Discovering Latent Class Labels for Multi-Label Learning

Jun Huang[1,2,*] , **Linchuan Xu**[1] , **Jing Wang**[3] , **Lei Feng**[4,*] and **Kenji Yamanishi**[1]

[1]Graduate School of Information Science and Technology, The University of Tokyo.
[2]School of Computer Science and Technology, Anhui University of Technology.
[3]School of Computing and Mathematical Sciences, University of Greenwich.
[4]School of Computer Science and Engineering, Nanyang Technological University.
{jun_huang,linchuan_xu,jing_wang,yamanishi}@mist.i.u-tokyo.ac.jp, feng0093@e.ntu.edu.sg

Abstract

Existing multi-label learning (MLL) approaches mainly assume all the labels are observed and construct classification models with a fixed set of target labels (*known labels*). However, in some real applications, multiple *latent labels* may exist outside this set and hidden in the data, especially for large-scale data sets. Discovering and exploring the latent labels hidden in the data may not only find interesting knowledge but also help us to build a more robust learning model. In this paper, a novel approach named DLCL (i.e., *D*iscovering *L*atent *C*lass *L*abels for MLL) is proposed which can not only discover the latent labels in the training data but also predict new instances with the latent and known labels simultaneously. Extensive experiments show a competitive performance of DLCL against other state-of-the-art MLL approaches.

1 Introduction

MLL [Zhang and Zhou, 2014; Gibaja and Ventura, 2015] deals with data examples with multiple class labels simultaneously. Many well-known approaches have been proposed to solve different problems of MLL, such as partial MLL [Wang *et al.*, 2019; Xu *et al.*, 2019], extreme MLL [Wei *et al.*, 2019; Jain *et al.*, 2019; Chen *et al.*, 2019], missing labels [Yu *et al.*, 2014; Yang *et al.*, 2016; Tan *et al.*, 2018], and multi-view MLL [Xing *et al.*, 2018; Wu *et al.*, 2019]. It is noted that existing approaches mainly assume that all the class labels are observed in advance. However, in some applications, some *latent labels* might be completely unobserved and hidden in the data, and below is a summary of two possible reasons:

1. **Labeling Cost**. In the big data era, it is difficult to provide a complete label set for a data, especially a large-scale data with an extreme number of labels. Labelling efforts usually focus on the given set of target labels, while labels outside this set will not be considered.
2. **Limitation of knowledge**. For example, in medical diagnosis, possible diseases will be predicted according to the patient's symptoms by the model constructed on the history data [Zhang *et al.*, 2018]. However, complicated diseases may definitely exist but have not been discovered due to the limitation of human's knowledge.

Discovering and exploring the class labels hidden in the data may not only find interesting knowledge but also improve the performance on known labels [Pham *et al.*, 2015; Zhu *et al.*, 2017a]. Therefore, it is important to construct a robust model for MLL which can not only discover the latent labels but also could predict new data examples with both the known and latent labels simultaneously.

Approaches have been proposed to solve data classification with an unfixed label set, such as online learning for single-label learning (SLL) [Kuzborskij *et al.*, 2013; Nguyen *et al.*, 2016; Zhu *et al.*, 2017b] and MLL [Hua and Qi, 2008; Xioufis *et al.*, 2011; Zhu *et al.*, 2018; Zhang *et al.*, 2020], However, in the settings of online learning, novel labels are only induced by new instances, and they will not be discovered if they are hidden in the existing training data. MIMLNC [Pham *et al.*, 2015] and DMNL [Zhu *et al.*, 2017a] are two highly related studies on discovering new label(s) for multi-instance multi-label learning (MIMLL). MIMLNC is a probabilistic model to identify novel instances for MIMLL, but it assumes that all novel instances belong to a single new label. DMNL tries to discover multiple novel labels for MIMLL. It assumes that there are k novel labels, and the problem is formulated as a non-negative orthogonal constrained optimization problem which has a bag-dependent loss term and a bag-independent clustering regularization term. However, these two approaches cannot be applied to general single-instance MLL problems directly.

In this paper, a novel approach named DLCL is proposed for MLL which can not only discover the latent labels in the training data but also predict new instances with both the latent and known labels. On the one hand, we try to improve the performance of known labels by exploring the information provided by the discovered latent labels. On the other hand, we exploit the knowledge of known labels to guide the discovery of latent labels.

2 Proposed Method

Let $\mathcal{X}\in\mathbb{R}^d$ be the feature space, and $\hat{\mathcal{Y}}=\mathcal{Y}\cup\bar{\mathcal{Y}}$ be the label set, where $\mathcal{Y}=\{y_1, ..., y_q\}$ and $\bar{\mathcal{Y}}=\{y_{q+1}, ..., y_{q+k}\}$ indicate the

*corresponding authors

observed and latent labels respectively. $\mathbf{X}\in\mathbb{R}^{n\times d}$ indicates the data matrix, and $\hat{\mathbf{Y}}=[\mathbf{Y},\bar{\mathbf{Y}}]\in\{0,1\}^{n\times l}$ represents the label matrix, where $l=q+k$. $\mathbf{Y}\in\{0,1\}^{n\times q}$ and $\bar{\mathbf{Y}}\in\{0,1\}^{n\times k}$ indicates the observed and latent label matrices respectively. If $\mathbf{x}_i$ belongs to y_j, then $y_{ij}=1$; otherwise $y_{ij}=0$.

2.1 Discovering Latent Class Labels

Problem Definition (Discovering Latent Class Labels for MLL). *Given an MLL data set with q known labels, the problem of discovering latent class labels for MLL is to detect previously unknown labels (e.g., k latent labels) for each instance in the training set, and build a model which can predict unseen data examples with both the known and latent labels.*

The aim is to construct a MLL model $h:\mathcal{X}\rightarrow 2^{\hat{\mathcal{Y}}}$, but $\bar{\mathbf{Y}}$ is unknown at first. Motivated by previous work on clustering based matrix factorization [Hu and Chen, 2019], we try to learn an approximate representation $\mathbf{U}\in\{0,1\}^{n\times l}$ for the completed label matrix $\hat{\mathbf{Y}}=[\mathbf{Y},\bar{\mathbf{Y}}]$. Since $\mathbf{Y}$ is known in advance, and thus the results of the first q columns of $\mathbf{U}$ should be consistent with that of known class labels. Therefore, the optimization problem can be defined as

$$\min_{\mathbf{U},\mathbf{V}} \frac{\lambda_1}{2}\|\mathbf{X}-\mathbf{U}\mathbf{V}\|_F^2+\frac{\lambda_2}{2}\|\mathbf{U}\mathbf{P}-\mathbf{Y}\|_F^2 \quad (1)$$
$$s.t.\ \mathbf{U}\in\{0,1\}^{n\times l}$$

where $\mathbf{V}\in\mathcal{R}^{l\times d}$ is the coefficient, and $\mathbf{P}\in\mathcal{R}^{l\times q}$ is a projection matrix which is composed of the first q columns of an $l\times l$ identity matrix. Once $\mathbf{U}$ is obtained, we can initialize $\bar{\mathbf{Y}}$ according to $\mathbf{U}$, i.e., $\mathbf{U}_{(:,(q+1):l)}$. Then, we can construct a MLL model with a squared loss as

$$\min_{\mathbf{W},\mathbf{U},\mathbf{V}} \frac{1}{2}\|\mathbf{X}\mathbf{W}-\hat{\mathbf{Y}}\|_F^2+\frac{\lambda_1}{2}\|\mathbf{X}-\mathbf{U}\mathbf{V}\|_F^2+ \quad (2)$$
$$\frac{\lambda_2}{2}\|\mathbf{U}\mathbf{P}-\mathbf{Y}\|_F^2+\frac{\lambda_3}{2}\sum_{i,j}^{q} r_{ij}c_{ij}d_{ij}^2+\lambda_4\|\mathbf{W}\|_1$$
$$s.t.\ \mathbf{U}\in[0,1]^{n\times l}$$

where $\mathbf{W}=[\mathbf{w}_1,...,\mathbf{w}_l]\in\mathbb{R}^{d\times l}$ is the coefficient matrix, and ℓ_1-norm regularization is used to learn sparse label-specific features [Zhang and Wu, 2015; Huang *et al.*, 2016; Huang *et al.*, 2018; Wei *et al.*, 2019; Wu *et al.*, 2019; Huang *et al.*, 2019]. For the simplicity of optimization, the discrete constraint on the values of matrix $\mathbf{U}$ is relaxed to continuous, i.e., $\forall\ u_{ij}\in[0,1]$.

It is worth noting that the latent labels may have correlations with known labels more or less, and thus it was expected that the performance on known labels and latent labels will be both boosted by exploiting the correlations between them. Thus, let $\mathbf{C}$ be the label correlation matrix. Each element c_{ij} indicates the value of correlation between the i-th and j-th labels, and is estimated by calculating the cosine similarity between $\hat{\mathbf{Y}}_{:,i}$ and $\hat{\mathbf{Y}}_{:,j}$. Since $\bar{\mathbf{Y}}$ is unknown, the calculated correlations between known and latent labels may not be reliable enough. Thus, we introduce an extra matrix $\mathbf{R}\in\mathbb{R}^{l\times l}$, and each element r_{ij} indicates the confidence of c_{ij} as

$$r_{ij}=\begin{cases}1 & ,\ \text{if } 1\le i,j\le q\\ \alpha & ,\ \text{otherwise; } \alpha\in[0,1]\end{cases} \quad (3)$$

Then, we try to exploit the pairwise label correlation by modeling the Euclidean distance between any pair of model coefficient vectors. Specifically, if y_i and y_j have a strong correlation, their corresponding coefficients $\mathbf{w}_i$ and $\mathbf{w}_j$ will be similar, and thus the distance (i.e., $d_{ij}=\|\mathbf{w}_i-\mathbf{w}_j\|_2$) will be small. Otherwise, the distance will be large. The fourth term of (2) was utilized to model pairwise label correlation.

3 Optimization

An alternating optimization strategy is adopted to solve problem (2), and $\mathcal{L}$ represents the objective function of it.

3.1 Update W

With $\mathbf{U}$ and $\mathbf{V}$ fixed, problem (2) reduces to

$$\min_{\mathbf{W}} \frac{1}{2}\|\mathbf{X}\mathbf{W}-\hat{\mathbf{Y}}\|_F^2+\frac{\lambda_3}{2}\mathrm{tr}(\mathbf{W}\mathbf{L}\mathbf{W}^T)+\lambda_4\|\mathbf{W}\|_1 \quad (4)$$

where $\mathbf{L}$ is graph Laplacian of the weighted correlation matrix $\mathbf{C}\odot\mathbf{R}$. Thus, the gradient w.r.t $\mathbf{W}$ can be calculated as

$$\nabla_{\mathbf{W}}\mathcal{L}=\mathbf{X}^T\mathbf{X}\mathbf{W}-\mathbf{X}^T\hat{\mathbf{Y}}+\lambda_3\mathbf{W}\mathbf{L} \quad (5)$$

The ℓ_1-norm regularization w.r.t $\mathbf{W}$ can be solved by the element-wise soft-threshold operator. According to the proximal gradient descend algorithm [Beck and Teboulle, 2009], $\mathbf{W}$ can be updated by

$$\mathbf{W}_{t+1}=\mathbf{prox}_{\frac{\lambda_4}{L_f}}(\mathbf{W}^{(t)}-\frac{1}{L_f}\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{P},\mathbf{W}^{(t)},\mathbf{U})) \quad (6)$$

where $\mathbf{W}^{(t)}=\mathbf{W}_t+\frac{\alpha_{t-1}-1}{\alpha_t}(\mathbf{W}_t-\mathbf{W}_{t-1})$. L_f is the Lipschitz constant, and an upper bound of it is shown in Theorem 1. For a sequence α_t, it should satisfy the condition of $\alpha_t^2-\alpha_t\le\alpha_{t-1}^2$, and $\mathbf{prox}_{\epsilon}(a)$ is the element-wise operator which is defined as

$$\mathbf{prox}_{\epsilon}(a)=\mathbf{sign}(a)\max(|a|-\epsilon,0) \quad (7)$$

Theorem 1 (Lipschitz Continuous Gradient)**.** *Given two arbitrary distinct parameters $\mathbf{W}_1$ and $\mathbf{W}_2$, we have*

$$\|\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}_1)-\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}_2)\|_F^2\le\gamma\|\Delta\mathbf{W}\|_F^2$$

where $\gamma=2\|\mathbf{X}^T\mathbf{X}\|_2^2+2\|\lambda_3\mathbf{L}\|_2^2$ and $\Delta\mathbf{W}=\mathbf{W}_1-\mathbf{W}_2$, and an approximate Lipschitz constant can be calculated by,

$$L_f=\sqrt{2\|\mathbf{X}^T\mathbf{X}\|_2^2+2\|\lambda_3\mathbf{L}\|_2^2} \quad (8)$$

Proof. Given $\mathbf{W}_1$ and $\mathbf{W}_2$, according to Eq. (5), we have

$$\begin{aligned}
&\|\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}_1)-\nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}_2)\|_F^2\\
&=\|\mathbf{X}^T\mathbf{X}\mathbf{W}_1+\lambda_3\mathbf{W}_1\mathbf{L}-\mathbf{X}^T\mathbf{X}\mathbf{W}_2-\lambda_3\mathbf{W}_2\mathbf{L}\|_F^2\\
&=\|\mathbf{X}^T\mathbf{X}(\mathbf{W}_1-\mathbf{W}_2)+\lambda_3(\mathbf{W}_1-\mathbf{W}_2)\mathbf{L}\|_F^2\\
&=\|\mathbf{X}^T\mathbf{X}\Delta\mathbf{W}+\lambda_3\Delta\mathbf{W}\mathbf{L}\|_F^2\\
&\le 2\|\mathbf{X}^T\mathbf{X}\Delta\mathbf{W}\|_F^2+2\|\lambda_3\Delta\mathbf{W}\mathbf{L}\|_F^2\\
&\le 2\|\mathbf{X}^T\mathbf{X}\|_2^2\|\Delta\mathbf{W}\|_F^2+2\|\lambda_3\mathbf{L}\|_2^2\|\Delta\mathbf{W}\|_F^2\\
&=(2\|\mathbf{X}^T\mathbf{X}\|_2^2+2\|\lambda_3\mathbf{L}\|_2^2)\|\Delta\mathbf{W}\|_F^2
\end{aligned}$$

□

Algorithm 1 Training of DLCL

Input: Training data: $\mathbf{X} \in \mathbb{R}^{n \times d}$, label matrix $\mathbf{Y} \in \mathbb{R}^{n \times q}$;
Parameter: The non-negative weighting parameters λ_1, λ_2, λ_3, and λ_4, the number of latent class labels k, and α;
Output: $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{V}$

$\alpha_1 = 1$.
Initialize $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{V}$ with random value.
$\bar{\mathbf{Y}} = \mathbf{U}_{(:,(q+1):l)}$.
while stop criterion not reached **do**
  Calculate $\mathbf{C}$ based on $\hat{\mathbf{Y}}$.
  Let $\mathbf{C} = \mathbf{C} \odot \mathbf{R}$, and then calculate $\mathbf{L}$.
  calculate L_f according to Eq. (8).
  update $\mathbf{W}$ according to Eq. (6).
  update $\mathbf{U}$ according to Eq. (10).
  update $\mathbf{V}$ according to Eq. (13).
  search β^* according to (14).
  update $\bar{\mathbf{Y}}$ according to Eq. (15).
  $\alpha_{t+1} \leftarrow \frac{1+\sqrt{4\alpha_t^2+1}}{2}$
end while
return $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{V}$

3.2 Update U

With $\mathbf{W}$ and $\mathbf{V}$ fixed, problem (2) becomes

$$\min_{\mathbf{U}} \frac{\lambda_1}{2}\|\mathbf{X} - \mathbf{U}\mathbf{V}\|_F^2 + \frac{\lambda_2}{2}\|\mathbf{U}\mathbf{P} - \mathbf{Y}\|_F^2 \tag{9}$$

Thus, the gradient w.r.t $\mathbf{U}$ can be calculated as

$$\nabla_{\mathbf{U}}\mathcal{L} = \lambda_1(\mathbf{U}\mathbf{V}\mathbf{V}^T - \mathbf{X}\mathbf{V}^T) + \lambda_2(\mathbf{U}\mathbf{P}\mathbf{P}^T - \mathbf{Y}\mathbf{P}^T)$$

Therefore, we can obtain a closed-form solution for $\mathbf{U}$ as

$$\mathbf{U} = (\lambda_1\mathbf{X}\mathbf{V}^T + \lambda_2\mathbf{Y}\mathbf{P}^T)(\lambda_1\mathbf{V}\mathbf{V}^T + \lambda_2\mathbf{P}\mathbf{P}^T)^{-1} \tag{10}$$

Then, $\forall\, u_{ij} \in [0, 1]$ can be achieved by $\mathbf{U} = \max(\mathbf{U}, \mathbf{0})$ and the min-max normalization over each column of $\mathbf{U}$.

3.3 Update V

With $\mathbf{U}$ and $\mathbf{W}$ fixed, problem (2) is simplified as

$$\min_{\mathbf{V}} \frac{\lambda_1}{2}\|\mathbf{X} - \mathbf{U}\mathbf{V}\|_F^2 \tag{11}$$

Consequently, the gradient w.r.t $\mathbf{V}$ can be calculated by

$$\nabla_{\mathbf{V}}\mathcal{L} = \lambda_1\mathbf{U}^T\mathbf{U}\mathbf{V} - \lambda_1\mathbf{U}^T\mathbf{X} \tag{12}$$

Then, a closed-form solution for $\mathbf{V}$ can be obtained as

$$\mathbf{V} = (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\mathbf{X} \tag{13}$$

3.4 Update $\hat{\mathbf{Y}}$

In (2), $\hat{\mathbf{Y}}$=[$\mathbf{Y}$, $\bar{\mathbf{Y}}$] is the full label matrix, and $\bar{\mathbf{Y}}$ indicates the latent label matrix and is unknown in advance. Therefore, we need to update $\bar{\mathbf{Y}}$ after each iteration of the optimization. As aforementioned, $\mathbf{U}$ is an approximate representation of $\hat{\mathbf{Y}}$. Besides, we can also obtain a result from the classifier, i.e., $\mathbf{X}\mathbf{W}$, and it is expected that the useful information (i.e., label correlation) induced from the known label can lead to a good prediction on the latent labels. Therefore, we plan to update $\bar{\mathbf{Y}}$ according to both of $\mathbf{U}$ and $\mathbf{X}\mathbf{W}$ with a balance weight $\beta \in (0, 1)$. Since $\mathbf{Y}$ is known, we can search a proper value for β from $\{0.05, 0.1, ..., 0.95\}$ according to (14).

$$\beta^* = \arg\min_{\beta} \ \|\mathbf{Y} - (\beta\mathbf{U}\mathbf{P} + (1-\beta)\mathbf{X}\mathbf{W}\mathbf{P})\|_1 \tag{14}$$

Then, $\bar{\mathbf{Y}}$ can be updated by

$$\bar{\mathbf{Y}} = \beta^*\mathbf{U}_{(:,(q+1):l)} + (1-\beta^*)\mathbf{X}\mathbf{W}_{(:,(q+1):l)} \tag{15}$$

Data set	#Instance	#Feature	#Label	Card	Domain
arts	5000	462	26	1.64	text
bibtex	7395	1836	159	2.40	text
corel16k001	13766	500	153	2.86	image
corel16k002	13761	500	164	2.88	image
corel5k	5000	499	374	3.52	image
education	5000	438	30	1.59	text
medical	978	1449	45	1.245	text
rcv1v2(subset1)	6000	944	101	2.88	text
stackex-chemistry	6961	540	175	2.11	text
stackex-cooking	10491	577	400	2.23	text
stackex-cs	9270	635	274	2.56	text
stackex-philosophy	3971	842	233	2.27	text

Table 1: Description of data sets. (*Card* indicates the average number of labels per instance.)

4 Experiment

4.1 Experimental Setting

By surveying previous work on MLL, we noted that there is no previous work doing the same topic like us. MIMLNC [Pham *et al.*, 2015] and DMNL [Zhu *et al.*, 2017a] are the only two highly related studies on discovering novel labels, but these two approaches are tailored for MIMLL, and cannot be applied to general single-instance MLL directly. In order to verify the effectiveness of our proposed method, we compare DLCL with four state-of-the-art approaches in terms of their performance on known labels. Detailed configurations of them are summarized as: 1) **BR** [Boutell *et al.*, 2004]: Binary Relevance. It learns a binary classifier (one-vs-rest) for each label independently, and Linear Regression is utilized as the base binary learner for it, and the regularization parameter is tuned in $\{10^i | i = -2, ..., 2\}$. 2) **MLKNN** [Zhang and Zhou, 2007][1]: A lazy learning approach to MLL, the number of nearest neighbors k is searched in$\{7, ..., 17\}$. 3) **LLSF** [Huang *et al.*, 2016][2]: Learning label-specific features for MLL, the regularization parameters are tuned in $\{2^i | i = -10, ..., 10\}$. 4) **KRAM** [Jia and Zhang, 2019][3]: Multi-dimensional classification via knn feature augmentation. The number of nearest neighbors k is searched in$\{7, ..., 17\}$. 5) **DLCL**: The proposed approach in this paper. Parameters λ_1 and λ_2 are tuned in $\{10^i | i = 2, ..., 6\}$, λ_3 is tuned in $\{2^i | i = -2, ..., 4\}$, λ_4 is tuned in $\{10^i | i = -2, ..., 1\}$, and α is tuned in $\{0.4, 0.5, 0.6\}$. Parameter tuning for each of them is based on a 5-fold cross validation over the training data of each data set.

Table 1 shows a summarization of the twelve experimental data sets. We adopt six common evaluation metrics [Zhang

[1]code: http://palm.seu.edu.cn/zhangml/files/ML-kNN.rar
[2]code: http://www.escience.cn/people/huangjun/index.html
[3]code: http://palm.seu.edu.cn/zhangml/files/KRAM.rar

Data sets	BR	MLkNN	LLSF	KRAM	DLCL	BR	MLkNN	LLSF	KRAM	DLCL	BR	MLkNN	LLSF	KRAM	DLCL
	Hamming Loss $\downarrow$ $(q = 70\%l, k = 30\%l)$					**Hamming Loss** $\downarrow$ $(q = 80\%l, k = 20\%l)$					**Hamming Loss** $\downarrow$ $(q = 90\%l, k = 10\%l)$				
arts	.079	.086	.075	**.073**	.075	.071	.078	.070	**.066**	.069	.067	.074	.065	**.063**	.067
bibtex	**.017**	.018	**.017**	.020	**.017**	**.014**	.018	.016	.020	.016	**.015**	.016	.016	.018	**.015**
corel16k1	.030	**.028**	.033	.030	.029	.029	.028	.030	.031	**.027**	.027	.030	.030	.030	**.024**
corel16k2	.029	.029	.028	.028	**.027**	.027	**.023**	.027	.032	.024	.029	.027	.028	.030	**.026**
corel5k	.021	**.018**	.019	.022	**.018**	.019	**.017**	.020	.020	.018	.018	**.016**	.017	.019	.020
education	.063	.063	.062	**.058**	**.058**	.059	.058	.059	**.053**	.054	.052	.051	.052	**.046**	.048
medical	.015	.015	**.008**	**.008**	.009	**.010**	.015	.011	**.010**	.011	**.010**	.016	.011	.011	.011
rcv1v2s1	.034	**.030**	.032	**.030**	.034	.031	.032	.031	**.029**	.031	.030	.031	.031	**.028**	.030
chemistry	.019	.020	.019	**.017**	**.017**	.018	.021	.018	**.017**	**.017**	.017	.018	.017	**.015**	.017
cs	**.006**	.007	**.006**	**.006**	**.006**	**.006**	**.006**	**.006**	**.006**	**.006**	**.006**	.007	**.006**	.007	**.006**
cooking	**.011**	**.011**	.012	.012	**.011**	**.011**	.012	.013	**.011**	**.011**	.011	.011	.012	.011	**.010**
philosophy	.014	**.013**	**.013**	**.013**	**.013**	.013	.013	.013	**.012**	**.012**	.013	.013	.013	**.012**	**.012**
Data sets	**Average Precision** $\uparrow$ $(q = 70\%l, k = 30\%l)$					**Average Precision** $\uparrow$ $(q = 80\%l, k = 20\%l)$					**Average Precision** $\uparrow$ $(q = 90\%l, k = 10\%l)$				
arts	.451	.435	.460	.472	**.476**	.452	.434	.455	.469	**.473**	.579	.541	.583	.595	**.600**
bibtex	.486	.449	.502	.461	**.525**	.539	.481	.543	.484	**.566**	.600	.546	.609	.549	**.622**
corel16k1	.305	.294	.303	.272	**.318**	.299	.276	.296	.264	**.307**	.323	.295	.323	.278	**.329**
corel16k2	.307	.284	.302	.272	**.318**	.326	.287	.325	.287	**.330**	.329	.310	.330	.291	**.331**
corel5k	.315	.327	.315	.291	**.339**	.305	.316	.311	.281	**.332**	.298	.306	.293	.267	**.314**
education	.631	.621	.627	**.655**	.651	.626	.616	.621	**.648**	.646	.626	.618	.622	**.649**	**.649**
medical	.623	.619	**.648**	.634	.647	.754	.708	.757	.741	**.758**	.818	.750	.819	.791	**.821**
rcv1v2s1	.442	.460	.455	.463	**.471**	.442	.457	.441	.448	**.458**	.427	**.443**	.430	.430	**.443**
chemistry	.434	.405	.464	.413	**.476**	.458	.393	.465	.402	**.474**	.454	.396	.461	.394	**.470**
cs	.484	.416	.485	.378	**.490**	.495	.425	.497	.369	**.502**	.508	.415	.511	.375	**.516**
cooking	.538	.470	.527	.475	**.545**	.539	.465	.531	.466	**.548**	.531	.461	.538	.442	**.540**
philosophy	.486	.445	.518	.436	**.533**	.489	.437	.521	.420	**.526**	.478	.430	.511	.407	**.525**
Data sets	**One Error** $\downarrow$ $(q = 70\%l, k = 30\%l)$					**One Error** $\downarrow$ $(q = 80\%l, k = 20\%l)$					**One Error** $\downarrow$ $(q = 90\%l, k = 10\%l)$				
arts	.626	.669	.615	.615	**.612**	.617	.661	.612	**.607**	**.607**	.500	.575	.495	**.491**	.494
bibtex	.528	.582	.517	.556	**.512**	.471	.547	.471	.527	**.461**	.359	.424	.355	.416	**.350**
corel16k1	.744	.764	.748	.775	**.740**	**.745**	.774	.749	.778	.749	.688	.727	.690	.732	**.687**
corel16k2	**.737**	.770	.743	.770	**.737**	**.693**	.747	.694	.729	.697	.670	.705	**.667**	.708	.676
corel5k	.648	.655	.646	.683	**.633**	.651	.657	.649	.684	**.626**	**.644**	.659	.648	.694	.652
education	.478	.501	.477	**.454**	.470	.473	.498	.475	**.455**	.469	.471	.495	.472	**.456**	.461
medical	.395	.407	**.369**	.380	**.369**	.284	.355	**.279**	.294	.282	.223	.316	**.218**	.241	.219
rcv1v2s1	.592	.589	.573	**.572**	.581	.579	.571	.576	**.567**	.581	.577	**.569**	.575	**.569**	.580
chemistry	.607	.637	.576	.619	**.573**	.566	.638	.565	.616	**.563**	.563	.627	.561	.610	**.559**
cs	**.497**	.551	.499	.589	.499	.472	.522	**.471**	.587	**.471**	.438	.502	.438	.556	**.436**
cooking	**.473**	.513	.483	.537	.476	.457	.522	.464	.499	**.456**	.455	.522	.452	.504	**.451**
philosophy	.501	.564	.480	.548	**.468**	.481	.555	**.458**	.554	.526	.480	.549	.456	.552	**.448**
Data sets	**Ranking Loss** $\downarrow$ $(q = 70\%l, k = 30\%l)$					**Ranking Loss** $\downarrow$ $(q = 80\%l, k = 20\%l)$					**Ranking Loss** $\downarrow$ $(q = 90\%l, k = 10\%l)$				
arts	.119	.097	.116	**.084**	.086	.117	.096	.117	**.084**	.087	.129	.110	.129	**.095**	.096
bibtex	.097	.087	.084	.082	**.056**	.092	.091	.085	.088	**.060**	.095	.093	.084	.092	**.074**
corel16k1	.181	.158	.182	.167	**.147**	.189	.169	.188	.172	**.147**	.184	.169	.182	.172	**.145**
corel16k2	.175	.151	.175	.156	**.139**	.176	.163	.173	.157	**.134**	.175	.151	.172	.158	**.135**
corel5k	.200	**.115**	.174	.124	.125	.201	**.119**	.173	.127	.131	.202	**.118**	.179	.126	.123
education	.116	.084	.124	**.075**	.078	.119	.085	.129	**.078**	.080	.114	.076	.124	**.070**	.075
medical	.025	.022	**.011**	.021	**.011**	.019	.027	.023	.026	**.016**	**.018**	.032	.024	.030	.019
rcv1v2s1	.061	.039	.055	.042	**.035**	.057	.045	.056	.045	**.036**	.058	.046	.057	.047	**.037**
chemistry	.109	.101	.096	.097	**.072**	.111	.107	.100	.105	**.076**	.115	.114	.104	.111	**.082**
cs	.087	.117	.083	.106	**.064**	.089	.115	.084	.124	**.067**	.092	.133	.083	.130	**.069**
cooking	.068	.086	.085	.084	**.052**	.071	.093	.084	.092	**.055**	.074	.099	.061	.093	**.057**
philosophy	.123	.095	.091	.108	**.070**	.129	.104	.098	.118	**.071**	.138	.109	.099	.126	**.083**
Data sets	**Coverage** $\downarrow$ $(q = 70\%l, k = 30\%l)$					**Coverage** $\downarrow$ $(q = 80\%l, k = 20\%l)$					**Coverage** $\downarrow$ $(q = 90\%l, k = 10\%l)$				
arts	.142	.109	.137	**.097**	.102	.151	.119	.148	**.108**	.113	.184	.150	.182	**.136**	.140
bibtex	.146	.128	.126	.119	**.088**	.151	.139	.136	.136	**.099**	.174	.159	.149	.153	**.136**
corel16k1	.267	.234	.260	.246	**.217**	.287	.260	.277	.262	**.223**	.313	.290	.301	.295	**.248**
corel16k2	.259	.231	.254	.236	**.208**	.286	.270	.273	.259	**.218**	.317	.275	.302	.288	**.244**
corel5k	.428	**.258**	.356	.273	.281	.442	**.272**	.349	.286	.299	.451	**.273**	.376	.288	.287
education	.154	.104	.163	**.096**	.103	.165	.112	.177	**.105**	.112	.162	.101	.172	**.096**	.106
medical	.017	.016	**.002**	.014	.003	.021	.032	.024	.030	**.017**	**.024**	.041	.030	.039	**.024**
rcv1v2s1	.134	.091	.118	.099	**.083**	.130	.106	.124	.107	**.087**	.137	.113	.131	.114	**.093**
chemistry	.174	.157	.149	.149	**.117**	.187	.176	.163	.171	**.131**	.201	.189	.176	.192	**.147**
cs	.141	.169	.130	.180	**.103**	.151	.192	.138	.201	**.116**	.170	.231	.147	.229	**.128**
cooking	.122	.148	.141	.145	**.094**	.138	.172	.152	.168	**.108**	.151	.178	**.116**	.187	.117
philosophy	.206	.163	.151	.181	**.123**	.227	.188	.170	.209	**.132**	.253	.206	.182	.231	**.161**
Data sets	**Macro AUC** $\uparrow$ $(q = 70\%l, k = 30\%l)$					**Macro AUC** $\uparrow$ $(q = 80\%l, k = 20\%l)$					**Macro AUC** $\uparrow$ $(q = 90\%l, k = 10\%l)$				
arts	.576	.603	.580	**.614**	.609	.580	.606	.581	**.616**	.610	.743	.773	.744	**.785**	.780
bibtex	.745	.753	.759	.761	**.785**	.818	.820	.829	.823	**.852**	.883	.886	.898	.891	**.906**
corel16k1	.742	.761	.743	.752	**.776**	.751	.766	.754	.764	**.793**	.794	.805	.797	.801	**.832**
corel16k2	.747	.763	.747	.760	**.782**	.787	.794	.793	.801	**.829**	.813	.834	.818	.827	**.853**
corel5k	.790	**.876**	.819	.867	.867	.795	**.878**	.820	.870	.867	.798	**.881**	.822	.873	.878
education	.827	.869	.817	**.877**	.871	.836	.879	.825	**.887**	.880	.846	.895	.837	**.901**	.892
medical	.652	.653	**.667**	.655	.666	.810	.800	.808	.801	**.814**	**.884**	.869	.879	.871	**.884**
rcv1v2s1	.664	.675	.672	.683	**.692**	.671	.682	.673	.682	**.693**	.673	.684	.675	.683	**.695**
chemistry	.783	.793	.800	.798	**.823**	.824	.828	.838	.831	**.860**	.846	.846	.860	.850	**.879**
cs	.792	.771	.798	.762	**.816**	.828	.800	.835	.789	**.850**	.866	.827	.878	.825	**.891**
cooking	.856	.837	.842	.840	**.873**	.891	.870	.881	.871	**.908**	.902	.884	.918	.877	**.920**
philosophy	.791	.818	.828	.805	**.848**	.811	.832	.847	.818	**.871**	.820	.845	.863	.829	**.878**

Table 2: Experimental results (mean) of all the comparing approaches on *known labels* in terms of each evaluation metric.

Metric	F_F	Critical Value ($\alpha = 0.05$)
Hamming Loss	4.4744	2.4363
Average Precision	48.8990	
One Error	27.2180	
Ranking Loss	28.6542	
Coverage	27.1924	
Macro AUC	28.6498	

Table 3: Summary of the Friedman statistics $F_F(k = 5, N = 36)$ and the critical value in terms of each evaluation metric (k: # of comparing algorithms; N: # of data sets)

and Zhou, 2014], i.e., *Hamming Loss*, *Average Precision*, *One Error*, *Ranking Loss*, *Coverage*, and *Macro AUC*, to evaluate the performance of the comparing algorithms on *known labels*. To evaluate the performance of our method on discovered *latent labels*, we adopt the following metric proposed in [Zhu *et al.*, 2017a],

$$F_{\text{novel}} = \frac{1}{k}\sum_{i=1}^{k}\max(\{\mathcal{F}(\hat{\mathbf{Y}}_{:,q+i}, \mathbf{G}_{:,q+j}), j \in \{1, ..., k\}\})$$

where $\mathcal{F}(\cdot)$ is the function of F-measure, and $\mathbf{G}$ indicates the ground-truth label matrix. F_{novel} measures the average performance on detected multiple latent labels on the ground-truth label that best matches.

4.2 Experimental Results

For each data set, 80% of it are randomly generated as the training part and 20% for testing, which is repeated 10 times. Following the settings in previous work on discovering new labels for MIMLL [Pham *et al.*, 2015; Zhu *et al.*, 2017a], the first 70%, 80% and 90% labels are set be to known labels and the rest are taken as latent ones respectively. The average results of each comparing algorithm on the known labels are shown in Table 2. $\uparrow$ ($\downarrow$) indicates the larger (smaller) the value, the better the performance. Best results are highlighted in bold face.

Results on Known Labels. Friedman test [Demšar, 2006] is employed to conduct performance analysis among the comparing approaches, and the result of it is shown in Table 3. As shown in Table 3, the null hypothesis that all the comparing algorithms perform equivalently is clearly rejected in terms of all the evaluation metrics at significance level $\alpha = 0.05$, Consequently, the Nemenyi test [Demšar, 2006] is adopted to analyse the relative performance among them. For Nemenyi test, the critical difference CD=$q_\alpha\sqrt{\frac{k(k+1)}{6N}}$=1.0167 ($k$=5, N=36) with q_α=2.728 at significance level α=0.05, where k is the number of algorithms and N (16×3) is the number of data sets. The CD diagrams of DLCL w.r.t to the comparing algorithms on each evaluation metric are shown in Figure 1. In each sub-figure, any comparing algorithm whose average rank is within one CD to that of DLCL is connected. Otherwise, any algorithm not connected with DLCL is considered to have significant different performance between them. According to these experimental results, the following observations can be made: 1) The proposed method DLCL significantly outperforms the comparing algorithms in terms of *ranking loss*, *average precision*, *coverage*, and *AUC*. Besides, DLCL statistically outperforms the comparing algorithms in terms of *hamming loss* and *one error*. These results definitely demonstrate the effectiveness of our method on MLL. 2) KRAM and MLkNN algorithms are all constructed based on the information of k nearest neighbors of each instance. It is worth noting that these two algorithms achieve worse performance on those data with a large number of labels (e.g., $l \geq 100$) than on those data with a small number of labels. One possible reason might be that the k nearest neighbors can not provide sufficient information for model construction when many labels are hidden in the data. These results verify the importance of discovering latent class labels for MLL.

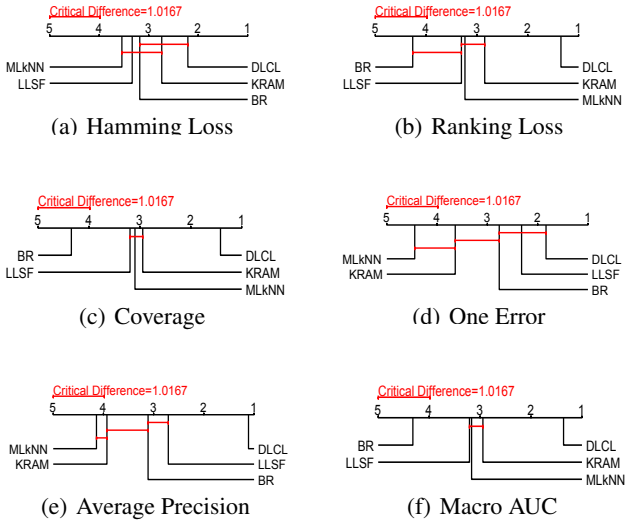

Figure 1: Comparison of DLCL against the comparing approaches with the Nemenyi test. Groups of classifiers that are not significantly different from DLCL (at $p = 0.05$) are connected.

Data sets	DLCL (k, q)			RS ($.3l$, $.7l$)		
	($.1l$, $.9l$)	($.2l$, $.8l$)	($.3l$, $.7l$)	r_1	r_2	r_3
arts	.235±.03	.315±.03	.265±.01	.153	.154	**.155**
bibtex	.093±.01	.197±.01	.215±.01	.090	.087	**.096**
corel16k1	.266±.01	.265±.01	.251±.01	**.066**	.066	.066
corel16k2	.223±.01	.240±.01	.249±.01	.065	.080	**.106**
corel5k	.094±.01	.145±.01	.144±.01	**.129**	.112	.094
education	.051±.01	.100±.01	.123±.01	.105	**.119**	.118
medical	.163±.02	.224±.03	.273±.03	**.148**	.145	.145
rcv1v2s1	.182±.02	.202±.02	.203±.02	.094	.106	**.117**
chemistry	.115±.02	.131±.01	.149±.01	**.088**	.084	.086
cooking	.176±.02	.203±.01	.229±.01	**.104**	.080	.073
cs	.146±.01	.171±.01	.203±.01	**.089**	.079	.079
philosophy	.154±.02	.160±.01	.192±.01	**.135**	.111	.102

Table 4: Experiment results ($F_{\text{novel}} \uparrow$) on *latent labels*. RS ($.3l$, $.7l$) indicates the average results of randomly setting $r_i (r_i = i)$ latent labels for each instance when k=$0.3l$ and q=$0.7l$ over 50 repetitions.

Results on Latent Labels. Table 4 shows the results of DLCL on discovered latent class labels according to F_{novel}. It is clearly indicated that DLCL can discover the latent labels for MLL, and DLCL significantly outperforms RS(Randomly Setting) when $q = 70l\%$ and $k = 30l\%$. For some data sets, the results of F_{novel} differ extremely under different ratios of latent labels. The possible reason might be that the difficulties of prediction of different labels are different. On the other hand, we want to know what latent labels have we discov-

Matched Label Name	F_{novel}	Top 20 Features (i.e., feature name (weight))
regular-languages	0.712	**regular**(0.898), beginalign(0.062), endalign(0.052), essenti(0.045), express(0.039), pump(0.036), confus(0.035), project(0.034), homework(0.034), attempt(0.034), independ(0.032), **languag**(0.03), let(0.027), digit(0.027), cup(0.026), lot(0.026), comment(0.026), identifi(0.026), algebra(0.025), here(0.025)
pumping-lemma	0.635	**pump**(0.855), **lemma**(0.681), essenti(0.139), easier(0.12), identifi(0.1), formul(0.083), split(0.076), condit(0.068), geq(0.068), respect(0.067), nice(0.067), forc(0.063), short(0.061), imagin(0.058), author(0.056), larger(0.054), yield(0.054), effect(0.053), pseudocod(0.053), insid(0.051)
reductions	0.564	**reduct**(0.83), beginalign(0.11), endalign(0.099), reduc(0.074), langl(0.065), notic(0.054), show(0.053), sat(0.039), suggest(0.034), shown(0.034), nice(0.031), maxim(0.03), naiv(0.029), undecid(0.029), simpl(0.027), research(0.027), flow(0.026), wikipedia(0.024), wrong(0.024), literatur(0.024)
turing-machines	0.465	**machin**(0.829), **ture**(0.617), tape(0.122), encod(0.041), power(0.038), occur(0.036), simul(0.034), digit(0.031), determinist(0.030), build(0.030), group(0.029), appear(0.029), qquad(0.028), make(0.028), automaton(0.027), uniqu(0.027), troubl(0.027), play(0.026), halt(0.026), endalign(0.026)
sorting	0.448	**sort**(0.849), comparison(0.136), insert(0.119), quick(0.061), compar(0.035), addit(0.031), entri(0.030), notat(0.029), origin(0.028), select(0.027), algorithm(0.027), automata(0.027), free(0.026), easi(0.026), sourc(0.026), qquad(0.026), mark(0.026), shown(0.026), digit(0.025), argument(0.025)

Table 5: Results of the five best matched labels

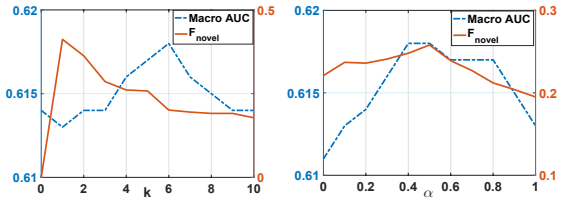

Figure 2: Influence of k and α on DLCL over *arts*.

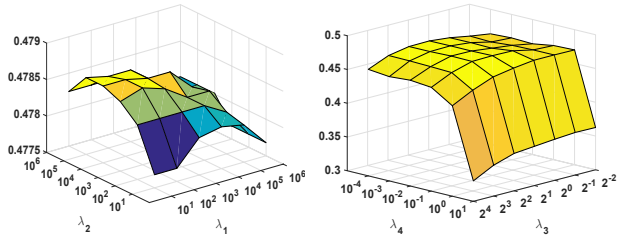

Figure 3: Parameter analysis on DLCL over *stackex-chemistry*.

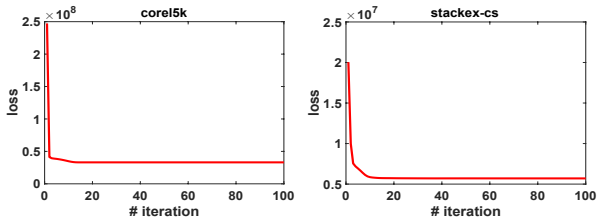

Figure 4: Two examples of the convergence curves of DLCL.

ered. We try to describe the meaning of them by the learned labels-specific features which are indicated by the non-zero entities of each column of $\mathbf{W}$. Table 5 shows the results of the five best matched labels over the *stackex-cs* data set with $q = 70l\%$ and $k = 30l\%$. It is noted that the names of top five matched labels are the occurred among the top 20 features or homologous with them, and most of features have a strong semantic correlation with the name of labels.

4.3 Parameter and Convergence Analysis

The number of Latent labels k and correlation confidence α. For *arts*, the first 20 and the rest 6 labels are set as known and latent labels respectively. Figure 2 shows the average results of DLCL over 10 repetitions with different values of k and α. The result (i.e., *Macro AUC*) on known labels is improved by discovering latent labels (i.e., $k > 0$), and the F_{novel} decreases with the increasing of k. The larger the number of latent labels, the harder it is to discover them. Therefore, we could set k to be a relative small value and run DLCL multiple times. It is also noted that the performance of latent labels can be significantly improved with the help of known labels and the improvement on known labels is slightly with the help of latent labels by exploiting their correlations (i.e., $\alpha > 0$). The known labels are observed in advance and can be considered as a teacher to guide the prediction on latent labels.

Analysis on regularization parameters. The average results (i.e, *Average Precision*) of DLCL with different values of λ_1, λ_2, λ_3, and λ_4 over *stackex-chemistry* are shown in Figure 3, and similar results were also obtained for the other data sets. It is noted that the performance of DLCL is insensitive to the parameters, and also the optimal performance is usually achieved at some intermediate values of each parameter.

Convergence. Figure 4 demonstrates two examples of the convergence curves of DLCL. It is noted that the values of the objective function are non-increasing and drop sharply around 15 iterations on *corel5k* and *stackex-cs* data sets.

5 Conclusion

In this paper, a novel approach named DLCL is proposed for MLL which can not only discover the latent labels in the training data but also predict new instances with these latent labels and known labels simultaneously. The experimental results demonstrate that the performance of latent labels can be significantly improved with the help of known labels and the performance of known labels can be improved with the help of latent labels by exploiting their correlations. Extensive experiments with other state-of-the-art MLL approaches have show a competitive performance of DLCL.

Acknowledgement

This work is supported by JST KAKENHI: 191400000190, JST-AIP: JPMJCR19U4, and NSFC: 61806005.

References

[Beck and Teboulle, 2009] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.

[Boutell *et al.*, 2004] M. R. Boutell, J.-B. Luo, X.-P. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognit.*, 37(9):1757–1771, 2004.

[Chen *et al.*, 2019] C. Chen, H. Wang, W. Liu, X. Zhao, T. Hu, and G. Chen. Two-stage label embedding via neural factorization machine for multi-label classification. In *AAAI*, pages 3304–3311, 2019.

[Demšar, 2006] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7:1–30, 2006.

[Gibaja and Ventura, 2015] E. Gibaja and S. Ventura. A tutorial on multilabel learning. *ACM Comput. Surv.*, 47(3):52:1–52:38, 2015.

[Hu and Chen, 2019] M.-L. Hu and S.-C. Chen. One-pass incomplete multi-view clustering. In *AAAI*, pages 3838–3845, 2019.

[Hua and Qi, 2008] X.-S. Hua and G.-J. Qi. Online multilabel active annotation: Towards large-scale content-based video search. In *ACM MM*, pages 141–150, 2008.

[Huang *et al.*, 2016] J. Huang, G. Li, Q. Huang, and X. Wu. Learning label-specific features and class-dependent labels for multi-label classification. *IEEE Trans. Knowl. Data Eng.*, 28(12):3309–3323, 2016.

[Huang *et al.*, 2018] J. Huang, G. Li, Q. Huang, and X. Wu. Joint feature selection and classification for multilabel learning. *IEEE Trans. Cybern.*, 48(3):876 – 889, 2018.

[Huang *et al.*, 2019] J. Huang, F. Qin, X. Zheng, Z. Cheng, Z. Yuan, W. Zhang, and Q. Huang. Improving multi-label classification with missing labels by learning label-specific features. *Inf. Sci.*, 492:124–146, 2019.

[Jain *et al.*, 2019] H. Jain, V. Balasubramanian, B. Chunduri, and M. Varma. Slice: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *WSDM*, pages 528–536, 2019.

[Jia and Zhang, 2019] B.-B. Jia and M.-L. Zhang. Multi-dimensional classification via knn feature augmentation. In *AAAI*, pages 3975–3982, 2019.

[Kuzborskij *et al.*, 2013] I. Kuzborskij, F. Orabona, and B. Caputo. From n to n+1: Multiclass transfer incremental learning. In *CVPR*, pages 3358–3365, 2013.

[Nguyen *et al.*, 2016] V. Nguyen, T. D. Nguyen, T. Le, S. Venkatesh, and D. Phung. One-pass logistic regression for label-drift and large-scale classification on distributed systems. In *ICDM*, pages 1113–1118, 2016.

[Pham *et al.*, 2015] A. Pham, R. Raich, X. Fern, and J. P. Arriaga. Multi-instance multi-label learning in the presence of novel class instances. In *ICML*, pages 2427–2435, 2015.

[Tan *et al.*, 2018] Q. Tan, G. Yu, C. Domeniconi, J. Wang, and Z. Zhang. Incomplete multi-view weak-label learning. In *IJCAI*, 2018.

[Wang *et al.*, 2019] H. Wang, W. Liu, Y. Zhao, C. Zhang, T. Hu, and G. Chen. Discriminative and correlative partial multi-label learning. In *IJCAI*, pages 3691–3697, 2019.

[Wei *et al.*, 2019] T. Wei, W.-W. Tu, and Y.-F. Li. Learning for tail label data: A label-specific feature approach. In *IJCAI*, pages 3842–3848, 2019.

[Wu *et al.*, 2019] X. Wu, Q. Chen, Y. Hu, D. Wang, X. Chang, X. Wang, and M.-L. Zhang. Multi-view multi-label learning with view-specific information extraction. In *IJCAI*, pages 3884–3890, 2019.

[Xing *et al.*, 2018] Y. Xing, G. Yu, C. Domeniconi, J. Wang, and Z. Zhang. Multi-label co-training. In *IJCAI*, 2018.

[Xioufis *et al.*, 2011] E. S. Xioufis, M. Spiliopoulou, G. Tsoumakas, and I. Vlahavas. Dealing with concept drift and class imbalance in multi-label stream classification. In *IJCAI*, pages 1583–1588, 2011.

[Xu *et al.*, 2019] N. Xu, J. Lv, and X. Geng. Partial label learning via label enhancement. In *AAAI*, pages 3838–3845, 2019.

[Yang *et al.*, 2016] H. Yang, J. T. Zhou, and J. Cai. Improving multi-label learning with missing labels by structured semantic correlations. In *ECCV*, pages 835–851, 2016.

[Yu *et al.*, 2014] H. Yu, P. Jain, P. Kar, and I. S. Dhillon. Large-scale multi-label learning with missing labels. In *ICML*, pages 593–601, 2014.

[Zhang and Wu, 2015] M.-L. Zhang and L. Wu. Lift: Multi-label learning with label-specific features. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(1):107–120, 2015.

[Zhang and Zhou, 2007] M.-L. Zhang and Z.-H. Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognit.*, 40(7):2038–2048, 2007.

[Zhang and Zhou, 2014] M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.*, 26(8):1819–1837, 2014.

[Zhang *et al.*, 2018] Y. Zhang, R. Henao, Z. Gan, Y. Li, and L. Carin. Multi-label learning from medical plain text with convolutional residual models. In *MLHC*, pages 280–294, 2018.

[Zhang *et al.*, 2020] Y. Zhang, Y. Wang, X.-Y. Liu, S. Mi, and M.-L. Zhang. Large-scale multi-label classification using unknown streaming images. *Pattern Recognit.*, page 107100, 2020.

[Zhu *et al.*, 2017a] Y. Zhu, K.-M. Ting, and Z.-H. Zhou. Discover multiple novel labels in multi-instance multi-label learning. In *AAAI*, pages 2977–2983, 2017.

[Zhu *et al.*, 2017b] Y. Zhu, K.-M. Ting, and Z.-H. Zhou. New class adaption via instance generation in one-pass class incremental learning. In *ICDM*, pages 1207–1212, 2017.

[Zhu *et al.*, 2018] Y. Zhu, K.-M. Ting, and Z.-H. Zhou. Multi-label learning with emerging new labels. *IEEE Trans. Knowl. Data Eng.*, 30(10):1901–1914, 2018.