol{x}}_i$ is comprised of a concatenation of $\boldsymbol{x}_i$ and $\boldsymbol{\Delta}_{\boldsymbol{x}_i}$. $\widetilde{\mathcal{X}}$ represents a synthetic feature space which corresponds to the Cartesian product between the original feature space (i.e., $\mathcal{X}$) and a $(\sum_{j=1}^{q} K_j)$-dimensional augmented one. Thereafter, an MDC predictive model $f : \widetilde{\mathcal{X}} \mapsto \mathcal{Y}$ can be trained over the new constructed data set $\widetilde{\mathcal{D}}$ by applying any off-the-shelf MDC training algorithm $\mathcal{L}$, i.e., $f \leftarrowtail \mathcal{L}(\widetilde{\mathcal{D}})$. Given an unseen instance $\boldsymbol{x}^*$, its predicted class vector $\boldsymbol{y}^*$ can be assigned by feeding its corresponding augmented instance $\tilde{\boldsymbol{x}}^*$ into f.

Obviously, discrete version of statistics in Eq.(2) utilizes standard kNN techniques. Standard kNN approach simply assigns a test instance the class of the majority of its k nearest neighbors, i.e., the class of test instance is voted uniformly by its k nearest neighbors. As a popular variation, weighted kNN predicts the class of a test instance via non-uniformly voting by its k nearest neighbors. The closer distance between test instance and neighbor is, the greater weight the neighbor has. Following the idea of weighted kNN, another continuous version of augmented feature vector is designed based on the discrete one defined in Eq.(1)$\sim$Eq.(3).

According to the motivation of weighted kNN, in this paper, the weight vector for k nearest neighbors is simply set as $\boldsymbol{w} = [1, 1/\sqrt{2}, \cdots, 1/\sqrt{k}]^\top$. Then, a bias $\zeta_{ja}^{\boldsymbol{x}}$ for $\delta_{ja}^{\boldsymbol{x}}$ in Eq.(2) is defined according to $\boldsymbol{v}_{ja}^{\boldsymbol{x}}$ in Eq.(1) as follows:

$$\zeta_{ja}^{\boldsymbol{x}} = \frac{\langle \boldsymbol{w}, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle - min(\boldsymbol{v}_{ja}^{\boldsymbol{x}})}{max(\boldsymbol{v}_{ja}^{\boldsymbol{x}}) - min(\boldsymbol{v}_{ja}^{\boldsymbol{x}})}(\zeta_{max} - \zeta_{min}) + \zeta_{min} \tag{5}$$

Here, $max(\boldsymbol{v}_{ja}^{\boldsymbol{x}}) = \sum_{r=1}^{\langle \mathbf{1}_k, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle} w(r)$, $min(\boldsymbol{v}_{ja}^{\boldsymbol{x}}) = \sum_{r=k-\langle \mathbf{1}_k, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle+1}^{k} w(r)$ represent the possible maximum and minimum of $\langle \boldsymbol{w}, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle$ respectively, where $w(r)$ denotes the r-th element of weight vector $\boldsymbol{w}$. ζ_{max} and ζ_{min} are two hyper-parameters, and $\zeta_{max} - \zeta_{min} < 1$ holds. In this paper, we set ζ_{max} as 0.5 and ζ_{min} as 0. We can easily know that $\zeta_{min} \leq \zeta_{ja}^{\boldsymbol{x}} \leq \zeta_{max}$.

Then, compared with the discrete version of statistics defined in Eq.(2), we can define another continuous version of statistics $\boldsymbol{\delta}_j^{\boldsymbol{x}} = [\delta_{j1}^{\boldsymbol{x}}, \delta_{j2}^{\boldsymbol{x}}, \ldots, \delta_{jK_j}^{\boldsymbol{x}}]^\top$ as follows:

$$\delta_{ja}^{\boldsymbol{x}} = \langle \mathbf{1}_k, \boldsymbol{v}_{ja}^{\boldsymbol{x}} \rangle + \zeta_{ja}^{\boldsymbol{x}} \quad (1 \leq a \leq K_j) \tag{6}$$

Based on Eq.(6), we can have the continuous version of $\boldsymbol{\Delta}_{\boldsymbol{x}}$ in Eq.(3) and then transformed data set $\widetilde{\mathcal{D}}$ in Eq.(4).

Algorithm 1 summarizes the complete procedure of the proposed KRAM approach. Firstly, the original feature vector of each training example is enriched by two different kNN-augmented features: for discrete type, simple counting statistics derived from neighboring MDC examples is used, and for continuous type, extra bias is added into the simple counting statistics, and then a transformed MDC training set is gradually constructed (steps 1-17). After that, a MDC model is trained over the transformed MDC training set $\widetilde{\mathcal{D}}$ (step 18). Finally, for unseen instance, its class vector can be predicted based on the original features combined with augmented features (steps 19-21). In the remaining parts of this paper, we denote the discrete version as KRAM_d and the continuous version as KRAM_c respectively.

As shown in Algorithm 1, the kNN-augmented features generation procedure (steps 1-17) and the MDC predictive model induction procedure (step 18) are detached. In other words, the proposed KRAM approach is actually a meta-

Algorithm 1 The proposed KRAM approach.

Input: $\mathcal{D}$: MDC training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq m\}$
k: number of nearest neighbors considered
$\mathfrak{L}$: MDC training algorithm
$\boldsymbol{x}^*$: unseen instance

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

1: $\widetilde{\mathcal{D}} = \varnothing$;
2: **for** $i = 1$ to m **do**
3: Identify k nearest neighbors of $\boldsymbol{x}_i$ in $\mathcal{D}$ and store their indices in $\mathcal{N}(\boldsymbol{x}_i)$;
4: **for** $j = 1$ to q **do**
5: **for** $a = 1$ to K_j **do**
6: **switch** `type` **do**
7: **case** `discrete`:
8: Calculate $\delta_{ja}^{\boldsymbol{x}_i}$ according to Eq.(2);
9: **case** `continuous`:
10: Calculate $\delta_{ja}^{\boldsymbol{x}_i}$ according to Eq.(6);
11: **end switch**
12: **end for**
13: Set $\boldsymbol{\delta}_j^{\boldsymbol{x}_i} = [\delta_{j1}^{\boldsymbol{x}_i}, \delta_{j2}^{\boldsymbol{x}_i}, \ldots, \delta_{jK_j}^{\boldsymbol{x}_i}]^\top$;
14: **end for**
15: Set $\boldsymbol{\Delta}_{\boldsymbol{x}_i} = [\boldsymbol{\delta}_1^{\boldsymbol{x}_i}, \boldsymbol{\delta}_2^{\boldsymbol{x}_i}, \ldots, \boldsymbol{\delta}_q^{\boldsymbol{x}_i}]$;
16: $\widetilde{\mathcal{D}} = \widetilde{\mathcal{D}} \cup (\widetilde{\boldsymbol{x}}_i, \boldsymbol{y}_i)$, where $\widetilde{\boldsymbol{x}}_i = [\boldsymbol{x}_i, \boldsymbol{\Delta}_{\boldsymbol{x}_i}]$;
17: **end for**
18: Train MDC model f over $\widetilde{\mathcal{D}}$, i.e., $f \leftarrowtail \mathcal{L}(\widetilde{\mathcal{D}})$;
19: Identify k nearest neighbors of $\boldsymbol{x}^*$ in $\mathcal{D}$ and store their indices in $\mathcal{N}(\boldsymbol{x}^*)$;
20: Augment $\boldsymbol{x}^*$ with $\boldsymbol{\Delta}_{\boldsymbol{x}^*}$ being calculated the same as training set, i.e., $\widetilde{\boldsymbol{x}}^* = [\boldsymbol{x}^*, \boldsymbol{\Delta}_{\boldsymbol{x}^*}]$;
21: Return $\boldsymbol{y}^* = f(\widetilde{\boldsymbol{x}}^*)$.

strategy for MDC model induction, where any existing MDC training algorithm (i.e., $\mathcal{L}$ in step 18) can be employed to instantiate KRAM. Besides, both the two versions of kNN-augmented features designed in this paper should only be considered as a first attempt towards feature manipulation techniques for MDC and are not meant to be the best possible implementation among other feasible choices in future.

Generally speaking, KRAM embodies two major merits: 1) *Simplicity*: As shown in Algorithm 1, the KRAM is very succinct and can be implemented

easily. Specifically, the most time-consuming operation of KRAM is the k nearest neighbors identification process which has been well studied in kNN researches. The number of augmented features' dimension equals $\sum_{j=1}^{q} K_j$ which is not large usually, so there will not be too much extra computation. 2) *Effectiveness*: experimental studies reported in Section 4 clearly validate the fact that KRAM can improve the generalization abilities of any off-the-shelf MDC approaches.

Table 1: The characteristics of the employed benchmark data sets.

Data Set	#Exam.	#Dim.	#Labels/Dim.	#Features†
Edm	154	2	3	$16n$
Flare1	323	3	3,4,2	$10x$
Song	785	3	3	$98n$
WQplants	1060	7	4	$16n$
WQanimals	1060	7	4	$16n$
WaterQuality	1060	14	4	$16n$
Voice	3136	2	4,2	$19n$
Thyroid	9172	7	5,5,3,2,4,4,3	$7n$, $20b$, $2x$
Flickr	12198	5	3,4,3,4,4	$1536n$
Music	591	6	2	$71n$
Enron	1677	10	2	$1001b$
Image	2000	5	2	$294n$
Scene	2407	6	2	$294n$
Yeast	2417	14	2	$103n$
Mediamill	41583	11	2	$120n$

† n, x and b denote numeric, nominal and binary features respectively.

4. Experiments

4.1. Experimental Setup

4.1.1. Benchmark data sets

To validate the effectiveness of our proposed KRAM approach in improving the predictive abilities of existing MDC approaches, a total of 15 data sets have been used for performance comparison. Table 1 summarizes the characteristics of all benchmark data sets, including *the number of examples* (#Exam.),

the number of class spaces (#Dim.), *the number of class labels per class space* (#Labels/Dim.),[2] and *the number of features* (#Features).

The first nine benchmark data sets in Table 1 are collected from different MDC tasks in real world:

- `Edm` aims at predicting control operations during electrical discharge machining process [26]. This data set includes totally 2 class spaces which correspond to two parameters of controlling gap and flow respectively.

- `Flare1` aims at predicting the number of times certain types of solar flare occurred within 24 hours period [27]. This data set includes totally 3 class spaces which correspond to common, moderate, and severe solar flares respectively.

- `Song` aims at predicting different characteristics of songs. This data set includes totally 3 class spaces which correspond to the scenarios, genre and emotion of one song respectively. Besides, all songs are collected and annotated by ourselves.

- `Water Quality` aims at predicting the amount of different species in Slovenian rivers [28]. This data set includes totally 14 class spaces which correspond to 7 plants and 7 animals species respectively. By focusing on 7 plants species or the 7 animals species, then we have data sets `WQplants` and `WQanimals` [29].

- `Voice` aims at predicting some characteristics of a piece of human voice.[3] This data set includes totally 2 class spaces which correspond to mean frequency range and speaker's gender respectively [20].

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

[3] http://www.primaryobjects.com/2016/06/22/identifying-the-gender-of-a-voice-using-machine-learning

- `Thyroid` aims at estimating types of thyroid problems based on personal information of patients [27]. This data set includes totally 7 class spaces which correspond to seven different diagnosis respectively.

- `Flickr` aims at predicting objects in MIRFLICKR25000 [30] which are re-annotated by ourselves according to MDC framework and just part of pictures are reserved. This data set includes totally 5 class spaces which correspond to sky, people, night, plant, indoor respectively.

The last six data sets in Table 1 are selected from multi-label classification tasks including audio classification: `Music` [11], text classification: `Enron`,[4] image classification: `Image` [31], `Scene` [32], gene functional analysis: `Yeast` [33], and video classification: `mediamill` [34]. For these data sets, each class variable is binary-valued which is widely known as *label*. [5]

To the best of our knowledge, this paper employs more real-world MDC data sets than most state-of-the-art works on multi-dimensional classification [9, 11, 13]. Moreover, as shown in Table 1, the characteristics of all benchmark data sets are very diversified, e.g., the number of examples ranges from 154 to 41583, the number of features ranges from 10 to 1536, and the ratio of sum of all the numbers of class labels in each class space to the number of features ranges from 0.01 to 3.50 (i.e., the ratio of the number of augmented features to the number of original features). Therefore, the reported experimental results in this paper are quite comprehensive which can validate the effectiveness of KRAM more thoroughly.

4.1.2. Evaluation metrics

Given the test data set $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leq i \leq p\}$ with p MDC examples, where $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]^\top \in \mathcal{Y}$ is the ground-truth class vector of $\boldsymbol{x}_i$. More-

[4] http://mulan.sourceforge.net/datasets-mlc.html

[5] For Enron, we just use 10 out of all 53 labels with most positive instances. For Mediamill, we just use 11 out of all 120 labels similar to Enron. And instances without relevant labels are removed.

over, let $f : \mathcal{X} \mapsto \mathcal{Y}$ denote the induced MDC predictive model which is to be evaluated, and let $\hat{\boldsymbol{y}}_i = f(\boldsymbol{x}_i) = [\hat{y}_{i1}, \hat{y}_{i2}, \ldots, \hat{y}_{iq}]^\top$ be the predicted class vector for $\boldsymbol{x}_i$ by function f. Then, $r^{(i)} = \sum_{j=1}^{q} [\![y_{ij} = \hat{y}_{ij}]\!]$ denotes the number of class labels correctly classified by f. Based on these notations, the definitions of the three metrics employed in this paper are given as follows:

- *Hamming Score*:

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

  The value of this metric can be regarded as the probability of that any class label of one test example is correctly predicted by the induced MDC model.

- *Exact Match*:

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

  The value of this metric can be regarded as the probability of that all q class labels of one test example are correctly predicted simultaneously by the induced MDC model. Generally, the value of *exact match* might be rather low when the number of class spaces is large.

- *Sub-Exact Match*:

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

  The value of this metric can be regarded as the probability of that at least $q-1$ class labels of one test example are correctly predicted simultaneously by the induced MDC model. Obviously, it is a relaxed version of *exact match*.

4.1.3. Compared approaches

As stated before, the proposed KRAM approach is a meta-strategy to induce MDC predictive models. This means that any off-the-shelf MDC approaches can be employed to instantiate KRAM to further improve their generalization

performance. Here, a total of four well-established MDC approaches [11] serve this purpose:

- *Binary Relevance* (BR): This approach solves the MDC problem by training a number of multi-class classifiers independently, one per class space. In other words, BR completely ignores class dependencies.

- *Ensembles of Classifier Chains* (ECC): This approach solves the MDC problem by training a chain of multi-class classifiers, one per class space. Specifically, the subsequent multi-class classifiers in the chain are built by augmenting the feature space with the predictions of preceding multi-class classifiers. In other words, ECC models class dependencies via assuming chaining structures over class spaces. Besides, different base models in an ensemble of classifier chains consider different random chaining orders.

- *Ensembles of Class Powerset* (ECP): This approach solves the MDC problem by training a single multi-class classifier. Specifically, each distinct class combination in training set is treated as a new class label. In other words, ECP models class dependencies via powerset transformation. Besides, different base models in an ensemble of class powerset classifiers are built over different sub-training sets which are randomly sampled from the original training set.

- *Ensembles of Super Class classifiers* (ESC): This approach solves the MDC problem by partitioning class spaces into super-classes, where the partition process is fulfilled according to conditional dependencies among class spaces. In other words, ESC models class dependencies via the new generated super-classes. Specifically, different base models in an ensemble of super-class classifiers are built over different sub-training sets which are randomly sampled from the original training set.

For each ensemble approach (i.e., ECC, ECP and ESC), its base MDC model is induced over a sub-training set which contains 67% examples sampled from the original one randomly, and a total of 10 base MDC models are employed

in this paper [11]. Moreover, majority voting strategy is used to combine the predictions of all base MDC models for each example.

As implementing each MDC approach also necessiate a base multi-class classifier, both support vector machine (SVM) and Naïve Bayes (NB) are investigated in this paper. Specifically, SVM is implemented by LIBSVM [35] where the type of kernel function is linear and the regularization parameter C is set to 1. NB takes the common implementation where Gaussian pdf is used for continuous features and frequency counting with Laplacian correction is used for discrete features. Besides, the only parameter k for KRAM, which denotes the number of nearest neighbors considered, is set to 8 when conducting comparative experiments.

Let KRAM-$\mathcal{A}$ be the instantiation of KRAM with $\mathcal{A}$, where $\mathcal{A}$ denotes one of the compared approaches, i.e., $\mathcal{A} \in$\{BR, ECC, ECP, ESC\}. To show whether KRAM could improve the generalization abilities of MDC approaches, our aim is to compare predictive performance of KRAM-$\mathcal{A}$ against $\mathcal{A}$. On each configuration (each approach in terms of each metric on each data set), ten-fold cross-validation is conducted where both the mean metric value and corresponding standard deviation are recorded for performance comparison.

4.2. Experimental Results

The detailed experimental results are reported in Tables 2 to 5 for all MDC approaches and their KRAM counterparts in terms of *hamming score*, *exact match*, and *sub-exact match* respectively. Based on the ten-fold cross-validation results of each data set, pairwise t-test at 0.05 significance level is further conducted to show whether the performance of each KRAM counterpart is significantly different to its corresponding MDC approach with different multi-class classifiers (SVM or NB). Accordingly, the resulting win/tie/loss counts are summarized in Table 6 in terms of each evaluation metric over the 15 benchmark data sets.

Based on the experimental results mentioned above, it is interesting to obtain the following observations:

Table 2: Predictive performance of BR and its two version KRAM counterparts (mean±std. deviation) in terms of all three evaluation metrics. In addition, •/○ indicates whether KRAM_d-BR or KRAM_c-BR is statistically superior/inferior to BR on each data set, and results in bold face of KRAM_d-BR or KRAM_c-BR indicate that current KRAM version is statistically superior to the other version on each data set (pairwise t-test at 0.05 significance level).

(a) Multi-class classifier: SVM

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	BR	KRAM_d-BR	KRAM_c-BR	BR	KRAM_d-BR	KRAM_c-BR	BR	KRAM_d-BR	KRAM_c-BR
Edm	.689±.070	.734±.083•	**.760±.095**•	.442±.125	.521±.141•	**.560±.150**•	.935±.061	.947±.076	.960±.072
Flare1	.922±.034	.922±.033	.921±.034	.821±.073	.818±.072	.814±.077	.947±.039	.951±.036	.951±.036
Song	.793±.023	.787±.023○	.786±.024	.479±.059	.476±.050	.476±.049	.903±.033	.888±.046	.885±.054
WQpla.	.657±.016	.664±.013	.667±.014•	.097±.033	.099±.034	**.104±.032**	.287±.055	.300±.042	.299±.045
WQani.	.630±.014	.635±.012•	.637±.012•	.058±.022	.063±.014	.065±.016	.229±.034	.232±.030	.233±.031
WQ	.644±.013	.646±.010	.648±.011	.007±.008	.008±.007	.008±.007	.051±.024	.053±.017	.058±.020
Voice	.964±.007	.957±.008○	.957±.007○	.929±.014	.915±.016○	.915±.014○	.999±.002	.999±.002	.998±.002
Thyroid	.965±.002	.969±.003•	.969±.003•	.773±.015	.800±.018•	.801±.016•	.982±.004	.983±.004	.983±.004
Flickr	.791±.005	.790±.006	.790±.006	.313±.014	.310±.018	.310±.020	.720±.014	.719±.014	.718±.015
Music	.808±.023	.818±.022•	.818±.023	.272±.075	.331±.082•	.321±.079•	.674±.067	.682±.054	.686±.061
Enron	.808±.010	.807±.010	.805±.010	.179±.036	.182±.031	.181±.030	.416±.034	.397±.041	.398±.032○
Image	.828±.010	.841±.011•	.842±.012•	.394±.028	.459±.033•	.464±.039•	.782±.031	.783±.027	.785±.029
Scene	.895±.009	.918±.008•	.918±.008•	.530±.035	.651±.038•	.649±.035•	.855±.018	.867±.020	.867±.018
Yeast	.801±.006	.811±.007•	.812±.006•	.151±.017	.199±.015•	.199±.015•	.269±.029	.307±.020•	.310±.011•
Mediamill	.830±.001	.850±.001•	**.851±.001**•	.148±.005	.239±.009•	**.247±.007**•	.411±.008	.500±.006•	**.507±.006**•

(b) Multi-class classifier: NB

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	BR	KRAM_d-BR	KRAM_c-BR	BR	KRAM_d-BR	KRAM_c-BR	BR	KRAM_d-BR	KRAM_c-BR
Edm	.677±.096	.680±.088	.693±.073	.432±.166	.445±.153	.444±.151	.922±.074	.916±.060	.941±.038
Flare1	.886±.061	**.872±.051**	.838±.059○	.774±.099	**.756±.095**	.722±.093○	.910±.066	**.895±.055**	.858±.068○
Song	.626±.038	.629±.034	.623±.033	.238±.054	.224±.050	.213±.050○	.678±.071	.695±.068	.689±.065
WQpla.	.397±.028	**.506±.033**•	.475±.037•	.001±.003	**.036±.026**•	.016±.018•	.018±.012	**.113±.040**•	.082±.035•
WQani.	.381±.021	**.419±.019**•	.400±.022•	.004±.009	**.008±.010**	.003±.006	.041±.016	.049±.019	.040±.020
WQ	.389±.017	**.488±.022**•	.443±.016•	.000±.000	.000±.000	.000±.000	.000±.000	.003±.005	.001±.003
Voice	.882±.008	.921±.008•	**.939±.010**•	.782±.015	.847±.016•	**.880±.020**•	.982±.006	.996±.003•	**.998±.003**•
Thyroid	.926±.005	**.925±.003**	.704±.030○	.580±.027	**.575±.015**	.047±.013○	.916±.011	**.912±.009**	.307±.053○
Flickr	.648±.007	.654±.007•	**.659±.007**•	.139±.011	.143±.011•	**.147±.011**•	.436±.013	.444±.012•	**.453±.014**•
Music	.743±.018	.761±.023•	.760±.027•	.206±.043	.218±.058	.208±.060	.552±.057	.591±.050•	.601±.057•
Enron	.551±.012	.576±.015•	.574±.014•	.025±.017	.027±.017	.029±.017	.109±.022	.119±.026	.114±.029
Image	.573±.016	.586±.018•	**.595±.016**•	.069±.016	.074±.021•	**.080±.019**•	.255±.028	.279±.034•	**.297±.035**•
Scene	.763±.009	.777±.009•	**.791±.011**•	.177±.023	.198±.022•	**.238±.034**•	.561±.021	.591±.026•	**.605±.024**•
Yeast	.699±.010	.695±.014	**.704±.013**•	.095±.018	.115±.018•	.115±.022•	.149±.020	.182±.027•	.175±.014•
Mediamill	.620±.001	.659±.002•	**.673±.001**•	.012±.001	.048±.003•	**.067±.005**•	.086±.005	.135±.005•	**.161±.007**•

Table 3: Predictive performance of ECC and its two version KRAM counterparts (mean±std. deviation) in terms of all three evaluation metrics. In addition, •/○ indicates whether KRAM_d-ECC or KRAM_c-ECC is statistically superior/inferior to ECC on each data set, and results in bold face of KRAM_d-ECC or KRAM_c-ECC indicate that current KRAM version is statistically superior to the other version on each data set (pairwise t-test at 0.05 significance level).

(a) Multi-class classifier: SVM

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ECC	KRAM_d-ECC	KRAM_c-ECC	ECC	KRAM_d-ECC	KRAM_c-ECC	ECC	KRAM_d-ECC	KRAM_c-ECC
Edm	.695±.065	.769±.087•	.756±.090•	.454±.123	.598±.169•	.585±.147•	.935±.069	.940±.058	.928±.066
Flare1	.922±.034	.922±.034	.922±.035	.817±.078	.818±.073	.818±.078	.951±.036	.951±.036	.951±.036
Song	.790±.024	.788±.026	.788±.027	.481±.057	.476±.051	.479±.048	.891±.036	.891±.047	.888±.051
WQpla.	.654±.016	.663±.014•	**.666±.014**•	.093±.037	.105±.037	.105±.038	.283±.049	.295±.044	.297±.043
WQani.	.630±.014	.637±.014•	.637±.012•	.061±.023	.064±.010	.067±.015	.229±.032	.241±.040	.238±.037
WQ	.643±.013	.644±.013	.646±.013	.006±.008	.009±.006	.009±.004	.050±.023	.048±.018	.050±.023
Voice	.961±.008	.953±.009○	.953±.009○	.923±.016	.908±.017○	.908±.019	.998±.002	.998±.003	.998±.003
Thyroid	.965±.002	.969±.003•	**.969±.003**•	.772±.014	.800±.016•	.802±.017•	.981±.004	.982±.004	.983±.004
Flickr	.797±.004	.797±.005	.797±.005	.328±.013	.325±.014	.325±.017	.730±.015	.732±.014	.732±.015
Music	.814±.025	.810±.022	.813±.021	.346±.079	.343±.078	.352±.086	.676±.064	.677±.051	.676±.053
Enron	.824±.010	**.822±.011**	.819±.012○	.215±.030	.203±.037	.199±.027○	.461±.037	.455±.037	.443±.040
Image	.831±.012	.844±.012•	.846±.012•	.479±.033	.522±.036•	.525±.036•	.730±.033	.745±.031	.749±.036
Scene	.905±.011	.921±.008•	.921±.007•	.649±.035	.708±.026•	.710±.025•	.796±.030	.825±.020•	.825±.018•
Yeast	.797±.007	.808±.007•	.808±.008•	.207±.014	.252±.014•	.253±.018•	.288±.023	.316±.022•	.319±.025•
Mediamill	.830±.002	.849±.001•	**.851±.001**•	.188±.008	.267±.006•	**.276±.006**•	.434±.006	.513±.005•	**.521±.005**•

(b) Multi-class classifier: NB

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ECC	KRAM_d-ECC	KRAM_c-ECC	ECC	KRAM_d-ECC	KRAM_c-ECC	ECC	KRAM_d-ECC	KRAM_c-ECC
Edm	.690±.084	.674±.097	.689±.070	.451±.145	.438±.162	.444±.148	.929±.064	.909±.062	.935±.044
Flare1	.883±.059	**.875±.053**	.838±.062○	.774±.087	**.771±.088**	.737±.095	.904±.073	**.889±.060**	.852±.064○
Song	.621±.036	**.623±.034**	.613±.037	.228±.036	**.219±.043**	.191±.047○	.671±.068	.683±.066	.680±.069
WQpla.	.353±.033	**.494±.038**•	.444±.038•	.001±.003	**.035±.018**•	.024±.019•	.013±.010	**.123±.037**•	.073±.030•
WQani.	.377±.024	**.416±.020**•	.395±.021•	.007±.008	.006±.007	.004±.007	.039±.016	.049±.015•	.042±.018
WQ	.360±.020	**.487±.021**•	.431±.018•	.000±.000	.000±.000	.000±.000	.000±.000	.001±.003	.000±.000
Voice	.880±.009	.921±.008•	**.939±.010**•	.780±.015	.847±.015•	**.879±.020**•	.980±.007	.996±.003•	**.998±.003**•
Thyroid	.926±.007	**.929±.004**	.758±.016○	.593±.026	**.592±.022**	.058±.013○	.906±.020	**.922±.008**•	.392±.063○
Flickr	.649±.007	.655±.007•	**.660±.007**•	.140±.012	.143±.011•	**.147±.011**•	.438±.014	.447±.014•	**.455±.015**•
Music	.745±.020	.761±.023•	.763±.026•	.230±.058	.221±.065	.221±.064	.557±.051	.603±.048•	.608±.049•
Enron	.551±.011	.573±.015•	.573±.013•	.027±.016	.029±.016	.029±.017	.105±.026	.115±.029	.115±.027•
Image	.576±.014	.587±.014•	**.596±.017**•	.069±.019	.074±.020•	**.078±.021**•	.261±.028	.283±.033•	**.296±.036**•
Scene	.767±.010	.780±.010•	**.793±.010**•	.181±.024	.200±.021•	**.238±.030**•	.569±.027	.595±.031•	**.612±.023**•
Yeast	.696±.009	.698±.013	**.704±.013**•	.102±.016	.125±.024•	.124±.022•	.163±.020	.193±.027•	.188±.016•
Mediamill	.628±.001	.667±.002•	**.679±.002**•	.013±.001	.050±.003•	**.069±.005**•	.095±.004	.142±.005•	**.167±.007**•

Table 4: Predictive performance of ECP and its two version KRAM counterparts (mean±std. deviation) in terms of all three evaluation metrics. In addition, •/◦ indicates whether KRAM_d-ECP or KRAM_c-ECP is statistically superior/inferior to ECP on each data set, and results in bold face of KRAM_d-ECP or KRAM_c-ECP indicate that current KRAM version is statistically superior to the other version on each data set (pairwise t-test at 0.05 significance level).

(a) Multi-class classifier: SVM

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ECP	KRAM_d-ECP	KRAM_c-ECP	ECP	KRAM_d-ECP	KRAM_c-ECP	ECP	KRAM_d-ECP	KRAM_c-ECP
Edm	.721±.082	.763±.107•	.766±.094•	.559±.136	.612±.170	.618±.140•	.883±.074	.915±.075	.915±.075
Flare1	.921±.036	.922±.034	.923±.035	.817±.078	.821±.073	.821±.073	.947±.039	.947±.039	.951±.042
Song	.786±.029	.781±.028	.781±.035	.484±.054	.467±.059	.470±.071	.878±.040	.877±.040	.874±.043
WQpla.	.647±.015	.585±.027◦	.586±.026◦	.093±.028	.067±.029◦	.065±.029◦	.281±.049	.187±.040◦	.191±.042◦
WQani.	.629±.013	.556±.014◦	.550±.015◦	.065±.018	.029±.011◦	.024±.012◦	.230±.032	.151±.030◦	.136±.022◦
WQ	.628±.015	.557±.010◦	.555±.012◦	.001±.003	.004±.005	.004±.005	.035±.018	.019±.016	.016±.015◦
Voice	.955±.013	.950±.010	.950±.010	.912±.025	.903±.020	.903±.019	.998±.003	.998±.003	.997±.004
Thyroid	.965±.002	.968±.002•	.969±.003•	.773±.014	.802±.015•	**.804±.015**•	.981±.005	.979±.003◦	.979±.003◦
Flickr	.772±.004	.760±.006◦	.761±.006◦	.297±.012	.281±.009◦	.281±.010◦	.680±.011	.658±.016◦	.658±.018◦
Music	.799±.032	.802±.025	.800±.030	.343±.076	.341±.073	.343±.083	.640±.064	.659±.066	.650±.072
Enron	.830±.008	.824±.009◦	.822±.010◦	.235±.029	.224±.026	.224±.033	.482±.021	.459±.043	.462±.034
Image	.832±.012	.842±.009•	.841±.011•	.513±.024	.540±.024•	.535±.029•	.710±.036	.727±.029•	.725±.032
Scene	.914±.009	.925±.008•	.925±.007•	.700±.029	.731±.029•	.729±.029•	.796±.028	.825±.018•	.825±.021•
Yeast	.795±.007	.795±.007	.794±.010	.252±.012	.262±.018	.264±.018•	.304±.020	.317±.018	.317±.022•
Mediamill	.818±.002	.841±.002•	**.843±.002**•	.218±.008	.286±.004•	**.293±.003**•	.430±.008	.506±.007•	**.513±.008**•

(b) Multi-class classifier: NB

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ECP	KRAM_d-ECP	KRAM_c-ECP	ECP	KRAM_d-ECP	KRAM_c-ECP	ECP	KRAM_d-ECP	KRAM_c-ECP
Edm	.731±.062	.722±.089	.699±.076	.554±.112	.548±.120	.528±.118	.909±.047	.896±.081	.870±.069
Flare1	.908±.045	**.903±.046**	.867±.048◦	.790±.081	**.777±.084**	.734±.094◦	.941±.057	**.938±.057**	.883±.059◦
Song	.674±.044	**.684±.042**	.674±.044	.311±.053	.317±.051	.308±.057	.733±.079	.749±.080	.733±.077
WQpla.	.607±.015	**.647±.019**•	.609±.028	.034±.021	**.067±.038**•	.043±.028	.175±.043	**.258±.056**•	.208±.049•
WQani.	.590±.020	**.625±.017**•	.598±.016	.020±.014	**.042±.016**•	.030±.011•	.143±.054	**.221±.049**•	.188±.031•
WQ	.599±.018	**.597±.018**	.562±.019◦	.008±.009	.004±.007◦	.007±.008	.032±.024	.033±.020	.027±.018
Voice	.903±.010	.927±.011•	**.937±.008**•	.811±.019	.857±.022•	**.877±.018**•	.995±.004	.997±.003	.997±.004
Thyroid	.966±.003	**.963±.003**◦	.948±.002◦	.793±.017	**.768±.015**◦	.678±.018◦	.974±.005	**.973±.005**	.962±.006◦
Flickr	.714±.007	.717±.008•	**.719±.007**•	.197±.006	.201±.006•	.200±.006	.563±.015	.568±.016•	**.571±.014**•
Music	.770±.029	.784±.019•	.789±.018•	.249±.078	.281±.073•	.277±.077•	.591±.071	.617±.053	.635±.036•
Enron	.777±.011	.784±.009•	.788±.009•	.166±.034	.177±.033	.176±.036	.324±.035	.343±.034•	**.358±.031**•
Image	.746±.012	.754±.011•	.759±.008•	.285±.022	.302±.022•	**.315±.020**•	.597±.034	.612±.033•	.622±.028•
Scene	.867±.011	.875±.013•	**.883±.012**•	.550±.030	.575±.040•	**.591±.034**•	.693±.031	.713±.036•	**.736±.033**•
Yeast	.773±.011	.787±.008•	.784±.009•	.203±.018	.240±.024•	.234±.022•	.258±.022	.293±.022•	.292±.029•
Mediamill	.656±.005	.689±.004•	**.692±.005**•	.031±.003	**.056±.003**•	.053±.004•	.097±.008	.147±.008•	.145±.008•

Table 5: Predictive performance of ESC and its two version KRAM counterparts (mean±std. deviation) in terms of all three evaluation metrics. In addition, •/◦ indicates whether KRAM$_d$-ESC or KRAM$_c$-ESC is statistically superior/inferior to ESC on each data set, and results in bold face of KRAM$_d$-ESC or KRAM$_c$-ESC indicate that current KRAM version is statistically superior to the other version on each data set (pairwise t-test at 0.05 significance level).

(a) Multi-class classifier: SVM

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC
Edm	.701±.079	.751±.102•	.766±.105•	.513±.122	.592±.165•	.624±.164•	.890±.076	.909±.070	.909±.070
Flare1	.923±.033	.923±.036	.923±.035	.821±.073	.821±.073	.824±.073	.951±.036	.951±.042	.947±.039
Song	.790±.030	.789±.029	.787±.029	.480±.067	.481±.058	.480±.057	.893±.038	.888±.047	.884±.047
WQpla.	.651±.016	.664±.015•	.665±.016•	.094±.038	.104±.039	.101±.037	.284±.050	.283±.049	**.299±.049**
WQani.	.630±.014	.636±.016	.633±.013	.062±.021	.065±.018	.058±.017	.232±.033	.239±.041	.230±.039
WQ	.641±.013	.638±.017	.635±.014	.006±.008	.012±.010	.009±.006	.046±.022	.050±.022	.048±.017
Voice	.961±.008	.953±.009◦	.953±.010◦	.924±.016	.908±.018◦	.908±.020◦	.998±.002	.998±.003	.998±.003
Thyroid	.965±.002	.969±.002•	.969±.003•	.771±.014	.801±.015•	.802±.016•	.982±.004	.981±.004	.981±.004
Flickr	.791±.004	.784±.005◦	.783±.004◦	.320±.011	.314±.011◦	.313±.012	.718±.009	.704±.013◦	.700±.010◦
Music	.809±.022	.810±.028	.810±.025	.330±.069	.352±.093	.335±.086	.669±.062	.671±.065	.676±.057
Enron	.833±.009	.829±.009	.830±.011	.224±.039	.216±.032	.211±.028	.485±.038	.470±.041	.479±.040
Image	.833±.008	.845±.013•	.843±.015•	.494±.025	.529±.034•	.528±.040•	.719±.028	.745±.036•	.738±.038•
Scene	.910±.013	.923±.008•	.924±.008•	.668±.045	.720±.030•	.722±.028•	.799±.032	.825±.022•	.827±.027•
Yeast	.800±.006	.807±.007•	.807±.007•	.236±.019	.258±.018•	.260±.022•	.309±.028	.320±.025•	.320±.024•
Mediamill	.824±.002	.845±.002•	**.848±.001•**	.207±.006	.277±.005•	**.285±.004•**	.434±.007	.508±.008•	**.517±.008•**

(b) Multi-class classifier: NB

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC	ESC	KRAM$_d$-ESC	KRAM$_c$-ESC
Edm	.674±.095	.674±.101	.696±.064	.432±.166	.438±.162	.450±.140	.915±.063	.909±.062	.941±.049
Flare1	.896±.059	**.892±.053**	.857±.058◦	.780±.093	**.768±.086**	.728±.094◦	.929±.064	**.926±.060**	.870±.069◦
Song	.646±.031	.666±.037•	.662±.037•	.274±.047	.304±.054•	.290±.063	.692±.066	.719±.067•	.722±.067•
WQpla.	.442±.034	**.549±.031•**	.487±.030•	.001±.003	**.040±.025•**	.017±.017•	.042±.019	**.133±.031•**	.082±.026•
WQani.	.577±.022	**.598±.013•**	.574±.022	.024±.018	.026±.023	.026±.019	.139±.050	.167±.045	.136±.043
WQ	.609±.017	**.609±.017**	.553±.023◦	.002±.004	.002±.004	.001±.003	.023±.013	.025±.017	.011±.012
Voice	.881±.012	.922±.008•	**.939±.010•**	.782±.020	.847±.015•	**.880±.020•**	.981±.008	.996±.003•	**.998±.003•**
Thyroid	.958±.004	**.952±.006◦**	.945±.003◦	.738±.022	**.703±.036◦**	.660±.015◦	.970±.007	**.966±.006**	.960±.007◦
Flickr	.703±.008	.705±.009	**.710±.010•**	.178±.013	.181±.014	.186±.016•	.533±.017	.537±.015	**.546±.020•**
Music	.738±.023	.764±.030•	.762±.026•	.210±.070	.242±.089•	.218±.063	.524±.039	.581±.082•	.603±.051•
Enron	.768±.016	.776±.018	.766±.022	.098±.029	.106±.039	.080±.045	.302±.049	**.337±.051**	.288±.078
Image	.593±.017	.608±.015•	**.618±.020•**	.069±.021	.074±.021	**.084±.021•**	.289±.032	.315±.029•	**.332±.035•**
Scene	.866±.010	.868±.013	**.878±.015•**	.541±.024	.528±.046	.525±.072	.703±.031	.733±.038•	**.793±.021•**
Yeast	.716±.006	**.743±.006•**	.731±.008•	.110±.014	**.154±.015•**	.135±.022•	.167±.019	**.217±.022•**	.195±.021•
Mediamill	.655±.003	.693±.008•	**.695±.009•**	.038±.004	.071±.005•	**.076±.004•**	.114±.006	.177±.009•	**.184±.007•**

Table 6: Win/tie/loss counts of pairwise t-test (at 0.05 significance level) between each MDC approach and its KRAM counterpart in terms of *hamming score* (HScore), *exact match* (EMatch), and *sub-exact match* (SEMatch).

	multi-class classifier: SVM			multi-class classifier: NB			
	HScore	EMatch	SEMatch	HScore	EMatch	SEMatch	**In Total**
KRAM$_{d}$-BR against BR	8/5/2	7/7/1	2/13/0	10/5/0	7/8/0	8/7/0	**42/45/3**
KRAM$_{d}$-ECC against ECC	8/6/1	6/8/1	3/12/0	10/5/0	7/8/0	10/5/0	**44/44/2**
KRAM$_{d}$-ECP against ECP	5/5/5	4/8/3	3/8/4	10/4/1	9/4/2	8/7/0	**39/36/15**
KRAM$_{d}$-ESC against ESC	7/6/2	6/7/2	4/10/1	8/6/1	6/8/1	8/7/0	**38/44/7**
KRAM$_{c}$-BR against BR	8/6/1	7/7/1	2/12/1	11/2/2	7/5/3	8/5/2	**43/37/10**
KRAM$_{c}$-ECC against ECC	8/5/2	6/8/1	3/12/0	11/2/2	7/6/2	9/4/2	**44/37/9**
KRAM$_{c}$-ECP against ECP	5/5/5	6/6/3	3/7/5	8/4/3	7/6/2	9/4/2	**38/32/20**
KRAM$_{c}$-ESC against ESC	7/6/2	6/8/1	4/10/1	9/3/3	6/7/2	9/4/2	**41/38/11**
In Total	**56/44/20**	**48/59/13**	**24/84/12**	**77/31/12**	**56/52/12**	**69/43/8**	**330/313/77**

- Among all the 720 configurations (15 data sets $\times$ 4 MDC approaches $\times$ 3 metrics $\times$ 2 multi-class classifiers $\times$ 2 versions of KRAM), the KRAM counterpart can achieve superior or at least comparable performance against its corresponding MDC approach in 643 configurations (about 89.3%).

- BR deals with MDC tasks by independent decomposition, where potential class dependencies are totally neglected in this approach. It is shown in Table 6 that KRAM-BR can achieve superior or at least comparable performance against BR in 167 out of 180 cases. This observation indicates that helpful discriminative information is indeed brought into feature space via the kNN-augmented features generated by KRAM. Specifically, these discriminative information can be regarded as an alternative way of dependency modeling when designing MDC approaches.

- Both ECC and ESC deal with MDC tasks by explicitly modeling class dependencies, which are fulfilled by specifying a chaining structure over class spaces or grouping class spaces into super-classes. It is worth noting that for these two MDC approaches which are designed under the inherent mechanism of dependency modeling, the kNN-augmented features generated by KRAM can also help improve their generalization performance

significantly.

- ECP deals with MDC tasks by modeling full-order class dependencies, where all distinct class combinations in training set have been treated as new classes in the learning phase. It is shown that the kNN-augmented features generated by KRAM can generally help improve ECP's generalization performance, while there are 35 cases where KRAM-ECP performs significantly inferior to ECP (the number of loss cases is a lot more than BR, ECC and ESC relatively). Most of these under-performing cases (29 out of 35) occur on data sets *Flickr*, *Thyroid* and *WaterQuality* (including its two divisions *WQplants* and *WQanimals*), in which the possible number of class combinations is high (e.g., 4^{14} for *WaterQuality*).

Table 7: Win/tie/loss counts of pairwise t-test (at 0.05 significance level) between KRAM$_{\mathrm{d}}$-$\mathcal{A}$ and KRAM$_{\mathrm{c}}$-$\mathcal{A}$ ($\mathcal{A} \in$\{BR, ECC, ECP, ESC\})in terms of *hamming score* (HScore), *exact match* (EMatch), and *sub-exact match* (SEMatch).

	multi-class classifier: SVM			multi-class classifier: NB			
	HScore	EMatch	SEMatch	HScore	EMatch	SEMatch	**In Total**
KRAM$_{\mathrm{d}}$-BR against KRAM$_{\mathrm{c}}$-BR	0/13/2	0/12/3	0/14/1	5/4/6	4/6/5	3/7/5	**12/56/22**
KRAM$_{\mathrm{d}}$-ECC against KRAM$_{\mathrm{c}}$-ECC	1/11/3	0/14/1	0/14/1	6/3/6	4/6/5	3/7/5	**14/55/21**
KRAM$_{\mathrm{d}}$-ECP against KRAM$_{\mathrm{c}}$-ECP	0/14/1	0/13/2	0/14/1	6/5/4	5/7/3	4/8/3	**15/61/14**
KRAM$_{\mathrm{d}}$-ESC against KRAM$_{\mathrm{c}}$-ESC	0/14/1	0/14/1	0/13/2	6/4/5	4/8/3	5/5/5	**15/58/17**
In Total	**1/52/7**	**0/53/7**	**0/55/5**	**23/16/21**	**17/27/16**	**15/27/18**	**56/230/74**

4.3. Further Analysis

4.3.1. Comparison between two KRAM *versions*

To show whether the performance between two versions of KRAM, i.e. KRAM$_{\mathrm{d}}$-$\mathcal{A}$ and KRAM$_{\mathrm{c}}$-$\mathcal{A}$, is significantly different, pairwise t-test based on ten-fold cross-validation (at 0.05 significance level) is also conducted, and we use bold face type for the statistically superior one in Tables 2 to 5. Table 7 summarizes the resulting win/tie/loss counts accordingly.

Based on the experimental results mentioned above, it is interesting to obtain the following observations:

- Across all the 360 configurations (15 data sets × 4 MDC approaches × 3 metrics × 2 multi-class classifiers), KRAM$_{\mathrm{d}}$ and KRAM$_{\mathrm{c}}$ tie 230 times (about 63.9%), KRAM$_{\mathrm{d}}$ wins 56 times, and KRAM$_{\mathrm{c}}$ wins 74 times. So, two versions of KRAM have comparable performance in general.

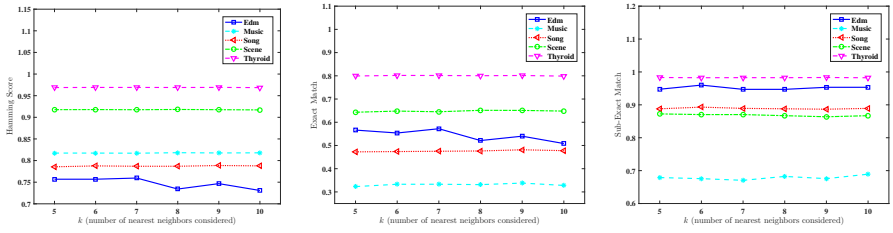
- When SVM is utilized as the multi-class classifier, performance of KRAM$_{\mathrm{c}}$ is relatively better. Specifically, the performance of KRAM$_{\mathrm{c}}$ is comparable with KRAM$_{\mathrm{d}}$ in most cases and superior in some cases with SVM (with only one loss on HScore with ECC approach).

- When NB is utilized as the multi-class classifier, KRAM$_{\mathrm{d}}$ achieves relatively more robust performance than KRAM$_{\mathrm{c}}$ does. Specifically, there are more win and loss cases with NB between KRAM$_{\mathrm{c}}$ and KRAM$_{\mathrm{d}}$. However, KRAM$_{\mathrm{c}}$ will achieve very poor performance in some configurations compared with KRAM$_{\mathrm{d}}$ (e.g., *Thyroid*) while KRAM$_{\mathrm{d}}$ won't even if it achieves significantly inferior performance to KRAM$_{\mathrm{c}}$. Possible reason is that NB assumes Gaussian pdf for continuous features, while KRAM$_{\mathrm{c}}$'s continuous augmented features will unfit this assumption sometimes. Obviously, the more imbalanced class distribution is, the more severe unfitness will be, then KRAM$_{\mathrm{c}}$ will achieve worse performance, and vise versa.

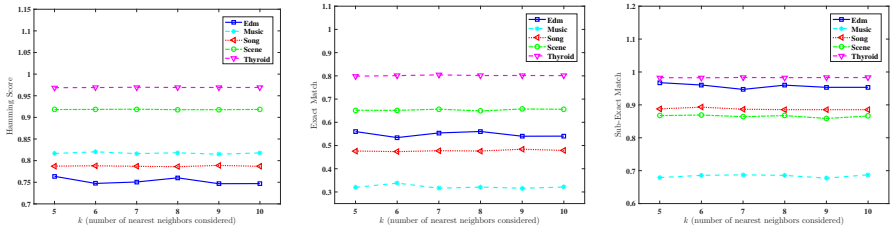
- In summary, KRAM$_{\mathrm{c}}$ should be a better choice when SVM is utilized as the multi-class classifier, while it depends on the characteristics of concrete data set and MDC approach when making a choice between KRAM$_{\mathrm{d}}$ and KRAM$_{\mathrm{c}}$ with NB as the multi-class classifier.

4.3.2. Parameter sensitivity analysis

As shown in Algorithm 1, there is only one parameter to be specified for KRAM, i.e., k, which is fixed to 8 in previous sections. To justify this parameter setting, we also investigate the sensitivity of KRAM w.r.t. the value of k. In terms of each evaluation metric, Figure 2 illustrates how the performance of KRAM (with MDC approach BR) changes as the value of k increases from 5

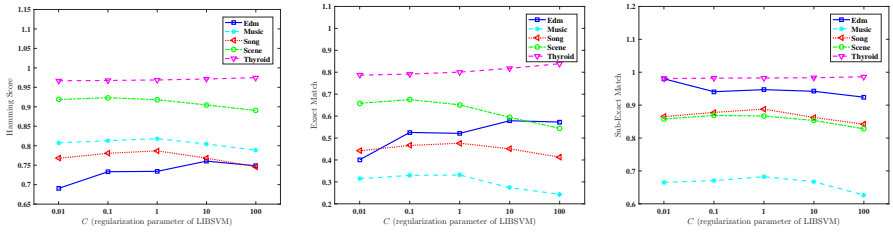


(a) HScore of KRAM$_{\mathrm{d}}$-BR (b) EMatch of KRAM$_{\mathrm{d}}$-BR (c) SEMatch of KRAM$_{\mathrm{d}}$-BR

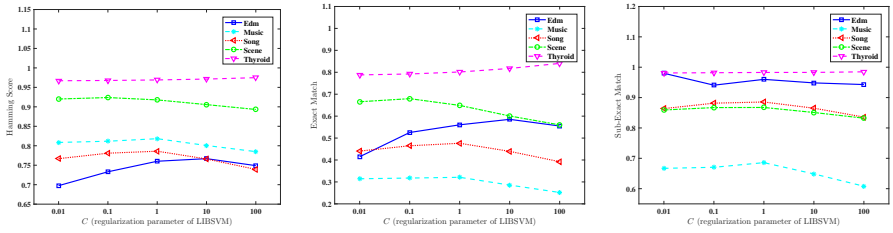


(d) HScore of KRAM$_{\mathrm{c}}$-BR (e) EMatch of KRAM$_{\mathrm{c}}$-BR (f) SEMatch of KRAM$_{\mathrm{c}}$-BR

Figure 2: Performance of KRAM-BR changes in terms of *hamming score* (HScore), *exact match* (EMatch), and *sub-exact match* (SEMatch) as k ranges from 5 to 10 on five data sets.



(a) HScore of KRAM$_{\mathrm{d}}$-BR (b) EMatch of KRAM$_{\mathrm{d}}$-BR (c) SEMatch of KRAM$_{\mathrm{d}}$-BR



(d) HScore of KRAM$_{\mathrm{c}}$-BR (e) EMatch of KRAM$_{\mathrm{c}}$-BR (f) SEMatch of KRAM$_{\mathrm{c}}$-BR

Figure 3: Performance of KRAM-BR changes in terms of *hamming score* (HScore), *exact match* (EMatch), and *sub-exact match* (SEMatch) as the LIBSVM regularization parameter C ranges in $\{0.01, 0.1, 1, 10, 100\}$ on five data sets.

to 10. It is shown that both two KRAM versions can achieve relatively stable performance when the value of k varies. Therefore, in this paper, the value of k is moderately set to 8 which is also the recommended parameter setting of KRAM. Moreover, parameter insensitivity serves as a desirable property which keeps KRAM away from sophisticated parameter-tuning issue for practical use.

In this paper, both SVM and NB are utilized as the multi-class classifier to implement each MDC approach. Here, we also investigate the sensitivity of KRAM w.r.t. the regularization parameter C of LIBSVM [35] (The common NB implementation doesn't involve specific parameters to be tuned). In terms of each evaluation metric, Figure 3 illustrates how the performance of KRAM (with MDC approach BR) changes as the regularization parameter C ranges in $\{0.01, 0.1, 1, 10, 100\}$. It is shown that both of the two KRAM versions can achieve relatively better performance when $C = 1$, which is also the default parameter setting of LIBSVM.

4.3.3. Alternative experimental analysis

The four selected compared approaches (i.e. BR, ECC, ECP, ESC) all necessiate a multi-class classifier for implementation, in this subsection, we also investigate the gMML approach [13] as compared approach which does not necessiate a base multi-class classifier. Specifically, gMML alternately learns linear regression models for each class label as well as a Mahalanobis distance metric to solve MDC problem effectively, where the Mahalanobis distance metric can make the distance between linear regression outputs of one example and its ground-truth class label vector closer. The detailed experimental results are reported in Table 8. It is shown that the kNN-augmented features generated by KRAM can also help improve the generalization performance of gMML and the performance of the KRAM_c version is relatively better.

5. Conclusion

Most existing works for multi-dimensional classification focus on modeling class dependencies in output space, while this paper extends our preliminary

Table 8: Predictive performance of gMML and its two version KRAM counterparts (mean±std. deviation) in terms of all three evaluation metrics. In addition, •/◦ indicates whether KRAM_d-gMML or KRAM_c-gMML (denoted as KRAM_d or KRAM_c for short) is statistically superior/inferior to gMML on each data set, and results in bold face of KRAM_d or KRAM_c indicate that current KRAM version is statistically superior to the other version on each data set (pairwise t-test at 0.05 significance level).

	Hamming Score			Exact Match			Sub-Exact Match		
Data Set	gMML	KRAM_d	KRAM_c	gMML	KRAM_d	KRAM_c	gMML	KRAM_d	KRAM_c
Edm	.714±.083	.770±.090•	.766±.082 •	.487±.145	.586±.188•	.579±.166 •	.941±.065	.954±.055	.954±.055
Flare1	.923±.033	.923±.035	.924±.034	.821±.073	.818±.075	.821±.073	.951±.036	.954±.039	.954±.039
Song	.788±.027	.787±.024	.786±.025	.484±.059	.481±.054	.481±.054	.883±.041	.883±.040	.882±.041
WQpla.	.655±.015	.663±.018	.662±.016	.092±.035	.098±.039	.096±.036	.286±.053	.295±.043	.290±.040
WQani.	.630±.015	.642±.014•	.642±.013 •	.062±.023	.062±.010	.062±.013	.227±.033	.240±.033	**.248±.033**
WQ	.643±.013	.649±.012	.649±.012	.006±.008	.008±.006	.008±.006	.049±.024	.054±.018	.057±.020
Voice	.842±.009	.945±.010•	.946±.010 •	.699±.017	.892±.020•	.893±.018 •	.985±.011	.998±.003•	.998±.003 •
Thyroid	.960±.003	.967±.003•	.967±.003 •	.741±.015	.790±.018•	.790±.100 •	.982±.005	.981±.004	.981±.004
Flickr	.779±.004	.782±.005•	.782±.006 •	.287±.009	.296±.011•	.295±.030 •	.690±.016	.696±.015•	.696±.016 •
Music	.801±.018	.815±.023•	.817±.027 •	.254±.057	.320±.095•	.333±.027 •	.652±.040	.681±.033•	.677±.049
Enron	.832±.009	.833±.011	.833±.011	.197±.027	.202±.038	**.206±.024**•	.477±.042	.476±.042	.478±.041
Image	.811±.010	.839±.010•	.840±.010 •	.289±.025	.448±.028•	.454±.019 •	.787±.027	.781±.021	.779±.024
Scene	.893±.009	.917±.008•	**.919±.007**•	.457±.046	.646±.025•	**.657±.011**•	.908±.017	.858±.023◦	.861±.018 ◦
Yeast	.800±.005	.811±.006•	.811±.006 •	.134±.018	.210±.020•	.211±.035 •	.266±.026	.311±.016•	.309±.018 •
Mediamill	.811±.001	.847±.001•	**.850±.001**•	.111±.006	.250±.005•	**.258±.005**•	.342±.007	.497±.006•	**.505±.007**•

work [12] which solves MDC problems by manipulating the input space. The major contribution of our research is to propose a feature augmentation strategy for multi-dimensional classification that augments feature space with augmented features generated via combining with class space, which suggests an alternative solution to induce MDC models.

To justify the proposed strategy, a simple yet effective approach named KRAM is designed based on kNN techniques, and comprehensive comparative studies have been conducted to validate its effectiveness accordingly. Experimental results show that: (a) Both versions of KRAM, i.e., discrete version KRAM_d and continuous version KRAM_c, can improve predictive performance of existing MDC approaches; (b) KRAM_d has more stable performance than KRAM_c when the base multi-class classifier is sensitive to the feature type (discrete or con-

tinuous), e.g., Naïve Bayes classifier. For other base multi-class classifier less sensitive to feature type, e.g., SVM, KRAM$_c$ is likely to be a better choice; (c) In light of the effectiveness of KRAM, feature augmentation technique can be further studied as an alternative strategy for modeling class dependencies.

As an initial attempt towards solving MDC problem with feature augmentation, there are several potential ways that the current KRAM instantiation can be improved: (a) In addition to the simple counting statistics derived from k nearest neighbors, more advanced information could be utilized for feature augmentation by trying to exploit available domain knowledge [36]; (b) Other than the meta-strategy for feature augmentation, customized feature augmentation techniques can be investigated for given MDC approaches; (c) Similar to the label-specific features techniques for multi-label classification [37, 38], it is worthwhile to investigate the feasibility of generating specific augmented features w.r.t. each class space.

References

[1] J. Ortigosa-Hernández, J. D. Rodriguez, L. Alzate, M. Lucania, I. Inza, J. A. Lozano, Approaching sentiment analysis by using semi-supervised learning of multi-dimensional classifiers, Neurocomputing 92 (2012) 98–115.

[2] C. Tu, Z. Liu, H. Luan, M. Sun, PRISM: Profession identification in social media, ACM Transactions on Intelligent Systems and Technology 8 (6) (2017) Article 81.

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

[4] H. Borchani, C. Bielza, C. Toro, P. Larrañaga, Predicting human immun-

odeficiency virus inhibitors using multi-dimensional Bayesian network classifiers, Artificial Intelligence in Medicine 57 (3) (2013) 219–229.

[5] F. Serafino, G. Pio, M. Ceci, D. Malerba, Hierarchical multidimensional classification of web documents with multiwebclass, in: Lecture Notes in Computer Science 9356, Springer, Berlin, 2015, pp. 236–250.

[6] J. Arias, J. A. Gamez, T. D. Nielsen, J. M. Puerta, A scalable pairwise class interaction framework for multidimensional classification, International Journal of Approximate Reasoning 68 (2016) 194–210.

[7] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, 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.

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

[9] C. Bielza, G. Li, P. Larrañaga, Multi-dimensional classification with Bayesian networks, International Journal of Approximate Reasoning 52 (6) (2011) 705–727.

[10] J. H. Bolt, L. C. van der Gaag, Balanced sensitivity functions for tuning multi-dimensional Bayesian network classifiers, International Journal of Approximate Reasoning 80 (2017) 361–376.

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

[12] B.-B. Jia, M.-L. Zhang, Multi-dimensional classification via kNN feature augmentation, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, HI, 2019, pp. 3975–3982.

[13] Z. Ma, S. Chen, Multi-dimensional classification via a metric approach, Neurocomputing 275 (2018) 1121–1131.

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

[15] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Computing Surveys 47 (3) (2015) Article 52.

[16] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, X. Geng, Binary relevance for multi-label learning: An overview, Frontiers of Computer Science 12 (2) (2018) 191–202.

[17] D. Xu, Y. Shi, I. W. Tsang, Y. Ong, C. Gong, X. Shen, Survey on multi-output learning, IEEE Transactions on Neural Networks and Learning Systems (2020) in press.

[18] B.-B. Jia, M.-L. Zhang, Maximum margin multi-dimensional classification, in: Proceedings of the 34th AAAI Conference on Artificial Intelligence, New York, NY, 2020, in press.

[19] B.-B. Jia, M.-L. Zhang, Multi-dimensional classification via stacked dependency exploitation, Science China Information Sciences (2020) in press.

[20] C. Liu, P. Zhao, S.-J. Huang, Y. Jiang, Z.-H. Zhou, Dual set multi-label learning, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, New Orleans, LA, 2018, pp. 3635–3642.

[21] Z. Ma, S. Chen, A convex formulation for multiple ordinal output classification, Pattern Recognition 86 (2019) 73–84.

[22] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning 85 (3) (2011) 333–359.

[23] I. Batal, C. Hong, 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, 2013, pp. 2417–2422.

[24] M. Zhu, S. Liu, J. Jiang, A hybrid method for learning multi-dimensional Bayesian network classifiers based on an optimization model, Applied Intelligence 44 (1) (2016) 123–148.

[25] M. Benjumeda, C. Bielza, P. Larrañaga, Tractability of most probable explanations in multidimensional Bayesian network classifiers, International Journal of Approximate Reasoning 93 (2018) 74–87.

[26] A. Karalič, I. Bratko, First order regression, Machine Learning 26 (1997) 147–176.

[27] D. Dheeru, E. Karra Taniskidou, UCI machine learning repository, `http://archive.ics.uci.edu/ml` (2017).

[28] S. Džeroski, D. Demšar, J. Grbović, Predicting chemical parameters of river water quality from bioindicator data, Applied Intelligence 13 (1) (2000) 7–17.

[29] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Ensembles of multi-objective decision trees, in: Lecture Notes in Computer Science 4701, Springer, Berlin, 2007, pp. 624–631.

[30] M. J. Huiskes, M. S. Lew, The MIR flickr retrieval evaluation, in: Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval, Vancouver, Canada, 2008, pp. 39–43.

[31] M.-L. Zhang, Z.-H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, Pattern Recognition 40 (7) (2007) 2038–2048.

[32] M. R. Boutell, J. Luo, X. Shen, C. M. Brown, Learning multi-label scene classification, Pattern Recognition 37 (9) (2004) 1757–1771.

[33] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, in: Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 2001, pp. 681–687.

[34] C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, A. W. Smeulders, The challenge problem for automated detection of 101 semantic concepts in multimedia, in: Proceedings of the 14th ACM International Conference on Multimedia, Santa Barbara, CA, 2006, pp. 421–430.

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

[36] Z.-H. Zhou, Abductive learning: Towards bridging machine learning and logical reasoning, Science China Information Sciences 62 (7) (2019) Article 076101.

[37] M.-L. Zhang, L. Wu, LIFT: Multi-label learning with label-specific features, IEEE Transactions on Pattern Analysis and Machine Intelligence 37 (1) (2015) 107–120.

[38] X.-Y. Jia, S.-S. Zhu, W.-W. Li, Joint label-specific features and correlation information for multi-label learning, Journal of Computer Science and Technology 35 (2) (2020) 247–258.