Pattern Recognition 113 (2021) 107833

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Compact learning for multi-label classification

Jiaqi Lv[a,b], Tianran Wu[a,b], Chenglun Peng[a,b], Yunpeng Liu[a,b], Ning Xu[a,b], Xin Geng[a,b,*]

[a] School of Computer Science and Engineering, Southeast University, Nanjing, China
[b] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China

ARTICLE INFO

Article history:
Received 29 November 2019
Revised 22 November 2020
Accepted 16 January 2021
Available online 21 January 2021

Keywords:
Machine learning
Multi-label classification
Label compression
Compact learning

ABSTRACT

Multi-label classification (MLC) studies the problem where each instance is associated with multiple relevant labels, which leads to the exponential growth of output space. It confronts with the great challenge for the exploration of the latent label relationship and the intrinsic correlation between feature and label spaces. MLC gave rise to a framework named *label compression* (LC) to obtain a compact space for efficient learning. Nevertheless, most existing LC methods failed to consider the influence of the feature space or misguided by original problematic features, which may result in performance degradation instead. In this paper, we present a *compact learning* (CL) framework to embed the features and labels *simultaneously and with mutual guidance*. The proposal is a versatile concept that does not rigidly adhere to some specific embedding methods, and is independent of the subsequent learning process. Following its spirit, a simple yet effective implementation called *compact multi-label learning* (CMLL) is proposed to learn a compact low-dimensional representation for both spaces. CMLL maximizes the dependence between the embedded spaces of the labels and features, and minimizes the loss of label space recovery concurrently. Theoretically, we provide a general analysis for different embedding methods. Practically, we conduct extensive experiments to validate the effectiveness of the proposed method.

1. Introduction

Multi-label classification (MLC) [1] is one of the mostly-studied machine learning paradigms, owing its popularity to its capability to fit the pervasive real-world tasks. It allows each instance to be equipped with multiple relevant labels for explicitly expressing the rich semantic meanings simultaneously. In recent years, the need of MLC is widely witnessed in various applications, such as image annotation [2,3], face recognition [4,5], text categorization [6,7], etc.

Formally speaking, let $\mathcal{X} \subseteq \mathbb{R}^D$ be the instance space and $\mathcal{Y} = [M]$ be the label space, where D is the feature space dimension, $[M] := \{1, 2, \ldots, M\}$ and $M > 2$ is the number of classes. The training set is represented as $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^N$ consisting of a D-dimensional instance $\boldsymbol{x}_i \in \mathcal{X}$ and the associated label set $Y_i \subseteq \mathcal{Y}$. MLC techniques aim at inducing a multi-label classifier $g : \mathcal{X} \to \mathcal{Y}$ to assign a set of relevant labels for the unseen instances.

It is evident that MLC can be regarded as a generalization of traditional single-label learning. However, the generality inevitably leads to the output space grows exponentially as the number of classes increases. A prominent phenomenon is the *sparsity* of the label space, including *label-set sparsity* and *hypercube sparsity* [8]. The former means that the instances are usually associated with very few relevant labels compared to the label dimensionality, while the latter means that the label combinations covered by limited training data are far smaller in number than all possible combinations (i.e., the power sets of the label space). For example, suppose an image annotation task with 50 candidate labels, an image could often be related to no more than ten objects (i.e., label-set sparsity) and the collected training examples is far less than 2^{50} possible label combinations (i.e., hypercube sparsity). When the label space is considerably large, most of the conventional MLC algorithms become computationally inefficient, let alone tends to be corrupted by noisy labeling [9]. Therefore, how to fully capture the high-order correlation behind the observed features and labels efficiently becomes a major thrust of research.

There are many attempts to cope with the challenge, among which *label compression* (LC) [10–13] is the dominant strategy. LC embeds the original high-dimensional label space into a low-dimensional subspace so as to gain a tighter label representation that captures the intrinsic structural information among the labels, followed by learning an association between the feature space and embedded label space. And with proper decoding process that maps the embedding label space back to the original label space, the classification purpose achieves. In this way, problems such as

* Corresponding author at: School of Computer Science and Engineering, Southeast University, Nanjing, China.
E-mail address: xgeng@seu.edu.cn (X. Geng).

https://doi.org/10.1016/j.patcog.2021.107833
0031-3203/

redundancy and sparsity existing in the original label space can be alleviated to some extent, and also reduce the computational and space complexities.

While most LC methods only concentrated on the embedding of labels while keeping the features unchanged. Some of them [8,10,14,15] made assumptions on the embedded label space but totally ignored the influence of the features, which causes the loss of discriminant information. There are a few initial efforts in jointly utilizing the features [16,17]. However, learning informative features which are discriminative to the targets and also representative to the instances is still being explored, hence it is unrealistic to hope to obtain features without noise and redundancy for providing perfect guidance for label embedding. Correspondingly, some *feature embedding* (FE) methods [18,19] have been proposed to tackle the problem in feature space. Experience proves that the predictability of features is always task-dependent, in other words, label-specific, thus the FE process should also be guided by the labels. The separate embedding of a single space driven by another problematic space provokes the propagation and accumulation of errors.

In light of the above observations, we focus on studying a general framework that co-embeds the two spaces to fully capture the high-order correlation between features and labels. First we argue that:

- The embedding process of the label space and the feature space should be *linked* to each other and performed *simultaneously*;
- The embedding process of one space should be guided by another *well-disposed* space rather than the original problematic space.

Such a framework is named *Compact Learning* (CL) in the sense that the learning is based on the two compact spaces. Then, to this end, we propose a simple yet effective algorithm called *Compact Multi-Label Learning* (CMLL). CMLL aims to learn a more compact representation for both labels and features by maximizing the dependence between the embedded spaces of the labels and features, and simultaneously minimizing the recovery loss from the embedded labels to the original ones. In this way, the embedding processes of the two spaces are seamless and mutually guided, and the feature-only embedding or label-only embedding can be regarded as a simplified version of it. We conduct comprehensive experiments over twelve benchmark datasets to validate the effectiveness of CMLL in improving the classification performance.

The rest of the paper is organized as follows. Section 2 briefly discusses the related work. Section 3 presents the technical details of the proposed CMLL approach. Then, Section 4 conducts some theoretical analyses on LC and CL framework. Section 5 reports the experimental results. At last, Section 6 concludes this paper.

2. Related work

In this section, we mainly introduce the LC framework and review the important works in the field of LC. LC is a popular strategy for MLC where the target is to embed the original labels into a low-dimensional latent space. Generally speaking, LC consists of the following three processes:

1. Encoding/embedding process: It embeds the original label vectors into a compressed space through a specific transformation $e : \mathcal{Y} \to \mathcal{V}$, where $\mathcal{V} = [m](m \ll M)$ is the embedded m-dimensional label space.
2. Learning process. It induces a multi-label classifier from the feature space to the embedded space $g' : \mathcal{X} \to \mathcal{V}$.
3. Decoding/recovery process: It recovers the original labels from the embedded label space via a decoder $d : \mathcal{V} \to \mathcal{Y}$.

For a new instance $\boldsymbol{x}$, the predicted labels $\widehat{Y}$ is: $\widehat{Y} = d(g'(\boldsymbol{x}))$.

Most LC methods learned the embedding labels with the feature unchanged. Hsu et al. [10] conducted LC via compressive sensing, which is time-consuming in the decoding process since it needs to solve an optimization problem for each new instance. Unlike compressive sensing, Tai and Lin [8] proposed a principle label space transformation (PLST) method, which is essentially a principal components analysis in the label space. Then, based on canonical correlation analysis [20], Conditional PLST (CPLST) [16] and CCA-OC [21] improved PLST from the point of feature information. Zhang et al. [22] put forward a method to maximize the dependence between features and embedding labels. Some LC methods applied the randomized techniques to speed up the computing [23,24].

Aside from the linear mapping methods mentioned above, another kind of LC methods reduced the label-space dimensionality via a nonlinear mapping. Li and Guo [15] applied the kernel trick to the label space. Jing et al. [25] added a trace norm regularization to identify the low-dimensional representation of the original space. To address the unsatisfactory accuracy caused by the violation of low rank assumption, Bhatia et al. [26] learned a small ensemble of local distance preserving embeddings which non-linearly captures label correlations. Rai et al. [11] presented a scalable Bayesian framework via a non-linear mapping. Jian et al. [27] decomposed the original label space to a low-dimensional space to reduce the noisy information in the label space. Wicker et al. [28] proposed a model that compresses the labels by autoencoders and then used the same structure to decompress the labels, which is able to capture non-linear label dependencies.

Previous researches on embedding mostly required an explicit encoding function for mapping the original labels to the embedding labels. However, since the optimal mapping can be complicated and even indescribable, assuming an explicit encoding function may not model it well. For this reason, some methods made no assumptions on the encoding process but directly learned a code matrix [14,17].

Quite different from the approach of existing LC methods, we propose CMLL following the spirit of CL. There are several quite related works have been proposed which transformed both feature and label space onto the same space. From now on, we discuss the main differences between our proposal with them, and later in Section 5, we experimentally validate our superiority. One is canonical correlated autoencoder (C2AE) [12], whose embedded space is shared by feature and label, and the tasks of label embedding and multi-label prediction are integrated into the same framework. The other is co-hashing (CoH) proposed in Shen et al. [29], which also learned a common latent hamming space and then applied k-nearest neighbor (kNN) for predicting. Both of these two methods can be categorized as an implementation of CL, but they are coupled with some specifical learning algorithms, and cannot be a routine extension of feature-only embedding or label-only embedding. In our work, we surmount these difficulties through introducing a method that learn two embedding spaces for the feature and the label respectively, and moreover, inducing the classifier is independent of the encoder and the decoder, so that any parametric or non-parametric learning model is compatible. Futhermore, embedding to a single space makes the compression ratios of two spaces mutual restrict, by contrast, CMLL supports the compression ratio for each space varies independently.

3. Proposed algorithm

In this section, a novel method CMLL in the spirit of CL is proposed. In addition to a encoding process e for the labels, a mapping that projects features is also required, i.e., $e' : \mathcal{X} \to \mathcal{U}$, where $\mathcal{U} = \mathbb{R}^d$ is the embedded d-dimensional feature space ($d \leq D$). For

an unseen instance $\boldsymbol{x}_u$, the predicted relavant labels $\widehat{Y}_u$ are: $\widehat{Y}_u = d(g(e'(\boldsymbol{x}_u)))$, where $g : \mathcal{U} \to \mathcal{V}$ is a multi-label classifier.

In most cases, the classifier learned by a MLC system is a real-valued function [1] $f : \mathcal{X} \to \mathbb{R}_+^c$, where $f(\boldsymbol{x})$ can be regarded as the confidences of the labels being the relevant labels of $\boldsymbol{x}$. Specifically, given a multi-label example $(\boldsymbol{x}, Y)$, if $i \in Y$, $j \notin Y$, the i-element in $f(\boldsymbol{x})$ should be larger than the j-element in $f(\boldsymbol{x})$. Note that the multi-label classifier g can be derived from the real-valued function f via: $g(\boldsymbol{x}) = \mathbb{I}(f(\boldsymbol{x}), \delta) = \{i | f(\boldsymbol{x})_i > \delta\}$, where $\delta \in (0, 1)$ is a threshold and $f(\boldsymbol{x})_i$ is the i-element of $f(\boldsymbol{x})$.

3.1. The objective of CMLL

For boosting the performance, we should make the instances more *predictable* in the learning process and the embedded label vectors more *recoverable* in the decoding process. In this section, we propose a simple yet effective instantiation CMLL of CL framework.

It has been widely acknowledged that strong correlation usually leads to better predictability, hence CMLL maximizes the dependence between the embedded label space $\mathcal{V}$ and the embedded feature space $\mathcal{U}$. At the same time, CMLL minimizes the recovery loss from $\mathcal{Y}$ to $\mathcal{V}$. Let $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ be the feature matrix and $\boldsymbol{Y} \in \{0, 1\}^{N \times M}$ be the corresponding label matrix. Given the training dataset $S = \{\boldsymbol{X}, \boldsymbol{Y}\}$, we denote $\Omega(\boldsymbol{V}, \boldsymbol{Y})$ as the recovery loss and $\Theta(\boldsymbol{V}, \boldsymbol{U})$ as the measure of dependence, where $\boldsymbol{V}^{N \times m}$ is the embedded label matrix and $\boldsymbol{U}^{N \times d}$ is the the embedded feature matrix. Then, the objective can be formulized as follows:

$$\max_{\boldsymbol{U}, \boldsymbol{V}} \ \alpha \Theta(\boldsymbol{V}, \boldsymbol{U}) - \Omega(\boldsymbol{V}, \boldsymbol{Y}), \tag{1}$$

where α is a hyper-parameter that balances the importance of the dependence and the recovery loss. Next we discuss these two terms in Eq. (1) respectively with the concrete form.

To characterize the dependence between two spaces, we utilize Hilbert-Schmidt Independence Criterion (HSIC) [19,30] due to its simple form and theoretical properties. HSIC calculates the squared norm of the cross-covariance operator over the domain $\mathcal{X} \times \mathcal{Y}$ in reproducing kernel Hilbert spaces. An empirical estimate of HSIC is given by:

$$HSIC(\mathcal{X}, \mathcal{Y}) = (N-1)^{-2} \ tr[\boldsymbol{HKHL}],$$

where $tr[\cdot]$ denotes the trace of a matrix, $\boldsymbol{H} = \boldsymbol{I} - \frac{1}{N}\boldsymbol{e}\boldsymbol{e}^T$, $\boldsymbol{e}^{N \times 1}$ is an all-one vector, and $\boldsymbol{I}^{N \times N}$ is the unit matrix. $\boldsymbol{K}_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ and $\boldsymbol{L}_{ij} = l(\boldsymbol{y}_i, \boldsymbol{y}_j) = \langle \varphi(\boldsymbol{y}_i), \varphi(\boldsymbol{y}_j) \rangle$, where $\langle \cdot \rangle$ represents the inner product operation, $k(\cdot)$ and $l(\cdot)$ are the kernel functions, and $\phi(\cdot)$ and $\varphi(\cdot)$ are the corresponding mapping functions. Adopting HSIC and dropping the normalization term, the measure of dependence can be represented as:

$$\Theta(\boldsymbol{V}, \boldsymbol{U}) = tr[\boldsymbol{HKHL}],$$

where $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{U}^t$ and $\boldsymbol{L} = \boldsymbol{V}\boldsymbol{V}^t$. We first consider the linear embedding of features in CMLL, i.e. $\boldsymbol{U} = \boldsymbol{X}\boldsymbol{P}$, where $\boldsymbol{P}^{D \times d}$ is the learnt projection matrix. Constraining the basis of the projection matrix to be orthonormal, we derive:

$$\begin{aligned} \Theta(\boldsymbol{V}, \boldsymbol{U}) = & \ tr[\boldsymbol{H}\boldsymbol{X}\boldsymbol{P}\boldsymbol{P}^t\boldsymbol{X}^t\boldsymbol{H}\boldsymbol{V}\boldsymbol{V}^t] \\ & s.t. \ \ \boldsymbol{P}^t\boldsymbol{P} = \boldsymbol{I}. \end{aligned} \tag{2}$$

In order to minimize the loss of recovery, CMLL searches a decoding matrix $\boldsymbol{W}^{m \times M}$ through the ridge regression [31] to conduct a linear decoding. That is,

$$\Omega(\boldsymbol{V}, \boldsymbol{Y}, \boldsymbol{W}) = ||\boldsymbol{Y} - \boldsymbol{V}\boldsymbol{W}||_2^2 + \lambda ||\boldsymbol{W}||_F^2, \tag{3}$$

where $||\cdot||_F$ means the Frobenious norm and λ is the tradeoff parameter. Given the specific $\boldsymbol{V}$ and $\boldsymbol{Y}$, the goal is to find the $\boldsymbol{W}$ to minimize $\Omega(\boldsymbol{V}, \boldsymbol{Y}, \boldsymbol{W})$. To avoid redundancy in the embedded label space, we assume that the components of the embedded space are orthonormal and uncorrelated, i.e., $\boldsymbol{V}^t\boldsymbol{V} = \boldsymbol{I}$. Then, let the partial derivative of $\Omega(\boldsymbol{V}, \boldsymbol{Y}, \boldsymbol{W})$ with respect to $\boldsymbol{W}$ be zero:

$$\begin{aligned} \frac{\partial \Omega}{\partial \boldsymbol{W}} &= \frac{tr[\boldsymbol{Y}\boldsymbol{Y}^t + \boldsymbol{W}\boldsymbol{W}^t\boldsymbol{V}^t\boldsymbol{V} - 2\boldsymbol{W}^t\boldsymbol{V}^t\boldsymbol{Y} + \lambda \boldsymbol{W}^t\boldsymbol{W}]}{\partial \boldsymbol{W}} \\ &= 2\boldsymbol{V}^t\boldsymbol{V}\boldsymbol{W} - 2\boldsymbol{V}^t\boldsymbol{Y} + 2\lambda\boldsymbol{W} = 0. \end{aligned}$$

We can obtain:

$$\boldsymbol{W} = (\boldsymbol{V}^t\boldsymbol{V} + \lambda\boldsymbol{I})^{-1}\boldsymbol{V}^t\boldsymbol{Y} = \frac{1}{1+\lambda}\boldsymbol{V}^t\boldsymbol{Y}. \tag{4}$$

Substituting Eq. (4) into Eq. (3) and dropping unrelated items, we yield:

$$\begin{aligned} \Omega(\boldsymbol{V}, \boldsymbol{Y}) = & -\frac{1}{1+\lambda} tr[\boldsymbol{Y}^t\boldsymbol{V}\boldsymbol{V}^t\boldsymbol{Y}] \\ & s.t. \ \ \boldsymbol{V}^t\boldsymbol{V} = \boldsymbol{I}. \end{aligned} \tag{5}$$

Then substituting Eqs. (5) and (2) into Eq. (1), the terms in the objective can be derived as follows:

$$\begin{aligned} & \alpha \ tr[\boldsymbol{H}\boldsymbol{U}\boldsymbol{U}^t\boldsymbol{H}\boldsymbol{V}\boldsymbol{V}^t] + \frac{1}{1+\lambda} tr[\boldsymbol{Y}^t\boldsymbol{V}\boldsymbol{V}^t\boldsymbol{Y}] \\ \Leftrightarrow \ & \alpha(1+\lambda) \ tr[\boldsymbol{V}^t\boldsymbol{H}\boldsymbol{U}\boldsymbol{U}^t\boldsymbol{H}\boldsymbol{V}] + tr[\boldsymbol{V}^t\boldsymbol{Y}\boldsymbol{Y}^t\boldsymbol{V}] \\ \Leftrightarrow \ & tr[\boldsymbol{V}^t(\beta\boldsymbol{H}\boldsymbol{U}\boldsymbol{U}^t\boldsymbol{H} + \boldsymbol{Y}\boldsymbol{Y}^t)\boldsymbol{V}], \end{aligned}$$

where $\beta = \alpha(1+\lambda)$ is the normalized balance parameter. Adding the corresponding constraints, the learning objective becomes:

$$\begin{aligned} \max_{\boldsymbol{V}, \boldsymbol{P}} \ & tr[\boldsymbol{V}^t(\beta\boldsymbol{H}\boldsymbol{X}\boldsymbol{P}\boldsymbol{P}^t\boldsymbol{X}^t\boldsymbol{H} + \boldsymbol{Y}\boldsymbol{Y}^t)\boldsymbol{V}] \\ s.t. \ \ & \boldsymbol{V}^t\boldsymbol{V} = \boldsymbol{I}, \ \boldsymbol{P}^t\boldsymbol{P} = \boldsymbol{I}. \end{aligned} \tag{6}$$

3.2. Solution for CMLL

We solve Eq. (6) by alternating minimization. In each iteration, fixing one of $\{\boldsymbol{P}, \boldsymbol{V}\}$ and updating the other with coordinate ascent [31], in which way a close-form solution can be obtained.

To be specific, when $\boldsymbol{P}$ is fixed, the problem is converted into an eigen-decomposition problem after applying the Lagrangian method. Let $\boldsymbol{A} = (\beta\boldsymbol{H}\boldsymbol{U}\boldsymbol{U}^t\boldsymbol{H} + \boldsymbol{Y}\boldsymbol{Y}^t)$, the eigen-decomposition problem is specified as:

$$\begin{aligned} \max_{\boldsymbol{V}} \ & \sum_{j=1}^{m} \gamma_j \\ s.t. \ \ & \boldsymbol{A}\boldsymbol{V}_{\cdot j} = \gamma_j \boldsymbol{V}_{\cdot j}, \ \boldsymbol{V}_{\cdot i}^t\boldsymbol{V}_{\cdot j} = \mathbb{I}(i = j), \end{aligned}$$

where $\boldsymbol{V}_{\cdot j}$ is the jth column of $\boldsymbol{V}$, and γ_j means the eigenvalue. The optimal $\boldsymbol{V}$ consists of m normalized eigenvectors corresponding to the top m largest eigenvalues of $\boldsymbol{A}$. Notice that m is usually much smaller than M, so we can utilize some iterative approaches such as Arnoldi iteration [32] to accelerate computing to a minimal computational complexity $\mathcal{O}(Nm^2)$. When $\boldsymbol{V}$ is fixed, the optimal $\boldsymbol{P}$ consists of d normalized eigenvectors corresponding to the top d largest eigenvalues of $\boldsymbol{B} = \boldsymbol{X}^t\boldsymbol{H}\boldsymbol{V}\boldsymbol{V}^t\boldsymbol{H}\boldsymbol{X}$.

The procedures of CMLL are summarized in Algorithm 1. It is interesting to note that if we replace $\boldsymbol{V}$ with $\boldsymbol{Y}$ in $\boldsymbol{B}$, regardless of the embedding for the labels, the solution for $\boldsymbol{P}$ is actually the same as MDDM [19], a typical FE method for MLC. And if we replace $\boldsymbol{U}$ with $\boldsymbol{X}$ in $\boldsymbol{A}$, a standard LC algorithm regardless of the embedding for the features is derived, which we named as CMLL$_y$. Both MDDM and CMLL$_y$ can be regarded as the simplified versions of CMLL.

3.3. Kernelization for CMLL

We can utilize kernel tricks to extend CMLL to the non-linear case, denoted by k-CMLL. Assume the projection matrix $\boldsymbol{P}$ can be spanned by kernel feature vectors, i.e. $\boldsymbol{P} = \Phi(\boldsymbol{X})\boldsymbol{R}^{N \times d}$, where $\Phi(\boldsymbol{X}) = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_N)]$. $\phi(.)$ is the projection function

Algorithm 1 CMLL.

Input: Training dataset $S=\{\boldsymbol{X},\boldsymbol{Y}\}$, testing feature matrix $\boldsymbol{X}_{test}$, hyper-parameters β,λ, dimensionality of the embedded label space m and feature space d, maximal iteration count *maxc*, toleration *tol*.
Output: Predicted label matrix $\widehat{\boldsymbol{Y}}_{pre}$.

Initialize $j=0$, $\boldsymbol{V}^0_{N*m}$, $\boldsymbol{P}^0_{D*d}$ with a random matrix.
Get $\Gamma^0=tr[\boldsymbol{V}^0(\beta\boldsymbol{H}\boldsymbol{X}\boldsymbol{P}^0(\boldsymbol{P}^0)^t\boldsymbol{X}^t\boldsymbol{H}+\boldsymbol{Y}\boldsymbol{Y}^t)\boldsymbol{V}^0]$.
repeat
  Get $\boldsymbol{A}^{j+1}=\beta\boldsymbol{H}\boldsymbol{X}\boldsymbol{P}^{j+1}(\boldsymbol{P}^{j+1})^t\boldsymbol{X}^t\boldsymbol{H}+\boldsymbol{Y}\boldsymbol{Y}^t$, then obtain $\boldsymbol{V}^{j+1}$ via eigen-decomposition.
  Get $\boldsymbol{B}^{j+1}=\boldsymbol{X}^t\boldsymbol{H}\boldsymbol{V}^{j+1}(\boldsymbol{V}^{j+1})^t\boldsymbol{H}\boldsymbol{X}$, then obtain $\boldsymbol{P}^{j+1}$ via eigen-decomposition.
  Get Γ^{j+1} using $\boldsymbol{P}^{j+1}$ and $\boldsymbol{V}^{j+1}$
  Compute $\Delta=|\Gamma^{j+1}-\Gamma^j|/(\Gamma^j)$.
  Let $j=j+1$, $\boldsymbol{P}=\boldsymbol{P}^j$, $\boldsymbol{V}=\boldsymbol{V}^j$.
until $(j>maxc)$ or $(\Delta<tol)$
Compute $\boldsymbol{W}=\frac{1}{1+\lambda}\boldsymbol{V}^t\boldsymbol{Y}$.
Learn the classifier: $g:\boldsymbol{X}\boldsymbol{P}\rightarrow\boldsymbol{V}$.
Conduct prediction: $\boldsymbol{V}_{pre}=g(\boldsymbol{X}_{test}\boldsymbol{P})$.
Perform decoding: $\widehat{\boldsymbol{Y}}_{pre}=\mathbb{I}(\boldsymbol{V}_{pre}\boldsymbol{W},\delta)$.

corresponding to the kernel and $\boldsymbol{R}$ is the matrix of the corresponding linear combination coefficients.

Let $q(\boldsymbol{x}_i,\boldsymbol{x}_j)=\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_j)\rangle$ be the kernel function and $\boldsymbol{Q}=\Phi(\boldsymbol{X})^t\Phi(\boldsymbol{X})$ be the kernel matrix. Then, $\boldsymbol{U}=\Phi(\boldsymbol{X})^t\boldsymbol{P}=\boldsymbol{Q}\boldsymbol{R}$, $\boldsymbol{K}=\boldsymbol{U}\boldsymbol{U}^t=\boldsymbol{Q}\boldsymbol{R}\boldsymbol{R}^t\boldsymbol{Q}$, and the constraint $\boldsymbol{P}^t\boldsymbol{P}=\boldsymbol{R}^t\boldsymbol{Q}\boldsymbol{R}=\boldsymbol{I}$. So the objective of the kernel CMLL becomes:

$$\max_{\boldsymbol{O},\boldsymbol{R}}\ tr[\boldsymbol{V}^t(\beta\boldsymbol{H}\boldsymbol{Q}\boldsymbol{R}\boldsymbol{R}^t\boldsymbol{Q}\boldsymbol{H}+\boldsymbol{Y}\boldsymbol{Y}^t)\boldsymbol{V}],$$
$$s.t.\ \ \boldsymbol{V}^t\boldsymbol{V}=\boldsymbol{I},\ \ \boldsymbol{R}^t\boldsymbol{Q}\boldsymbol{R}=\boldsymbol{I}.$$

The solution for k-CMLL is similar to that of the linear case. When $\boldsymbol{R}$ is fixed, the optimal $\boldsymbol{V}$ consists of the top m eigenvectors of $\boldsymbol{A}'=\beta\boldsymbol{H}\boldsymbol{Q}\boldsymbol{R}\boldsymbol{R}^t\boldsymbol{Q}\boldsymbol{H}+\boldsymbol{Y}\boldsymbol{Y}^t$. And when $\boldsymbol{V}$ is fixed, the optimal $\boldsymbol{R}$ consists of the top d generalized eigenvectors of $\boldsymbol{B}'=\boldsymbol{Q}\boldsymbol{H}\boldsymbol{V}\boldsymbol{V}^t\boldsymbol{H}\boldsymbol{Q}$ and $\boldsymbol{Q}$. Given an unseen instance $\boldsymbol{x}$, the projection is $\boldsymbol{u}=\boldsymbol{P}^t\phi(\boldsymbol{x})=\boldsymbol{R}^tq(\boldsymbol{X},\boldsymbol{x})$, where $q(\boldsymbol{X},\boldsymbol{x})=[q(\boldsymbol{x}_1,\boldsymbol{x}),q(\boldsymbol{x}_2,\boldsymbol{x}),\ldots,q(\boldsymbol{x}_N,\boldsymbol{x})]^t$.

4. Theoretical analysis

In this section, we will conduct a general theoretical analysis for different embedding strategies and compare them basing on the proposed Theorem 1.

Theorem 1. *Given an instance $\boldsymbol{x}$ in a multi-label dataset S, denote $\boldsymbol{y}$ as its true label vector, and $\widehat{\boldsymbol{y}}$ as its predicted real-valued label vector obtained via a specific embedding framework. Assume a fixed threshold $\delta\in(0,1)$ is used in the final step to binarize $\widehat{\boldsymbol{y}}$. The zero-one loss in MLC is defined by $\ell_{01}(\boldsymbol{y}\widehat{\boldsymbol{y}})=\sum_{i=1}^{M}\mathbb{I}(\widehat{\boldsymbol{y}}^i\geq\delta)\mathbb{I}(\boldsymbol{y}^i=0)+\mathbb{I}(\widehat{\boldsymbol{y}}^i<\delta)\mathbb{I}(\boldsymbol{y}^i=1)$ where $\boldsymbol{y}^i/\widehat{\boldsymbol{y}}^i$ is ith dimension of $\boldsymbol{y}/\widehat{\boldsymbol{y}}$, then is upper-bounded by:*

$$\ell_{01}(\boldsymbol{y}\widehat{\boldsymbol{y}})\leq\ \tau||\widehat{\boldsymbol{y}}-\boldsymbol{y}||_2^2,\qquad \tau=\ max\ \{\frac{1}{\delta^2},\frac{1}{(1-\delta)^2}\}.$$

Proof.

$$\ell_{01}(\boldsymbol{y}\widehat{\boldsymbol{y}})=\sum_{i=1}^{M}\mathbb{I}(\widehat{\boldsymbol{y}}^i\geq\delta)\mathbb{I}(\boldsymbol{y}^i=0)+\mathbb{I}(\widehat{\boldsymbol{y}}^i<\delta)\mathbb{I}(\boldsymbol{y}^i=1)\tag{7}$$
$$\leq\sum_{i=1}^{M}\frac{(\widehat{\boldsymbol{y}}^i-\boldsymbol{y}^i)^2\mathbb{I}(\boldsymbol{y}^i=0)}{\delta^2}+\frac{(\widehat{\boldsymbol{y}}^i-\boldsymbol{y}^i)^2\mathbb{I}(\boldsymbol{y}^i=1)}{(1-\delta)^2}$$
$$\leq\sum_{i=1}^{M}max\{\frac{1}{\delta^2},\frac{1}{(1-\delta)^2}\}\ \ (\widehat{\boldsymbol{y}}^i-\boldsymbol{y}^i)^2=\tau||\widehat{\boldsymbol{y}}-\boldsymbol{y}||_2^2.$$

□

Eq. (7) takes the equal sign if and only if $\delta=0.5,\tau=4$. According to this theorem, we can minimize the surrogate loss (ℓ_2 loss) to upper bound the classification error, i.e., the standard performance measure in classification. From now on, we make an preliminary analysis on the risk of these different embedding frameworks based on Theorem 1. Assume that $h:\mathcal{U}\rightarrow\mathcal{Y}$, a natural result inferred directly from the Theorem 1 is that:

$$\widehat{\mathcal{R}}_{FE}=\sum_{i=1}^{N}\tau||h(e'(\boldsymbol{x}_i))-\boldsymbol{y}_i||_2^2,$$
$$\widehat{\mathcal{R}}_{LC}=\sum_{i=1}^{N}\tau||d(g'(\boldsymbol{x}_i))-\boldsymbol{y}_i||_2^2,$$
$$\widehat{\mathcal{R}}_{CL}=\sum_{i=1}^{N}\tau||d(g(e'(\boldsymbol{x}_i)))-\boldsymbol{y}_i||_2^2,$$

where $\widehat{\mathcal{R}}$ denotes the empirical risk. And the common practice of supervised classification, empirical risk minimization (ERM), is used.

While in practical implementation, LC usually formalizes as an equivalent form:

$$\widetilde{\mathcal{R}}_{LC}=\sum_{i=1}^{N}\tau\parallel[d(g'(\boldsymbol{x}_i))-d(e(\boldsymbol{y}_i))]+[d(e(\boldsymbol{y}_i))-\boldsymbol{y}_i]\parallel_2^2,\tag{8}$$

because directly minimizing the original $\widehat{\mathcal{R}}_{LC}$ is not intuitional and feasible when designing a concrete algorithm [8,15–17,22]. That is, LC tries to make the encoded labels $e(\boldsymbol{y})$ more predictable for $\boldsymbol{x}$ and more recoverable to $\boldsymbol{y}$. For example, substituting the concrete form of PLST to Eq. (8) yields:

$$\widetilde{\mathcal{R}}_{PLST}=\sum_{i=1}^{N}4\parallel[g'(\boldsymbol{x}_i)-\boldsymbol{y}_i\boldsymbol{O}]\boldsymbol{O}^T+[\boldsymbol{y}_i\boldsymbol{O}\boldsymbol{O}^T-\boldsymbol{y}_i]\parallel_2^2,$$

where $\boldsymbol{O}$ is the orthonormal projection matrix learnt by PLST. Note that $\widehat{\mathcal{R}}_{PLST}$ is derived through the equivalent transformation [8].

Similarly, $\widehat{\mathcal{R}}_{CL}$ can be formalized as follows:

$$\widetilde{\mathcal{R}}_{CL}=\sum_{i=1}^{N}\tau\parallel[d(g(e'(\boldsymbol{x}_i)))-d(e(\boldsymbol{y}_i))]+[d(e(\boldsymbol{y}_i))-\boldsymbol{y}_i]\parallel_2^2$$

Compared to FE and LC, CL considers the transformation for both label and feature space simultaneously, and thus provides a greater possibility as well as a more flexible and superior way to make upper bound tighter. Think of a specific example CMLL, the empirical risk can be expressed as:

$$\widehat{\mathcal{R}}_{CMLL}=\sum_{i=1}^{N}4||[g(e'(\boldsymbol{x}_i))-\boldsymbol{v}_i]W+[\boldsymbol{v}_iW-\boldsymbol{y}_i]||_2^2.$$

The derivation process of CMLL shows that CMLL indeed takes both terms into account, i.e., minimizes the first term by maximizing the dependence between the two embedded spaces, and at the same time minimizes the recovery loss that measures how well $\boldsymbol{v}W$ approximates $\boldsymbol{y}$. Besides, CL can also degenerate to FE or LC when necessary, as the example of special cases of CMLL (i.e. MDDM and CMLL$_{\boldsymbol{y}}$) indicates.

The upper bound derived here seems loose because it aims at embedding strategies rather than any concrete algorithm. There are few research on the analysis of the framework of embedding yet, although many related methods have been proposed. This section makes an initial attempt to analyze the reasonability of CL as well as LC and FE, on which existing LC methods can be explained/derived based. It provides guidance on the aspects that should be considered when designing a new CL or LC algorithm.

Table 1
Characteristics of the real-world multi-label datasets.

Datasets	#Label	#Feature	#Example	Feature Type	Cardinality	Density	Distinct	Domain
plant	12	440	978	numeric	1.079	0.090	32	biology
msra	19	898	1868	numeric	6.315	0.332	947	images
enron	53	1001	1702	nominal	3.378	0.064	753	text
llog	74	1004	1460	nominal	1.128	0.015	286	text
bibtex	159	1836	5000	nominal	2.397	0.015	2127	text
eurlex-sm	201	5000	5000	numeric	2.224	0.011	1236	text
bookmarks	208	2150	5000	nominal	2.016	0.010	1840	text
corel5k	374	499	5000	nominal	3.522	0.009	3175	images
eurlex-dc	412	5000	5000	numeric	1.296	0.003	859	text
espgame	1932	516	5000	numeric	4.689	0.002	4734	images
Delicious	983	500	16,105	numeric	19.020	0.002	15,806	text
Mediamill	101	120	43,907	numeric	4.376	0.004	6555	vedio

5. Experiments

5.1. Datasets

We evaluate the proposed method on a total of twelve public real-world multi-label datasets, which show obvious label sparsity. The dataset of espgame collected here is organized by Lin et al. [17], and other small-scale datasets (the number of examples is less than 5000) can be downloaded from Mulan[1] and Meka.[2] In addition, the extreme multi-label learning [33] which aims to learn relevant labels from an extremely large label set is a possible application domain of LC, thus, we also adopt two extreme classification datasets Mediamill and Delicious.[3] Table 1 summarizes the detailed characteristics of these datasets, which are organized in ascending order of the number of examples. Cardinality means the average number of relevant labels per instance, Density is the ratio of Cardinality to the number of classes, and Distinct is the number of distinct label combinations contained in the dataset. As indicated by quite small values of Density and Distinct compared to all the possible label combinations (i.e. $2^{\#Label}$), all datasets suffer from evident hypercube sparsity or label-set sparsity in the label space.

5.2. Setups

We compare CMLL with its special cases CMLL$_{\boldsymbol{y}}$, one FE algorithm MDDM [19], one state-of-the-art large-scale multi-label learning algorithm POP [33], and five well-established LC algorithms: PLST [8], CPLST [16], FaIE [17], DMLR [22] and C2AE [12].

The hyper-parameters of the baselines were selected according to the suggested parameter settings in original papers. The balance parameter of CMLL, CMLL$_{\boldsymbol{y}}$, FaIE, DMLR was selected from $\{10^{-5}, 10^{-4}, \ldots, 10^{4}, 10^{5}\}$. POP used Binary Relevance as the base classifier. The tradeoff parameter λ of CMLL and CMLL$_{\boldsymbol{y}}$ was selected from $\{0, 10^{-3}, 10^{-1}\}$, and $tol = 10^{-5}$, $maxc = 50$. And $\delta = 0.5$ in the final step for binarizing the real-valued outputs. Besides, we also compared our kernel versions k-CMLL and k-CMLL$_{\boldsymbol{y}}$ with the baselines (except PLST and POP that can hardly be extended to the kernel version) and C2AE, which adopted the DNN architectures. And the RBF kernel was applied. Following the previous works [8,22], we used the ridge regression and the kernel ridge regression to train the learning model for linear case and kernel case, respectively. We denote ORI to represent the classifier learning from the original spaces.

We denote by $\mu = \frac{d}{D}, \nu = \frac{m}{M}$ the feature and the label compression ratio, and all LC methods run with μ ranging from 10% to 100% with the interval of 10% and MDDM runs with ν similarly. CMLL and C2AE run with both ν and μ. That means, CMLL needs to run with 100 ratio pairs (10×10) in total while C2AE run with 20 ratio pairs ($10 + 10$). Because C2AE essentially conducts non-linear embedding by utilizing the DNN structure and learns a shared embedded space for both labels and features, while CMLL learns two sub-spaces for labels and features respectively.

To measure the performance, we use four widely adopted metrics in multi-label classification, including *Average Precision, micro-F1, Ranking Loss* and *One Error*. The concrete definition of these metrics can be found in Zhang and Zhou [1]. For Mediamill and Delicious, we supplement two metrics popularly used in extreme multi-label learning: *Precision*@3 and *nDCG*@3.

5.3. Results

The experimental results of CMLL and k-CMLL compared with the comparing methods are shown in Tables 2 and 3 respectively. For each metric, "↓" indicates the smaller the better while "↑" indicates the larger the better. We perform five-fold cross-validation on each dataset, and use paired t-test at 10% significance level. The mean results with standard deviation are reported and the best performance is highlighted in boldface. •/◦ represents whether CMLL or k-CMLL is significantly better/worse than the comparing methods. We can observe that across all metrics, CMLL ranks 1st in the most cases in both linear and non-linear cases.

As Tables 2 and 3 show, with a suitable compression ratio, most embedding methods can achieve better performance than the baseline ORI. This indicates that there are indeed some problems such as sparsity and noise existing in both the original spaces, which leads to performance decline if not tackled properly. With a more compact representation for labels ($\nu \leq 100\%$) and features ($\mu \leq 100\%$), CMLL performs better than all compared LC methods ($\mu = 100\%$) and FE method ($\nu = 100\%$) in most cases. By applying the kernel trick to extend methods to their corresponding non-linear version, each LC method actually guides the embedding process of labels with the well transformed rather than the original features implicitly, where k-CMLL still outperforms other methods on the whole.

5.4. Parameter sensitivity analysis

To explore the influence of balance parameter α in CMLL, we fix $\mu = \nu = 50\%$, $\lambda = 0$ and run CMLL with α ranging in $\{10^{-4}, 10^{-3}, \ldots, 10^{3}, 10^{4}\}$. To be convenient, we denote $dep = tr[\boldsymbol{V}^t\boldsymbol{H}\boldsymbol{X}\boldsymbol{P}\boldsymbol{P}^t\boldsymbol{X}^t\boldsymbol{H}\boldsymbol{H}\boldsymbol{V}]$ and $rec = tr[\boldsymbol{V}^t\boldsymbol{Y}\boldsymbol{Y}^t\boldsymbol{V}]$ as the dependence term and recovery term in objective (6). Dropping the recovery term in (6), we can find the solutions of $\boldsymbol{V}$ and $\boldsymbol{P}$, and then compute corresponding values of dependence term dep_{max} and recovery term rec_{min}. Similarly, by only considering recover term in (6),

[1] http://mulan.sourceforge.net/datasets-mlc.html
[2] http://meka.sourceforge.net/#datasets
[3] http://manikvarma.org/downloads/XC/XMLRepository.html

Table 2
Experimental results of CMLL with baselines.

Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
	Average precision ↑								
plant	0.4756 ± 0.0263•	0.4873 ± 0.0299•	0.4848 ± 0.0274•	0.4859 ± 0.0250•	0.4813 ± 0.0282•	0.5399 ± 0.0311	0.5587 ± 0.0294	0.5256 ± 0.0707	0.5733 ± 0.0252
msra	0.7433 ± 0.0129•	0.7733 ± 0.0107•	0.7689 ± 0.0117•	0.7766 ± 0.0129•	0.7782 ± 0.0113•	0.7820 ± 0.0112•	0.7933 ± 0.0090	0.7315 ± 0.0134•	0.7997 ± 0.0100
enron	0.5039 ± 0.0111•	0.5183 ± 0.0162•	0.5170 ± 0.0160•	0.5196 ± 0.0110•	0.5218 ± 0.0123•	0.6228 ± 0.0119•	0.6767 ± 0.0205	0.6360 ± 0.0867	0.6965 ± 0.0153
llog	0.2235 ± 0.0274•	0.2429 ± 0.0314•	0.2412 ± 0.0320•	0.2760 ± 0.0328•	0.2722 ± 0.0344•	0.3413 ± 0.0378•	0.4264 ± 0.0216	0.1532 ± 0.0563•	0.3717 ± 0.0283
bibtext	0.5278 ± 0.0091•	0.5258 ± 0.0077•	0.5280 ± 0.0079•	0.5281 ± 0.0086•	0.5259 ± 0.0078•	0.5241 ± 0.0039•	0.5287 ± 0.0091•	0.5676 ± 0.0069	0.5828 ± 0.0050
eurlex-sm	0.3972 ± 0.0208•	0.3998 ± 0.0209•	0.3998 ± 0.0209•	0.3979 ± 0.0208•	0.4289 ± 0.0219•	0.5706 ± 0.0119•	0.7299 ± 0.0198•	0.3419 ± 0.0058•	0.7565 ± 0.0084
bookmark	0.3086 ± 0.0059•	0.3043 ± 0.0065•	0.3030 ± 0.0059•	0.3054 ± 0.0064•	0.3051 ± 0.0061•	0.3161 ± 0.0064•	0.3804 ± 0.0060•	0.4024 ± 0.0088	0.4080 ± 0.0078
corel5k	0.2892 ± 0.0029•	0.2900 ± 0.0035•	0.2908 ± 0.0039•	0.2932 ± 0.0030•	0.2916 ± 0.0035•	0.2925 ± 0.0035•	0.2995 ± 0.0057	0.0900 ± 0.0076•	0.3028 ± 0.0070
eurlex-dc	0.3982 ± 0.0315•	0.4031 ± 0.0304•	0.4031 ± 0.0304•	0.4511 ± 0.0240•	0.5031 ± 0.0305•	0.6118 ± 0.0254•	0.6911 ± 0.0215•	0.4616 ± 0.0089•	0.7588 ± 0.0144
espgame	0.2171 ± 0.0081	0.2171 ± 0.0081	0.2173 ± 0.0081	0.2175 ± 0.0082	0.2169 ± 0.0081	0.2169 ± 0.0081	0.2175 ± 0.0083	0.0141 ± 0.0006•	0.2177 ± 0.0080
Delicious	0.3338 ± 0.0029•	0.3443 ± 0.0024•	0.3450 ± 0.0025•	0.3337 ± 0.0030•	0.3337 ± 0.0029•	0.3485 ± 0.0028•	0.3514 ± 0.0025	0.2101 ± 0.0039•	0.3543 ± 0.0030
Mediamill	0.7193 ± 0.0031•	0.7217 ± 0.0033•	0.7200 ± 0.0031•	0.7218 ± 0.0035•	0.7195 ± 0.0034•	0.7171 ± 0.0035•	0.7193 ± 0.0031•	0.5389 ± 0.0096•	0.7302 ± 0.0031
	micro-F1 ↑								
Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
plant	0.2644 ± 0.0312	0.2644 ± 0.0312	0.2646 ± 0.0288	0.2460 ± 0.0256•	0.2663 ± 0.0319	0.2991 ± 0.0458	0.2652 ± 0.0322	0.2222 ± 0.0247•	0.2993 ± 0.0190
msra	0.6471 ± 0.0134•	0.6644 ± 0.0076•	0.6606 ± 0.0126•	0.6605 ± 0.0125•	0.6641 ± 0.0118•	0.6677 ± 0.0118	0.6757 ± 0.0081	0.5942 ± 0.0135•	0.6817 ± 0.0096
enron	0.4045 ± 0.0057•	0.4480 ± 0.0105•	0.4473 ± 0.0138•	0.4437 ± 0.0121•	0.4579 ± 0.0109•	0.4995 ± 0.0119•	0.5278 ± 0.0057	0.4913 ± 0.0410	0.5335 ± 0.0147
llog	0.1390 ± 0.0094•	0.1710 ± 0.0177•	0.1726 ± 0.0178•	0.1754 ± 0.0180•	0.1724 ± 0.0196•	0.1926 ± 0.0223•	0.2289 ± 0.0114	0.0942 ± 0.0292•	0.2483 ± 0.0175
bibtext	0.3942 ± 0.0107○	0.3910 ± 0.0101○	0.3939 ± 0.0117○	0.3921 ± 0.0106○	0.3914 ± 0.0099○	0.3764 ± 0.0072•	0.3940 ± 0.0102○	0.3566 ± 0.0039•	0.3683 ± 0.0104
eurlex-sm	0.1181 ± 0.0073•	0.1225 ± 0.0078•	0.1225 ± 0.0078•	0.1204 ± 0.0072•	0.2220 ± 0.0080•	0.2327 ± 0.0094•	0.3235 ± 0.0079	0.3393 ± 0.0152	0.3350 ± 0.0125
bookmark	0.1616 ± 0.0094•	0.1879 ± 0.0080•	0.1882 ± 0.0073•	0.1927 ± 0.0091•	0.1879 ± 0.0081•	0.2218 ± 0.0100•	0.2163 ± 0.0085•	0.2176 ± 0.0060•	0.2386 ± 0.0092
corel5k	0.1032 ± 0.0050•	0.1002 ± 0.0064•	0.0998 ± 0.0070•	0.1017 ± 0.0047•	0.1523 ± 0.0051○	0.1258 ± 0.0044	0.1201 ± 0.0046	0.1456 ± 0.0084○	0.1266 ± 0.0048
eurlex-dc	0.0527 ± 0.0035•	0.0545 ± 0.0035•	0.0545 ± 0.0035•	0.0889 ± 0.0057•	0.1745 ± 0.0039•	0.2486 ± 0.0107•	0.2588 ± 0.0034•	0.2251 ± 0.0015•	0.3735 ± 0.0183
espgame	0.0863 ± 0.0054	0.0863 ± 0.0055	0.0858 ± 0.0048	0.0859 ± 0.0050	0.0861 ± 0.0052	0.0863 ± 0.0051	0.0833 ± 0.0036	0.0098 ± 0.0003•	0.0860 ± 0.0055
Delicious	0.1614 ± 0.0033•	0.1639 ± 0.0041•	0.1620 ± 0.0041•	0.1615 ± 0.0033•	0.1608 ± 0.0032•	0.1635 ± 0.0033•	0.1607 ± 0.0030•	0.2120 ± 0.0031○	0.2002 ± 0.0032
Mediamill	0.5315 ± 0.0021•	0.5351 ± 0.0022•	0.5353 ± 0.0021•	0.5354 ± 0.0022•	0.5356 ± 0.0022•	0.5367 ± 0.0023•	0.5315 ± 0.0020•	0.4682 ± 0.0043•	0.5427 ± 0.0019
	Ranking Loss ↓								
Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
plant	0.3292 ± 0.0166•	0.3109 ± 0.0193•	0.3261 ± 0.0174•	0.3276 ± 0.0239•	0.3250 ± 0.0189•	0.2594 ± 0.0221•	0.2123 ± 0.0240	0.2441 ± 0.0567	0.2099 ± 0.0156
msra	0.1938 ± 0.0103•	0.1677 ± 0.0105•	0.1742 ± 0.0118•	0.1663 ± 0.0116•	0.1642 ± 0.0109•	0.1611 ± 0.0105•	0.1531 ± 0.0061	0.2208 ± 0.0117•	0.1463 ± 0.0081
enron	0.2672 ± 0.0108•	0.2596 ± 0.0102•	0.2592 ± 0.0110•	0.2615 ± 0.0119•	0.2642 ± 0.0071•	0.1475 ± 0.0104•	0.1382 ± 0.0064•	0.1255 ± 0.0145	0.1209 ± 0.0093
llog	0.2567 ± 0.0306•	0.2610 ± 0.0276•	0.2637 ± 0.0306•	0.2585 ± 0.0271•	0.2625 ± 0.0295•	0.1538 ± 0.0283•	0.1661 ± 0.0151•	0.1345 ± 0.0385	0.1163 ± 0.0158
bibtext	0.1325 ± 0.0082•	0.1319 ± 0.0080•	0.1319 ± 0.0073•	0.1318 ± 0.0079•	0.1319 ± 0.0080•	0.0873 ± 0.0069	0.1020 ± 0.0081•	0.0421 ± 0.0071○	0.0834 ± 0.0062
eurlex-sm	0.2396 ± 0.0107•	0.2386 ± 0.0088•	0.2386 ± 0.0087•	0.2397 ± 0.0103•	0.1504 ± 0.0102•	0.0887 ± 0.0083•	0.0524 ± 0.0119	0.0979 ± 0.0125•	0.0481 ± 0.0022
bookmark	0.2563 ± 0.0072•	0.2577 ± 0.0066•	0.2581 ± 0.0060•	0.2571 ± 0.0057•	0.2602 ± 0.0071•	0.1710 ± 0.0068	0.2044 ± 0.0071•	0.1493 ± 0.0042	0.1642 ± 0.0032
corel5k	0.2096 ± 0.0044•	0.1937 ± 0.0047	0.1957 ± 0.0053	0.1943 ± 0.0040	0.1988 ± 0.0035•	0.1944 ± 0.0045	0.1964 ± 0.0070	0.4630 ± 0.0079•	0.1880 ± 0.0059
eurlex-dc	0.1841 ± 0.0110•	0.1838 ± 0.0095•	0.1839 ± 0.0096•	0.1943 ± 0.0126•	0.0941 ± 0.0111•	0.0442 ± 0.0112	0.0477 ± 0.0089	0.1486 ± 0.0053•	0.0390 ± 0.0026
espgame	0.2439 ± 0.0024•	0.2436 ± 0.0026•	0.2386 ± 0.0035•	0.2422 ± 0.0025•	0.2450 ± 0.0030•	0.2445 ± 0.0027•	0.1926 ± 0.0032	0.2858 ± 0.0035•	0.1932 ± 0.0025
Delicious	0.1755 ± 0.0026	0.1710 ± 0.0010•	0.1654 ± 0.0008	0.1656 ± 0.0026	0.1634 ± 0.0024	0.1681 ± 0.0023	0.1652 ± 0.0025	0.3608 ± 0.0043•	0.1651 ± 0.0024
Mediamill	0.0587 ± 0.0008	0.0589 ± 0.0007•	0.0599 ± 0.0006•	0.0590 ± 0.0008•	0.0598 ± 0.0010•	0.0585 ± 0.0009	0.0587 ± 0.0008	0.2272 ± 0.0120•	0.0576 ± 0.0009
	One Error ↓								
Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
plant	0.7099 ± 0.0337•	0.7089 ± 0.0350•	0.7058 ± 0.0374•	0.6986 ± 0.0335•	0.7058 ± 0.0365•	0.6547 ± 0.0393	0.6362 ± 0.0389	0.6649 ± 0.0820	0.6270 ± 0.0252
msra	0.1304 ± 0.0166•	0.1015 ± 0.0205•	0.0972 ± 0.0097•	0.0791 ± 0.0166•	0.0844 ± 0.0167•	0.0796 ± 0.0174	0.0646 ± 0.0160	0.0507 ± 0.0129	0.0576 ± 0.0142
enron	0.4454 ± 0.0234•	0.4471 ± 0.0209•	0.4483 ± 0.0223•	0.4318 ± 0.0187•	0.4295 ± 0.0221•	0.2930 ± 0.0236•	0.2524 ± 0.0308	0.3535 ± 0.2583	0.2462 ± 0.0215
llog	0.8526 ± 0.0302•	0.8451 ± 0.0334•	0.8476 ± 0.0353•	0.8418 ± 0.0428•	0.8460 ± 0.0388•	0.7371 ± 0.0423	0.7294 ± 0.0282	0.9900 ± 0.0096•	0.6898 ± 0.0352
bibtext	0.3950 ± 0.0134•	0.3996 ± 0.0088•	0.3952 ± 0.0087•	0.3978 ± 0.0102•	0.3998 ± 0.0084•	0.3728 ± 0.0082	0.3542 ± 0.0128	0.3522 ± 0.0094	0.3656 ± 0.0085
eurlex-sm	0.6600 ± 0.0253•	0.6576 ± 0.0239•	0.6576 ± 0.0239•	0.6592 ± 0.0253•	0.6586 ± 0.0259•	0.4059 ± 0.0267•	0.2626 ± 0.0234•	0.6326 ± 0.0186•	0.2306 ± 0.0105
bookmark	0.7156 ± 0.0051•	0.7188 ± 0.0098•	0.7234 ± 0.0062•	0.7200 ± 0.0072•	0.7186 ± 0.0061•	0.6650 ± 0.0095•	0.6642 ± 0.0066•	0.5800 ± 0.0142	0.6198 ± 0.0151
corel5k	0.6464 ± 0.0084	0.6532 ± 0.0093	0.6534 ± 0.0086	0.6448 ± 0.0100	0.6468 ± 0.0107	0.6454 ± 0.0098	0.6442 ± 0.0114	0.8620 ± 0.0110•	0.6370 ± 0.0158
eurlex-dc	0.6944 ± 0.0329•	0.6914 ± 0.0339•	0.6914 ± 0.0339•	0.6140 ± 0.0266•	0.5916 ± 0.0332•	0.4458 ± 0.0131•	0.3450 ± 0.0246•	0.7312 ± 0.0062•	0.3028 ± 0.0189
espgame	0.5622 ± 0.0166	0.5622 ± 0.0166	0.5624 ± 0.0173	0.5618 ± 0.0168	0.5622 ± 0.0166	0.5626 ± 0.0162	0.5624 ± 0.0161	0.9911 ± 0.0031•	0.5604 ± 0.0152
Delicious	0.3687 ± 0.0127•	0.3691 ± 0.0093•	0.3688 ± 0.0086•	0.3692 ± 0.0125•	0.3510 ± 0.0123	0.3542 ± 0.0122	0.3583 ± 0.0131	0.4069 ± 0.0162•	0.3386 ± 0.0135
Mediamill	0.1337 ± 0.0032	0.1322 ± 0.0040	0.1320 ± 0.0036	0.1317 ± 0.0045	0.1319 ± 0.0044	0.1315 ± 0.0037	0.1337 ± 0.0032	0.1374 ± 0.0035•	0.1307 ± 0.0035
	Precision@3 ↑								
Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
Delicious	0.5676 ± 0.0055•	0.5613 ± 0.0042•	0.5617 ± 0.0046•	0.5672 ± 0.0052•	0.5666 ± 0.0050•	0.5737 ± 0.0051	0.5774 ± 0.0051	0.4350 ± 0.0100•	0.5873 ± 0.0053
Mediamill	0.6682 ± 0.0036	0.6700 ± 0.0033	0.6697 ± 0.0031	0.6699 ± 0.0033	0.6697 ± 0.0034	0.6692 ± 0.0039	0.6682 ± 0.0035	0.7633 ± 0.0053○	0.6722 ± 0.0032
	nDCG@3 ↑								
Methods	ORI	PLST	CPLST	DMLR	FaIE	CMLL$_y$	MDDM	POP	CMLL
Delicious	0.5805 ± 0.0069•	0.5833 ± 0.0047	0.5836 ± 0.0050	0.5801 ± 0.0066•	0.5791 ± 0.0063•	0.5764 ± 0.0065•	0.5805 ± 0.0067•	0.4471 ± 0.0113•	0.5904 ± 0.0067
Mediamill	0.7508 ± 0.0036	0.7527 ± 0.0037	0.7525 ± 0.0034	0.7527 ± 0.0038	0.7525 ± 0.0039	0.7521 ± 0.0041	0.7508 ± 0.0036	0.8234 ± 0.0051○	0.7536 ± 0.0034

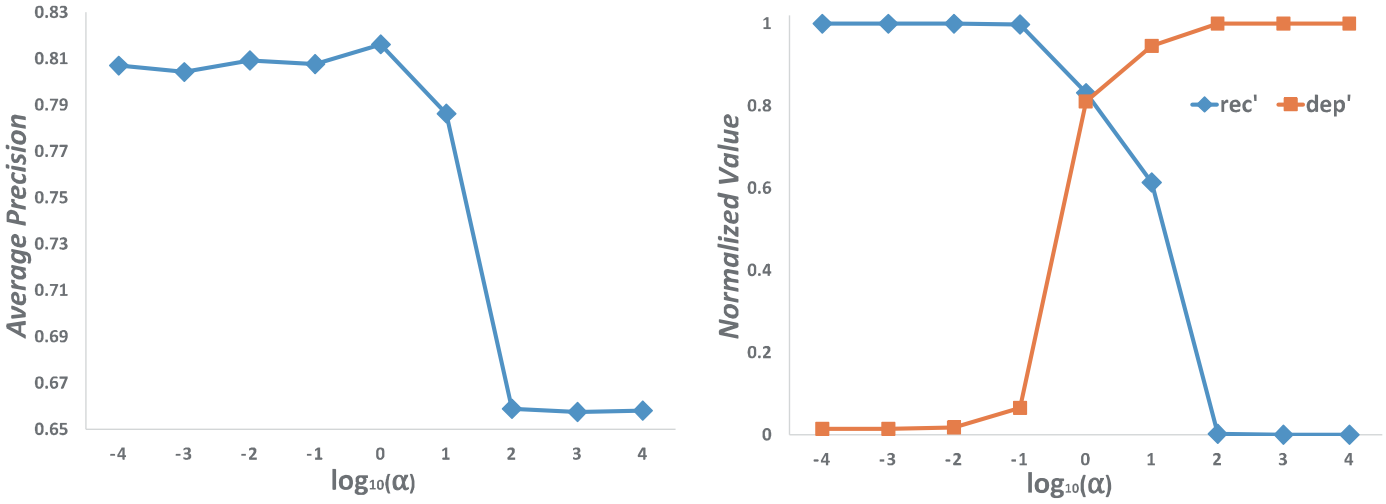
Table 3
Experimental results of k-CMLL with baselines.

	Average precision ↑							
Methods	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
plant	0.5894 ± 0.0385	0.5907 ± 0.0375	0.5985 ± 0.0383	0.5917 ± 0.0388	0.5928 ± 0.0387	0.6185 ± 0.0250	0.6277 ± 0.0280	0.6460 ± 0.0396
msra	0.8087 ± 0.0107	0.8090 ± 0.0105	0.8282 ± 0.0073	0.8231 ± 0.0090	0.8174 ± 0.0068	0.8197 ± 0.0111	0.8135 ± 0.0100	0.8209 ± 0.0103
enron	0.7001 ± 0.0180	0.7005 ± 0.0177	0.7001 ± 0.0180	0.6722 ± 0.0166	0.6930 ± 0.0155	0.7091 ± 0.0151	0.6856 ± 0.0535	0.7125 ± 0.0116
llog	0.4269 ± 0.0253•	0.4269 ± 0.0253•	0.4269 ± 0.0273•	0.4276 ± 0.0251•	0.4290 ± 0.0254•	0.4675 ± 0.0280	0.3770 ± 0.0130•	0.4755 ± 0.0207
bibtext	0.5957 ± 0.0051•	0.6220 ± 0.0069○	0.5948 ± 0.0064•	0.5969 ± 0.0058•	0.5970 ± 0.0059•	0.6030 ± 0.0060	0.6204 ± 0.0072○	0.6060 ± 0.0042
eurlex-sm	0.8011 ± 0.0157•	0.8006 ± 0.0170•	0.8011 ± 0.0157•	0.8112 ± 0.0158•	0.8010 ± 0.0157•	0.8060 ± 0.0161	0.7773 ± 0.0359•	0.8265 ± 0.0150
bookmark	0.4067 ± 0.0088•	0.4067 ± 0.0088•	0.4129 ± 0.0105•	0.4070 ± 0.0089•	0.4072 ± 0.0088•	0.4288 ± 0.0052•	0.3999 ± 0.0106•	0.4582 ± 0.0074
corel5k	0.3035 ± 0.0050•	0.3036 ± 0.0050•	0.3083 ± 0.0055•	0.3038 ± 0.0046•	0.3034 ± 0.0049•	0.3307 ± 0.0060	0.3169 ± 0.0019•	0.3321 ± 0.0084
eurlex-dc	0.7578 ± 0.0079•	0.7547 ± 0.0076•	0.7576 ± 0.0079•	0.7585 ± 0.0077•	0.7580 ± 0.0077•	0.7521 ± 0.0102•	0.7411 ± 0.0180•	0.7799 ± 0.0130
espgame	0.2298 ± 0.0065	0.2298 ± 0.0064	0.2298 ± 0.0065	0.2293 ± 0.0065	0.2293 ± 0.0063	0.2336 ± 0.0069	0.2286 ± 0.0022	0.2346 ± 0.0072
Delicious	0.3576 ± 0.0198•	0.3567 ± 0.0099•	0.3702 ± 0.0174•	0.3527 ± 0.0202•	0.3587 ± 0.0184•	0.3742 ± 0.0236	0.3596 ± 0.0025•	0.3863 ± 0.0041
Mediamill	0.7226 ± 0.1789	0.7213 ± 0.0282•	0.7696 ± 0.1142	0.7063 ± 0.0399•	0.7811 ± 0.1248	0.7857 ± 0.0864	0.6861 ± 0.0073•	0.7802 ± 0.0711
	micro-F1 ↑							
Methods	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
plant	0.3237 ± 0.0447	0.3268 ± 0.0432	0.3284 ± 0.0479	0.3261 ± 0.0468	0.3280 ± 0.0432	0.3430 ± 0.0454	0.3680 ± 0.034	0.3526 ± 0.0294
msra	0.6711 ± 0.0083•	0.6712 ± 0.0084•	0.6837 ± 0.0081•	0.6840 ± 0.0088•	0.6831 ± 0.0074•	0.6889 ± 0.0114	0.6708 ± 0.0084•	0.7004 ± 0.0114
enron	0.5849 ± 0.0062•	0.5852 ± 0.0064•	0.5849 ± 0.0062•	0.5857 ± 0.0060•	0.5851 ± 0.0060•	0.6037 ± 0.0119•	0.6512 ± 0.0310	0.6582 ± 0.0115
llog	0.1512 ± 0.0183	0.1512 ± 0.0183	0.1512 ± 0.0182	0.1513 ± 0.0185	0.1612 ± 0.0184	0.1566 ± 0.0247	0.2829 ± 0.0199○	0.1733 ± 0.0150
bibtext	0.3512 ± 0.0075•	0.3680 ± 0.0104•	0.3489 ± 0.0064•	0.3520 ± 0.0055•	0.3521 ± 0.0055•	0.3985 ± 0.0069	0.3996 ± 0.0081	0.4069 ± 0.0075
eurlex-sm	0.5580 ± 0.0205•	0.5578 ± 0.0205•	0.5580 ± 0.0205•	0.5586 ± 0.0204•	0.5581 ± 0.0205•	0.6564 ± 0.0159	0.6061 ± 0.0106•	0.6555 ± 0.0164
bookmark	0.2019 ± 0.0057•	0.2019 ± 0.0057•	0.2268 ± 0.0075•	0.2020 ± 0.0056•	0.2027 ± 0.0057•	0.2106 ± 0.0063•	0.2657 ± 0.0103	0.2378 ± 0.0053
corel5k	0.1146 ± 0.0056•	0.1146 ± 0.0056•	0.1166 ± 0.0050•	0.1149 ± 0.0054•	0.1147 ± 0.0057•	0.1431 ± 0.0012•	0.1685 ± 0.0057	0.1702 ± 0.0064
eurlex-dc	0.4648 ± 0.0160•	0.4647 ± 0.0160•	0.4663 ± 0.0161•	0.4659 ± 0.0161•	0.4651 ± 0.0161•	0.5489 ± 0.0199	0.4847 ± 0.0015•	0.5554 ± 0.0181
espgame	0.1039 ± 0.0049	0.1040 ± 0.0047	0.1039 ± 0.0049	0.1040 ± 0.0048	0.1039 ± 0.0049	0.1062 ± 0.0064	0.1178 ± 0.0131○	0.1063 ± 0.0078
Delicious	0.1795 ± 0.0133•	0.1780 ± 0.0073•	0.1876 ± 0.0162•	0.1851 ± 0.0199•	0.2050 ± 0.0220•	0.1900 ± 0.0034•	0.3416 ± 0.0019○	0.2361 ± 0.0029•
Mediamill	0.5405 ± 0.0983•	0.5415 ± 0.0382•	0.5611 ± 0.0675	0.5331 ± 0.0263•	0.5429 ± 0.1306	0.5913 ± 0.0247	0.5556 ± 0.0049•	0.5927 ± 0.0199
	Ranking Loss ↓							
Methods	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
plant	0.1771 ± 0.0312	0.1756 ± 0.0300	0.1701 ± 0.0257	0.1761 ± 0.0314	0.1756 ± 0.0312	0.1666 ± 0.0232	0.1578 ± 0.0194	0.1495 ± 0.0294
msra	0.1435 ± 0.0075•	0.1433 ± 0.0073•	0.1176 ± 0.0058○	0.1234 ± 0.0066	0.1182 ± 0.0058	0.1275 ± 0.0087	0.1113 ± 0.0073○	0.1289 ± 0.0088
enron	0.0973 ± 0.0138•	0.0969 ± 0.0138•	0.0973 ± 0.0138•	0.1012 ± 0.0129•	0.1000 ± 0.0128•	0.1011 ± 0.0120•	0.0833 ± 0.0139	0.0816 ± 0.0037
llog	0.1758 ± 0.0293•	0.1757 ± 0.0292•	0.1787 ± 0.0301•	0.1764 ± 0.0297•	0.1762 ± 0.0279•	0.1467 ± 0.0170	0.1249 ± 0.0141	0.1362 ± 0.0149
bibtext	0.0939 ± 0.0109	0.0980 ± 0.0090•	0.0933 ± 0.0105	0.0936 ± 0.0108	0.0934 ± 0.0107	0.0819 ± 0.0083	0.0565 ± 0.0721○	0.0798 ± 0.0091
eurlex-sm	0.0221 ± 0.0084	0.0266 ± 0.0078	0.0219 ± 0.0085	0.0219 ± 0.0084	0.0221 ± 0.0086	0.0210 ± 0.0049	0.0240 ± 0.1171	0.0202 ± 0.0061
bookmark	0.1604 ± 0.0071•	0.1604 ± 0.0071•	0.1662 ± 0.0088•	0.1603 ± 0.0072•	0.1509 ± 0.0075	0.1584 ± 0.0060•	0.1527 ± 0.0279	0.1447 ± 0.0050
corel5k	0.1941 ± 0.0068•	0.1440 ± 0.0068	0.1566 ± 0.0078	0.1741 ± 0.0070•	0.1640 ± 0.0068	0.1937 ± 0.0052•	0.1321 ± 0.0240	0.1523 ± 0.0119
eurlex-dc	0.0416 ± 0.0042	0.0374 ± 0.0053	0.0409 ± 0.0041	0.0417 ± 0.0039	0.0417 ± 0.0042	0.0361 ± 0.0084	0.0451 ± 0.0418	0.0357 ± 0.0046
espgame	0.0253 ± 0.0065	0.0251 ± 0.0065	0.0256 ± 0.0067	0.0255 ± 0.0065	0.0256 ± 0.0066	0.0249 ± 0.0070	0.0206 ± 0.0123	0.0232 ± 0.0071
Delicious	0.1746 ± 0.0468	0.1630 ± 0.0135	0.1701 ± 0.0371	0.2053 ± 0.0531	0.1684 ± 0.0560	0.1689 ± 0.0373	0.1232 ± 0.0012	0.1599 ± 0.0318
Mediamill	0.0671 ± 0.0027•	0.0644 ± 0.0051	0.0634 ± 0.0014•	0.0663 ± 0.0054	0.0624 ± 0.0056	0.0666 ± 0.0024•	0.0659 ± 0.0034•	0.0599 ± 0.0021
	One Error ↓							
Methods	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
plant	0.5857 ± 0.0457	0.5836 ± 0.0435	0.5826 ± 0.0518	0.5867 ± 0.0463	0.5847 ± 0.0457	0.5458 ± 0.0290	0.5581 ± 0.0473	0.5407 ± 0.0510
msra	0.0653 ± 0.0198	0.0642 ± 0.0180	0.0525 ± 0.0136	0.0553 ± 0.0057	0.0583 ± 0.0130	0.0637 ± 0.0166	0.0617 ± 0.0267	0.0507 ± 0.0099
enron	0.2290 ± 0.0165	0.2278 ± 0.0140	0.2290 ± 0.0165	0.2454 ± 0.0205•	0.2331 ± 0.0187	0.2213 ± 0.0196	0.2562 ± 0.0571	0.2091 ± 0.0169
llog	0.7145 ± 0.0272•	0.7145 ± 0.0272•	0.7153 ± 0.0310•	0.7145 ± 0.0272•	0.7154 ± 0.0316•	0.6756 ± 0.0328	0.7642 ± 0.0149•	0.6625 ± 0.0374
bibtext	0.3506 ± 0.0043•	0.3496 ± 0.0166	0.3442 ± 0.0064	0.3492 ± 0.0044•	0.3392 ± 0.0044	0.3416 ± 0.0050	0.3164 ± 0.0092	0.3368 ± 0.0090
eurlex-sm	0.1754 ± 0.0198•	0.1788 ± 0.0204•	0.1754 ± 0.0198•	0.1756 ± 0.0200•	0.1756 ± 0.0200•	0.1514 ± 0.0282	0.1293 ± 0.0116	0.1261 ± 0.0192
bookmark	0.6008 ± 0.0077•	0.6008 ± 0.0077•	0.5928 ± 0.0141•	0.6006 ± 0.0078•	0.6000 ± 0.0080•	0.5724 ± 0.0084	0.5430 ± 0.0097	0.5616 ± 0.0090
corel5k	0.6274 ± 0.0111•	0.6268 ± 0.0112•	0.6286 ± 0.0087•	0.6282 ± 0.0110•	0.6278 ± 0.0112•	0.6138 ± 0.0138•	0.5876 ± 0.0142	0.5880 ± 0.0169
eurlex-dc	0.3098 ± 0.0112•	0.3140 ± 0.0095•	0.3096 ± 0.0115•	0.3088 ± 0.0110•	0.3096 ± 0.0113•	0.2838 ± 0.0137	0.3012 ± 0.0208•	0.2800 ± 0.0163
espgame	0.5458 ± 0.0153	0.5462 ± 0.0157	0.5460 ± 0.0134	0.5464 ± 0.0140	0.5456 ± 0.0148	0.5363 ± 0.0123	0.5325 ± 0.0425	0.5306 ± 0.0091
Delicious	0.3446 ± 0.0640	0.3320 ± 0.0321	0.3236 ± 0.0515	0.3257 ± 0.0464	0.3336 ± 0.0258	0.3345 ± 0.1174	0.3497 ± 0.0058	0.3260 ± 0.0617
Mediamill	0.1567 ± 0.0045•	0.1386 ± 0.0029	0.1439 ± 0.0099•	0.1510 ± 0.0039•	0.1457 ± 0.0020•	0.1412 ± 0.0072	0.1507 ± 0.0131	0.1384 ± 0.0040
	Precision@3 ↑							
Methods	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
Delicious	0.5739 ± 0.0045•	0.5814 ± 0.0029•	0.5882 ± 0.0049•	0.5859 ± 0.0081•	0.5887 ± 0.0026•	0.5800 ± 0.0033•	0.5935 ± 0.0059•	0.6090 ± 0.0022
Mediamill	0.6480 ± 0.0062•	0.6576 ± 0.0080	0.6599 ± 0.0092	0.6400 ± 0.0032•	0.6582 ± 0.0080	0.6605 ± 0.0095	0.6492 ± 0.0038•	0.6622 ± 0.0072
	nDCG@3 ↑							
	k-ORI	k-CPLST	k-DMLR	k-FaIE	k-CMLL$_{\boldsymbol{y}}$	k-MDDM	C2AE	k-CMLL
Delicious	0.5911 ± 0.0087	0.5901 ± 0.0034•	0.5970 ± 0.0042	0.5944 ± 0.0097	0.5950 ± 0.0022•	0.5952 ± 0.0058	0.6073 ± 0.0060	0.6098 ± 0.0075
Mediamill	0.7360 ± 0.0098	0.7361 ± 0.0039	0.7349 ± 0.0066	0.7342 ± 0.0043•	0.7489 ± 0.0021	0.7479 ± 0.0077	0.7285 ± 0.0034•	0.7496 ± 0.0034

we can find the solution of $\boldsymbol{V}$ and calculate corresponding dep_{min} and rec_{max}. To obtain a comprehensive understanding, we normalize the value of two terms by:

$$rec' = \frac{rec - rec_{min}}{rec_{max} - rec_{min}}, \quad dep' = \frac{dep - dep_{min}}{dep_{max} - dep_{min}}.$$

The experimental results on msra with average precision are showed in Fig. 1 as an example. The curve in Fig. 1(a) indicates that setting α too big or too small will both result in bad performance. And it seems that an unreasonable big α suffers more, which indicates that an encoder with good recovery ability is very important for CMLL. Fig. 1(b) provides the explanation for the curve trend in Fig. 1(a). For example, the curve of average precision drops sharply when $log_{10}(\alpha)$ changes from 1 to 2. And the reason is that dep' is already very close to the upper bound when $\alpha = 10$. As α further rises to 100, the increment of dep' is very limited while rec' decreases obviously. This suggests that α do not need to be increased when dep' is close to its maximization, otherwise the decrement of rec' will result in a decline of the performance. Thus we can utilize such plot to guide the tuning process of α. To sum up, aiming to get better performance, we should consider an appropriate trade-off between dependence term and recovery term.



(a) Average Precision (b) dep' and rec'

Fig. 1. Average precision changes as parameter α varies on msra.

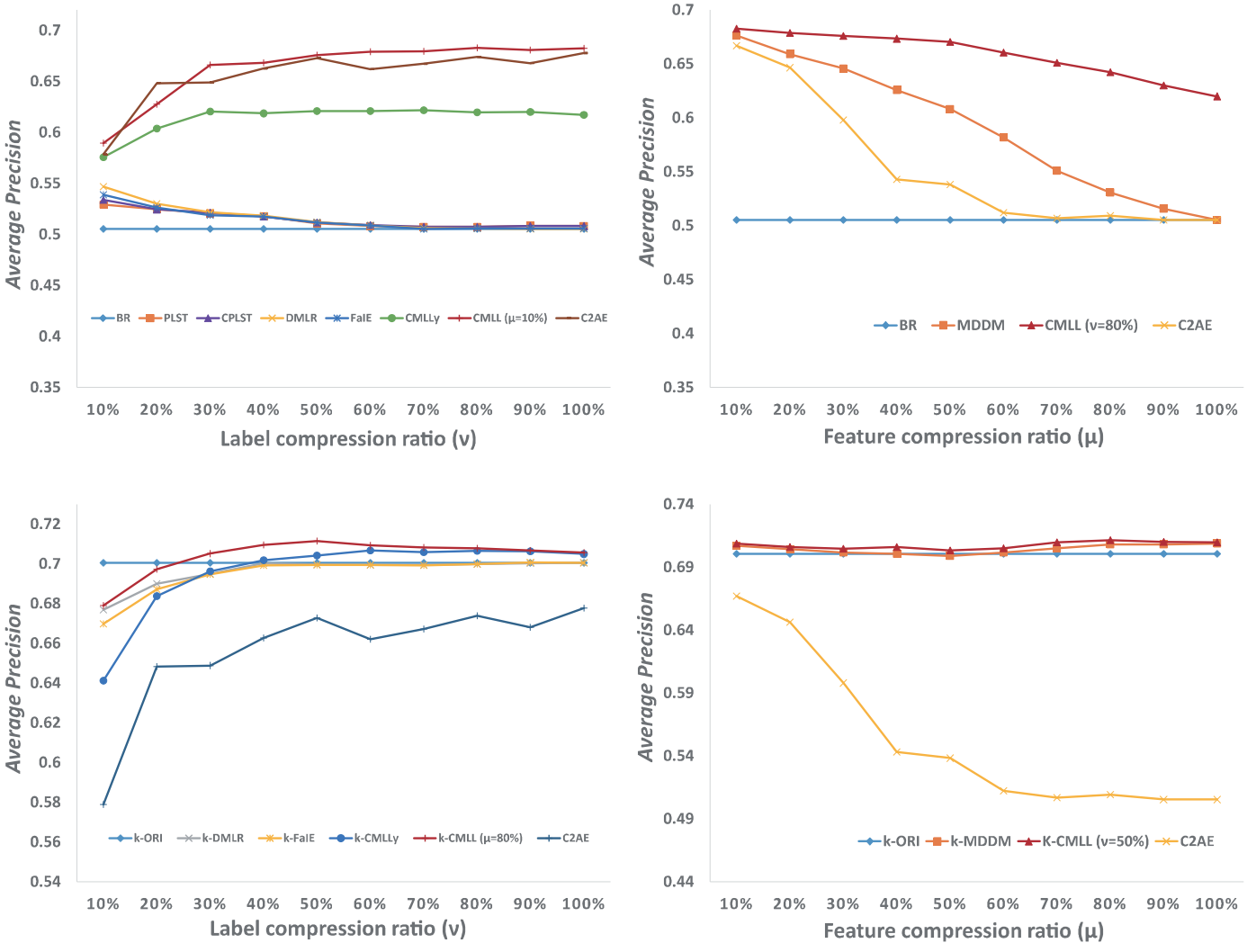
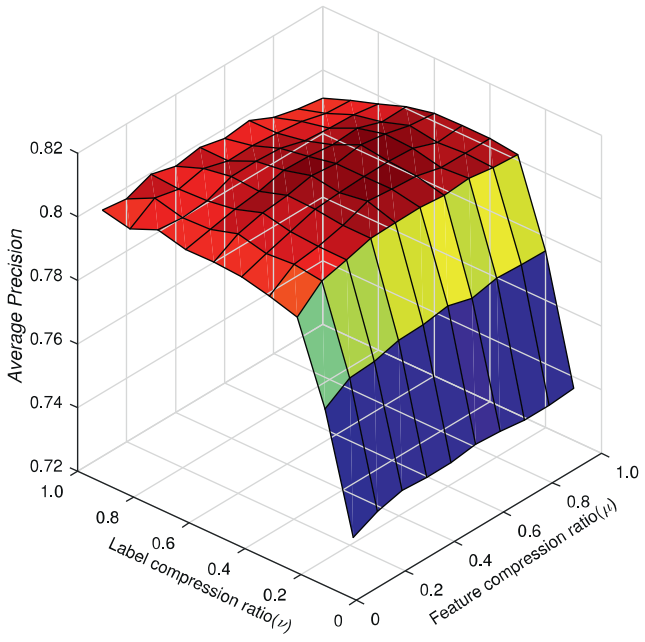
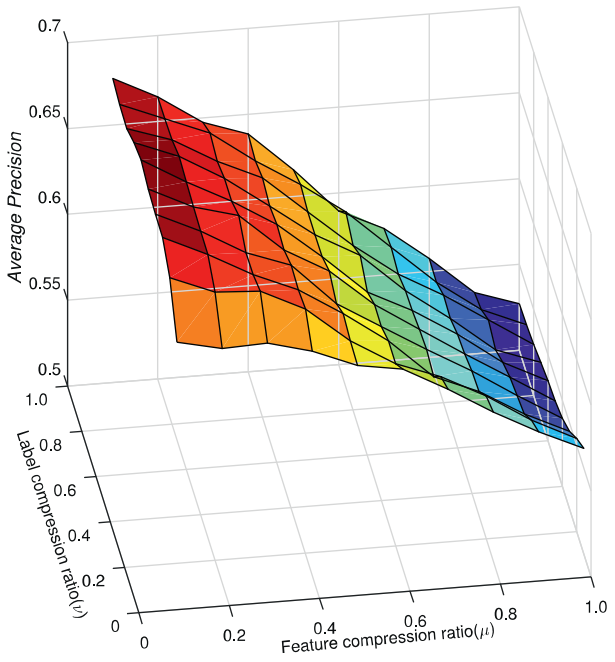


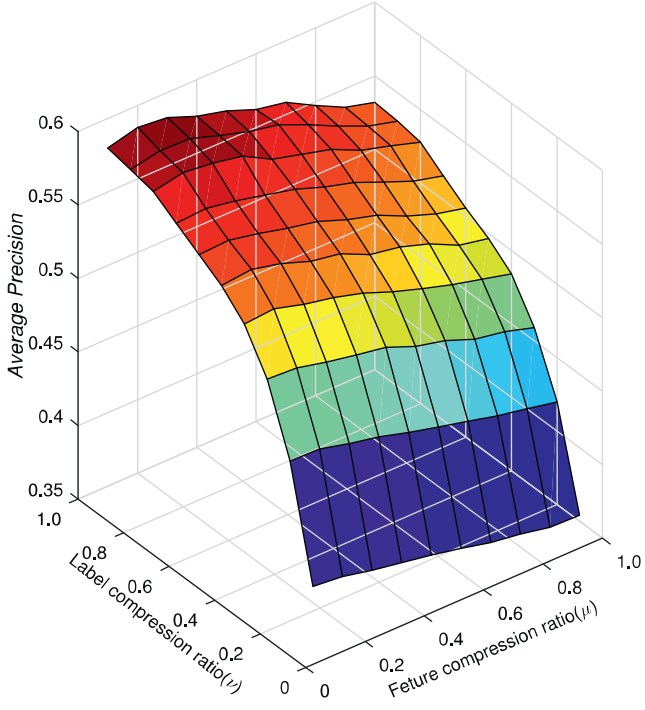
Fig. 2. The average precision curve of moving the compression ratio on enron.



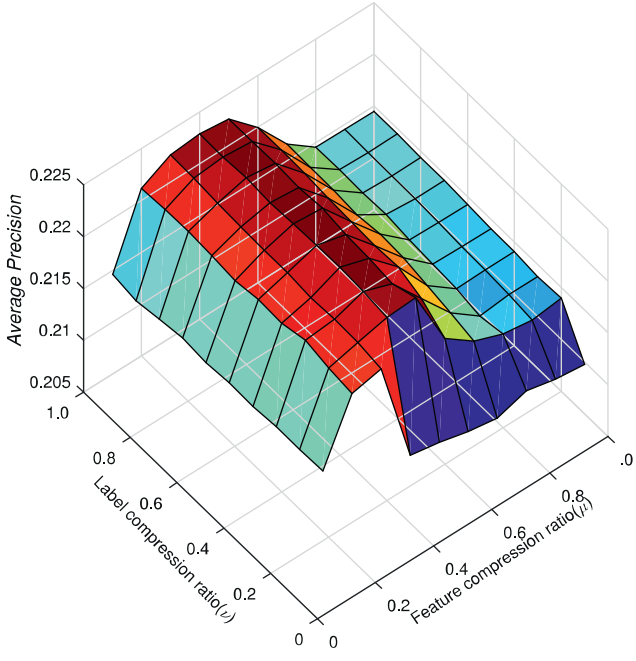
(a) msra



(b) enron



(c) bibtex



(d) esmpgame

Fig. 3. The spatial graphs of CMLL over various μ and ν.

5.5. Analysis on the compression ratio

We investigate how the compression ratio influences the performance by fixing one space changeless and gradually moving the compression ration of the other space from 100% to 10%, and due to page limitation, we only display the curve of average precision on enron in Fig. 2. We can observe that no matter how the compression ratio changes, CMLL usually achieves a better or comparable performance.

The reason for the superiority of CMLL is that it follows the spirit of CL. CMLL links the embedding process of the label space and the feature space to each other and guides each process by another well-disposed space. Instead, most other embedding methods either focus on the embedding of just one space, or guides the embedding process by original problematic space. Therefore, CMLL performs well especially in noisy, redundant and sparse datasets. However, the embedding may bring the loss of information when we compress the dense or non-redundant datasets into a very low dimension.

In reality, traversing every possible pair (μ, ν) to lock the best one is unaffordable. Here we give an empirical method for that. We draw the spatial graphs of CMLL for collected datasets over various μ and ν. And the spatial graphs of some datasets on average precision are displayed in Fig. 3 as examples.

It can be seen that, for different ν on each dataset, the general trends of CMLL over various μ are almost the same. Providing this empirical observation gives us a heuristic way to select (μ, ν): we first conduct CMLL with a random fixed ν over various μ, and lock the best μ^* in such situation. Then we search the best μ^* with fixing ν. Finally a near optimal ratio pair (μ^*, ν^*) is achieved. One can also try some different starting ν to make the searching process more precise. In practice, we find that the ratio pair searched by this empirical method can achieve comparable performance to the real best one in most cases. Especially, we find that μ may have little influence on the performance in some datasets. In other words, a very small compression ratio can perform well in CMLL, which proves the existence of redundancy and shows the superiority of CMLL to reduce the computational and space complexities.

6. Conclusion

In this paper, we provided a different insight into efficiently capturing the inherent high-order correlation between features and labels, named compact learning. We analyzed its rationality and necessity in the situation, where the feature space suffers from redundancy or noise, and meanwhile, the label space is deteriorated by noise or sparsity - frequent occurrences in MLC. Following the spirit of compact learning, a simple yet effective method termed CMLL that is compatible with flexible multi-label classifiers was proposed. By conducting the embedding process of the features and the labels seamlessly, CMLL achieved a more compact representation for both the spaces. We demonstrated through experiments that CMLL can result in significant improvements for MLC.

As an initial effort towards compact learning, there are several potential ways that the current CMLL can be further improved: (a) Except the linear embedding or its kernel version, other encoding and decoding strategies, such as autoencoders and its extension, are worthwhile to be investigated; (b) Inspired by the manifold learning that the local topological structure can be shared between the feature manifold and the label manifold [34], the structure information could be utilized for CMLL; (c) CMLL provides another possible solution to some weakly supervised learning problem, e.g., the missing label [35] or noisy label [36].

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the National Key Research and Development Program of China (No. 2017YFB1002801), and the National Natural Science Foundation of China (62076063).

References

[1] M. Zhang, Z. Zhou, A review on multi-label learning algorithms, IEEE Trans. Knowl. Data Eng. 26 (8) (2014) 1819–1837.
[2] F. Liu, T. Xiang, T.M. Hospedales, W. Yang, C. Sun, Semantic regularisation for recurrent image annotation, in: Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, 2017, pp. 2872–2880.
[3] Y. Liu, K. Wen, Q. Gao, X. Gao, F. Nie, SVM based multi-label learning with missing labels for image annotation, Pattern Recognit. 78 (2018) 307–317.
[4] K. Zhao, W. Chu, F.D. la Torre, J.F. Cohn, H. Zhang, Joint patch and multi-label learning for facial action unit and holistic expression recognition, IEEE Trans. Image Process. 25 (8) (2016) 3931–3946.
[5] N. Zhuang, Y. Yan, S. Chen, H. Wang, C. Shen, Multi-label learning based deep transfer neural network for facial attribute classification, Pattern Recognit. 80 (2018) 225–240.
[6] J. Liu, W. Chang, Y. Wu, Y. Yang, Deep learning for extreme multi-label text classification, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 115–124.
[7] X. Zhang, R. Henao, Z. Gan, Y. Li, L. Carin, Multi-label learning from medical plain text with convolutional residual models, arXiv preprint arXiv:1801.05062(2018).
[8] F. Tai, H. Lin, Multi-label classification with principal label space transformation, Neurocomputing 24 (9) (2012) 2508–2542.
[9] B. Wang, L. Chen, W. Sun, K. Qin, K. Li, H. Zhou, Ranking-based autoencoder for extreme multi-label classification, arXiv preprint arXiv:1904.05937(2019).
[10] D. Hsu, S. Kakade, J. Langford, T. Zhang, Multi-label prediction via compressed sensing, in: Advances in Neural Information Processing Systems 22, Vancouver, Canada, 2009, pp. 772–780.
[11] P. Rai, C. Hu, R. Henao, L. Carin, Large-scale Bayesian multi-label learning via topic-based label embeddings, in: Advances in Neural Information Processing Systems 28, Montreal, Canada, 2015, pp. 3222–3230.
[12] C. Yeh, W. Wu, W. Ko, Y.F. Wang, Learning deep latent space for multi-label classification, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, San Francisco, CA, 2017, pp. 2838–2844.
[13] V. Kumar, A.K. Pujari, V. Padmanabhan, V.R. Kagita, Group preserving label embedding for multi-label classification, Pattern Recognit. 90 (2019) 23–34.
[14] J. Wicker, B. Pfahringer, S. Kramer, Multi-label classification using boolean matrix decomposition, in: ACM Symposium on Applied Computing, Trento, Italy, 2012, pp. 179–186.
[15] X. Li, Y. Guo, Multi-label classification with feature-aware non-linear label space transformation, in: Proceedings of the 24th International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 2015, pp. 3635–3642.
[16] Y. Chen, H. Lin, Feature-aware label space dimension reduction for multi-label classification, in: Advances in Neural Information Processing Systems 25, Lake Tahoe, NV, 2012, pp. 1529–1537.
[17] Z. Lin, G. Ding, M. Hu, J. Wang, Multi-label classification via feature-aware implicit label space encoding, in: Proceedings of the 31th International Conference on Machine Learning, Beijing, China, 2014, pp. 325–333.
[18] D.R. Hardoon, S. Szedmak, J. Shawe-Taylor, Canonical correlation analysis: an overview with application to learning methods, Neurocomputing 16 (12) (2003) 2639–2664.
[19] Y. Zhang, Z. Zhou, Multi-label dimensionality reduction via dependence maximization, ACM Trans. Knowl. Discov. Data 4 (3) (2010) 14.
[20] L. Sun, S. Ji, J. Ye, Canonical correlation analysis for multilabel classification: a least-squares formulation, extensions, and analysis, IEEE Trans. Pattern Anal. Mach. Intell. 33 (1) (2010) 194–200.
[21] Y. Zhang, J. Schneider, Multi-label output codes using canonical correlation analysis, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, 2011, pp. 873–882.
[22] J. Zhang, M. Fang, H. Wang, X. Li, Dependence maximization based label space dimension reduction for multi-label classification, Eng. Appl. Artif. Intell. 45 (2015) 453–463.
[23] A. Joly, P. Geurts, L. Wehenkel, Random forests with random projections of the output space for high dimensional multi-label classification, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Nancy, France, 2014, pp. 607–622.
[24] P. Mineiro, N. Karampatziakis, Fast label embeddings via randomized linear algebra, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Porto, Portugal, 2015, pp. 37–51.

[25] L. Jing, L. Yang, J. Yu, M.K. Ng, Semi-supervised low-rank mapping learning for multi-label classification, in: Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, 2015, pp. 1483–1491.
[26] K. Bhatia, H. Jain, P. Kar, M. Varma, P. Jain, Sparse local embeddings for extreme multi-label classification, in: Advances in Neural Information Processing Systems 28, Montreal, Canada, 2015, pp. 730–738.
[27] L. Jian, J. Li, K. Shu, H. Liu, Multi-label informed feature selection, in: Proceedings of the 25th International Joint Conference on Artificial Intelligence, New York, NY, 2016, pp. 1627–1633.
[28] J. Wicker, A. Tyukin, S. Kramer, A nonlinear label compression and transformation method for multi-label classification using autoencoders, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Auckland, New Zealand, 2016, pp. 328–340.
[29] X. Shen, W. Liu, I.W. Tsang, Q. Sun, Y. Ong, Compact multi-label learning, in: Proceedings of 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, 2018, pp. 4066–4073.
[30] A. Gretton, O. Bousquet, A. Smola, B. Schölkopf, Measuring statistical dependence with hilbert-schmidt norms, in: Proceedings of the 16th International Conference on Algorithmic Learning Theory, Singapore, 2005, pp. 63–77.
[31] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, second ed., Springer, 2004.
[32] R.B. Lehoucq, D.C. Sorensen, Deflation techniques for an implicitly restarted Arnoldi iteration, SIAM J. Matrix Anal. Appl. 17 (4) (1996) 789–821, doi:10.1137/S0895479895281484.
[33] T. Wei, Y. Li, Learning compact model for large-scale multi-label data, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, HI, 33, 2019, pp. 5385–5392.
[34] P. Hou, X. Geng, M. Zhang, Multi-label manifold learning, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, Phoenix, Arizona, 2016, pp. 1680–1686.
[35] J. Lv, N. Xu, R. Zheng, X. Geng, Weakly supervised multi-label learning via label enhancement., in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macao, China, 2019, pp. 3101–3107.
[36] G. Patrini, A. Rozza, A.K. Menon, R. Nock, L. Qu, Making deep neural networks robust to label noise: aloss correction approach, in: Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, 2017, pp. 1944–1952.

Jiaqi Lv received the B.Sc degree in soft engineering from Southeast University, China in 2015. She is currently a doctoral student in the School of Computer Science and Engineering at Southeast University, China. Her research interests mainly include pattern recognition and machine learning.

Tianran Wu received the B.Sc. and M.Sc degrees in computer science from Southeast University, China in 2017 and 2020, respectively. She is currently working at Microsoft company.

Chenglun Peng received the B.Sc and M.Sc degrees in computer science from Southeast University, China in 2015 and 2018, respectively. He is currently an algorithm engineer at the Meituan-Dianping Group.

Yunpeng Liu received the B.Sc. degree in computer science from Southeast University, China in 2018. He is currently a postgraduate student in the School of Computer Science and Engineering at Southeast University, China. His research interests mainly include pattern recognition and machine learning.

Ning Xu received the B.Sc and M.Sc degrees from University of Science and Technology of China and Chinese Academy of Sciences China, respectively, and the PhD degree from Southeast University, China. He is now an assistant professor in the School of Computer Science and Engineering at Southeast University, China. His research interests mainly include pattern recognition and machine learning.

Xin Geng is currently a professor and the director of the PALM lab (http://palm.seu.edu.cn/) of Southeast University, China. He received the B.Sc. (2001) and M.Sc.(2004) degrees in computer science from Nanjing University, China, and the Ph.D. (2008) degree in computer science from Deakin University, Australia. His research interests include pattern recognition, machine learning, and computer vision. He has published more than 50 refereed papers in these areas, including those published in prestigious journals and top international conferences.