Partial Multi-label Learning using Label Compression

Tingting Yu[1,2], Guoxian Yu[1,2,3,4,*], Jun Wang[3], Carlotta Domeniconi[5], Xiangliang Zhang[4]
[1]School of Software, Shandong University, Jinan, China
[2]College of Computer and Information Sciences, Southwest University, Chongqing, China
[3]Joint SDU-NTU Centre for Artificial Intelligence Research, Shandong University, Jinan, China
[4]CEMSE, King Abdullah University of Science and Technology, Thuwal, SA
[5]Department of Computer Science, George Mason University, VA, USA
Email: tingting01@swu.edu.cn; {gxyu, kingjun}@sdu.edu.cn, carlotta@cs.gmu.edu; xiangliang.zhang@kaust.edu.sa

***Abstract*—Partial multi-label learning (PML) aims at learning a robust multi-label classifier from partial multi-label data, where a sample is annotated with a set of candidate labels, while only a subset of those labels is valid. The existing PML algorithms generally suffer from the high computational cost when learning with large label spaces. In this paper, we introduce a PML approach (PML-*LCom*) that uses *Label Compression* to efficiently learn from partial multi-label data. PML-*LCom* firstly splits the observed label data matrix into a latent relevant label matrix and an irrelevant one, and then factorizes the relevant label matrix into two low-rank matrices, one encodes the compressed labels of samples, and the other explores the underlying label correlations. Next, it optimizes the coefficient matrix of the multi-label predictor with respect to the compressed label matrix. In addition, it regularizes the compressed label matrix with respect to the feature similarity of samples, and optimizes the label matrix and predictor in a coherent manner. Experimental results on both semi-synthetic and real-world PML datasets show that PML-*LCom* achieves a performance superior to the state-of-the-art solutions on predicting the labels of unlabeled samples with a large label space. The label compression improves both the effectiveness and efficiency, and the coherent optimization mutually benefits the label matrix and predictor.**

***Index Terms*—Partial Multi-label Learning, Label Compression, Low-rank, Noisy labels**

I. Introduction

Multi-label learning (MLL) is a fundamental learning paradigm in various practical domains. It handles the scenario where each sample can be simultaneously annotated with multiple non-exclusive but semantically related labels [1], [2]. MLL has been studied with applications in diverse domains, such as image recognition [3], gene function prediction [4], [5], text categorization [6], and so on. These MLL solutions often assume that each training sample is precisely annotated with all of its relevant labels. However, in many real-world scenarios, it is difficult and costly to collect precisely annotated training data. For example, in Figure 1, the image is annotated via crowdsourcing with seven candidate labels. Among these labels, *tree, grass, giraffe*, and *zebra* are relevant, while the other three are irrelevant (**noisy**). For another example, the functional annotations of genes in public databases (i.e., Gene Ontology) are collaboratively annotated by different groups, although quality control techniques have been applied, these curated annotations still have some noisy ones [7]. It is non-trivial to induce a reliable classifier from such partial multi-label data, since the ground-truth labels are concealed in the candidate labels, and the classifier is prone to be mislead by the irrelevant ones. To handle such learning scenario, partial multi-label learning (PML) has been recently proposed [8], [9] and attracted considerable attention [10], [11], [12], [13].

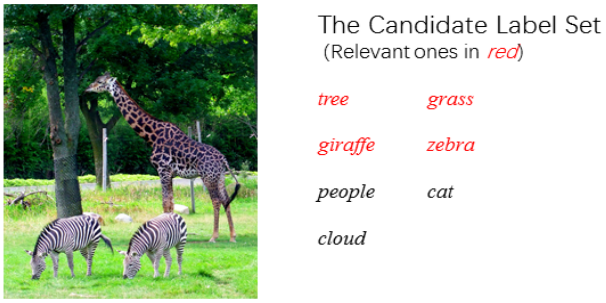


Fig. 1: A partial multi-label image is tagged with relevant (in red font) and noisy (in black font) labels.

Formally, let $\mathcal{X} \in \mathbb{R}^d$ denote the d-dimensional feature space, and $\mathcal{Y} = \{0|1\}^q$ be the output label space with q possible distinct labels. Given a PML training dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathcal{Y}_i)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector of the i-th training sample, and $\mathcal{Y}_i \subseteq \mathcal{Y}$ corresponds to its candidate label set of $\mathbf{x}_i$. The key assumption of PML is that the ground-truth labels $\widehat{\mathcal{Y}}_i$ of $\mathbf{x}_i$ is only a subset of $\mathcal{Y}_i$, i.e., $\widehat{\mathcal{Y}}_i \subseteq \mathcal{Y}_i$, but $\widehat{\mathcal{Y}}_i$ is not accessible to the learning algorithm, which makes the learning process much more difficult than MLL, and not to mention partial label learning (PLL) [14], [15], in which each sample is only annotated with one label from $\mathcal{Y}_i$ and the labels are mutually exclusive. The goal of PML is to learn a multi-label predictor $f: \mathcal{X} \mapsto 2^{\mathcal{Y}}$ from $\mathcal{D}$ with the consideration of the label correlations and noisy labels.

Based on the fact that relevant labels are a subset of

*Corresponding author: gxyu@sdu.edu.cn (Guoxian Yu). This work is supported by NSFC (No. 61872300, 62031003 and 62072380).

the candidate labels, a PML classifier was designed to rank candidate labels ahead of non-candidate ones [8]. However, the classifier would be misled when the proportion of noisy labels is high. Recently, PML solutions in [10], [16], [17], [12] follow a two-stage procedure that first refines labels by eliminating irrelevant ones and then induces a classifier on the refined labels. To jointly optimize PML models for multi-label prediction and irrelevant label identification, the ground-truth label matrix was assumed to be low-rank in [9], [11], [18]. Despite effective, these PML solutions all suffer from a high computational cost when handling a *large label space*. For example, the models in [8] have a prohibitive runtime to compute the relevance ranks between candidate labels and non-candidate ones with multiple SVM classifiers. The models are too costly to be applied when the number of labels is more than 100. The two-stage solution in [12] involves with iterative label propagation and a total of $q(q-1)/2$ binary training tasks (one for each pair of labels). DRAMA [16] solves a series of quadratic programming (QP) problems in the first stage to obtain label confidence matrix, and then induces a gradient boosting regressor to fit the label confidences. In each boosting round, it needs to augment the feature space with elicited labels to capture the correlation between q basic regressors. The low-rank approximation-based models [9], [11] involve with heavy computation for singular value decomposition and matrix inversion. Moreover, these PML solutions learn multi-label predictors with a large number of feature-to-label mapping coefficients (i.e., $d \times q$ for linear predictors, nq^2 for SVM-based predictors). During their training, the average number of labels per sample is much smaller than q, causing the issue of sparse label annotations. The noisy and sparse labels, and the large label space jointly hinder these PML solutions to induce robust multi-label predictors efficiently.

In this paper, we propose a novel PML approach based on label compression (PML-*LCom*). The intuition is to compress labels into a low-dimensional space where the multi-label predictor can be efficiently and effectively trained. Although label compression has been used for multi-label learning and extreme multi-label learning [19], [20], [21], our work is the first to take its advantage on addressing the PML problems. PML-*LCom* first splits the observed label matrix into a *latent relevant label matrix* and a *noisy label matrix*, and then factorizes the *relevant label matrix* into two low-rank matrices; one encodes the compressed labels of samples, and the other explores the underlying label correlations. An l_1-norm regularization is enforced on the noisy label matrix to account for sparse noisy labels. Next, it coordinates relevant label matrix learning using the feature data matrix, and trains a multi-label predictor with respect to the compressed label matrix. PML-*LCom* integrates the label compression and predictor training in a unified objective function, and further introduces an alternative optimization solution to efficiently optimize them.

The main contributions of this work are summarized as follows:

- We execute and verify that label compression can significantly improve the performance and efficiency of PML on datasets with large label spaces.
- We introduce a unified objective function to compress labels and train a multi-label predictor with compressed labels, along with an alternative optimization procedure to jointly achieve the two mutually beneficial objectives.
- PML-*LCom* significantly outperforms recent PML approaches [11], [16], [8], representative MLL methods [19], [21], [22] on diverse benchmark datasets in terms of classification effectiveness and run-time efficiency.

II. Related Work

PML is a new branch of the popular multi-label learning (MLL) and it is also closely related with partial label learning (PLL) [14]. MLL aims to train a multi-label predictor on labeled (and unlabeled) multi-label samples and thus to assign a set of non-exclusive labels to new samples [1], [2]. Most MLL solutions attempt different techniques to explore and exploit the label correlations of multi-label data to induce a multi-label classifier [3], [23], [24]. A line of methods handle a large label space by separately handling the frequent labels and the long tail ones, or by label space dimension reduction [19], [20], [21]. Other solutions try to train the predictor on multi-label data with missing labels and/or replenish the missing labels [25], [26], [27], [28], [29], [30]. Some other methods address the extreme classification problem that tag a most relevant subset of labels from an extremely large label set [31], [32]. Though these methods reduce the size of candidate labels from the extremely large label set, the most relevant subset still contains a lot of noisy labels with a large label space.

PLL is another popular learning paradigm where each training sample is annotated with a set of candidate labels while only one label among them is valid for the sample. The key to solve PLL problems is disambiguating the candidate label set. To this end, existing disambiguation based approaches can be categorized into the average-based strategy and the identification-based strategy. Where the former strategy treats each candidate label equally in the training process, and the final prediction for unseen samples is made by averaging the modeling outputs of all the candidate labels [15]. For the latter, it generally considers the unique ground-truth label as a latent variable, and then aggregates the modeling outputs to optimize specific objectives [33], [34].

PML is more challenging than MLL, given the unknown relevant/irrelevant labels of training data; it is also more difficult than PLL, since more than one label within the candidate label set can be identified as the ground-truth. This learning scenario was firstly studied in bioinformatics domain to identify the noisy functional annotations of genes using the similarity between genes and label hierarchy [35], [7]. Xie ang Huang [8] extended this scenario to more general settings, they minimized the confidence weighted ranking loss between candidate labels and non-candidate ones, and proposed two effective PML methods PML-*lc* and PML-*fp*, which additionally use the label correlations and feature prototypes, respectively. Some methods firstly elicited the ground-truth labels and then

induced the multi-label classifier based on the elicited labels. For example, PARTICLE [10] estimated the credible labels by label propagation; DRAMA [16] learnt the confidence value for each label by the feature manifold. Similarly, PML-LD [17] recovered the label distributions by leveraging the topological information of the feature space and correlations among the labels, and then induced the prediction model using the recovered labels in the second stage. While GRADIS [12] extended PML on multi-view data by disambiguating with a fused similarity graph obtained from different views. Because of the label correlations of multi-label data, the observed label data matrix should be low-rank [26], [28]. Motivated by this observation, fPML [9] collaboratively factorized the label and feature data matrices of the training data into low-rank matrices, and then identified irrelevant labels and simultaneously trained a compatible predictor based on the reconstructed label matrix. PML-LRS [11] separated the label matrix into a latent ground-truth matrix with low-rank constraint and the noisy matrix with sparse constraint, and induced the classifier with respect to the ground-truth label matrix. While PML-NI [13] assumed the noisy labels were caused by some ambiguous contents of samples and divided the predictor into a noise-free classifier and a noisy label identifier.

However, these two-stage based PML methods may be misled by erroneously elicited labels. Furthermore, when dealing with a large label space, the aforementioned ranking-based solutions (PML-fp and PML-lc) have to compute the loss for all pairwise labels, and all the other PML methods have to seek a classifier coefficient matrix with respect to all labels. All these issues compromise their efficiency and effectiveness. To address these issues, we introduce PML-*LCom* to simultaneously remove irrelevant labels by compressing labels and train a multi-label predictor in the compressed label space with much reduced coefficients.

III. The Proposed PML-*LCom* Method

A. Problem Formulation

As introduced before, PML works with a matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n] \in \mathbb{R}^{n\times d}$, which is the sample-feature data matrix for n training samples in the d-dimensional feature space, and a matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n] \in \{0,1\}^{n\times q}$, which is the observed sample-label association matrix, where q is the number of distinct labels. If $\mathbf{x}_i$ is annotated with the j-th label, $\mathbf{y}_{ij} = 1$; otherwise, $\mathbf{y}_{ij} = 0$. The main difficulty of PML is that, the multi-label classifier can only access the observed noise-corrupted sample-label association matrix $\mathbf{Y}$, other than the ground-truths, which are crucial for inducing a reliable multi-label predictor. PLL selects only one label in each row of $\mathbf{Y}$ as the ground-truth and disregards the correlations between labels. In contrast, PML has to select more than one entry in each row of $\mathbf{Y}$ as the ground-truths and these labels have diverse correlations. Therefore, PML is much more complex than PLL and MLL.

B. Label Compression

How to mine the important label correlations and train a robust multi-label predictor on partial multi-label data are the two main challenges of PML. Since the labels of multi-label samples are non-exclusive but correlated, the underlying label matrix of training samples should be low-rank [26], [28], [36]. To capture the label correlations in PML, we can factorize the observed non-negative sample-label association matrix into the product of two non-negative low-rank matrices as follows:

$$\min_{\mathbf{U}\geq 0,\mathbf{V}\geq 0} \|\mathbf{Y} - \mathbf{U}\mathbf{V}^\top\|_F^2 \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{n\times r}$ and $\mathbf{V} \in \mathbb{R}^{q\times r}$ $(r < min\{q, n\})$ are the two non-negative low-rank matrices to approximate the observed sample-label matrix $\mathbf{Y}$. Matrix $\mathbf{V}$ embeds q labels into a compressed semantic space, and matrix $\mathbf{U}$ denotes the low-rank semantic representation of n samples in the compressed space. $\mathbf{U}$ can be also viewed as storing the embedded labels of these n samples. Each of the q labels may be affected by all the r new semantic labels. As such, the high-order one-to-all label correlation is also explored by label compression, and each new semantic label also encodes the latent label correlations of original labels. However, since the observed label-matrix $\mathbf{Y}$ contains noisy information, the low-rank matrices $\mathbf{U}$ and $\mathbf{V}$ may not be reliable. To mitigate the influence of noisy labels on the factorized low-rank matrices, we introduce a sparse noisy label matrix $\mathbf{E} \in \mathbb{R}^{n\times q}$ to model the noisy information before factorizing the observed matrix $\mathbf{Y}$. The idea is implemented as follows:

$$\min_{\mathbf{U}\geq 0,\mathbf{V}\geq 0,\mathbf{E}\geq 0} \|(\mathbf{Y} - \mathbf{E}) - \mathbf{U}\mathbf{V}^\top\|_F^2 + \lambda_1\|\mathbf{E}\|_1 \tag{2}$$

Here, we use the l_1-norm regularization on $\mathbf{E}$ to extract the noisy label information concealed in the observed sample-label matrix $\mathbf{Y}$. Note that the low-rank approximation can also denoise the data matrix for the random perturbation [37], [38]. In this way, the low-rank matrices $\mathbf{U}$ and $\mathbf{V}$ are strengthened in reliability for capturing label correlations and inducing a robust classifier.

The above label compression may still suffer from the issue of sparse label annotations of multi-label training data, especially for a large label space, where each sample is only annotated with a very small subset of q labels, and each label is also assigned to a small portion of samples. To make a credible label compression, we should use the readily available feature information of training samples. If two samples are nearby in the feature space, they should have similar semantics. In other words, two samples ($\mathbf{x}_i$ and $\mathbf{x}_j$) with a high feature similarity should be annotated with similar labels and thus label embeddings ($\mathbf{u}_i$ and $\mathbf{u}_j$). To implement this, we define a smoothness term to regularize $\mathbf{U}$ as follows:

$$\min_{\mathbf{U}} \frac{1}{2}\sum_{i,j=1}^{n} \|\mathbf{u}_i - \mathbf{u}_j\|^2\mathbf{S}_{ij} = tr(\mathbf{U}^T\mathbf{L}\mathbf{U}) \tag{3}$$

where $tr(\cdot)$ is the matrix trace operator, $\mathbf{S} \in \mathbb{R}^{n\times n}$ is the sample similarity matrix, and $\mathbf{L} = \mathbf{D} - \mathbf{S}$ is the graph

Laplacian matrix with the diagonal matrix $\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{S}_{ij}$. Here, the sample similarity matrix $\mathbf{S}$ is computed as follows:

$$s_{ij} = \begin{cases} \exp(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\theta^2}), & \text{if} \quad \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where $\mathcal{N}_k(\mathbf{x}_i)$ is the set of k nearest neighbors of $\mathbf{x}_i$, and the Gaussian kernel width θ is empirically set to $\theta = \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{x}_{i_k}\|/n$, where $\mathbf{x}_{i_k}$ corresponds to the k-th nearest neighbor of $\mathbf{x}_i$. In this paper, we take the first five nearest neighbors of $\mathbf{x}_i$, i.e., $k = 5$. To ensure the symmetry of $\mathbf{S}$, we set $\mathbf{S} = \mathbf{S} + \mathbf{S}^T$.

Our label compression is different from the existing label reduction based multi-label learning algorithms, which separately apply low-rank constraints on the frequent labels and long tail labels to handle massive labels [21], or execute label space dimension reduction by principle component analysis (or by an encoder-decoder) on the label matrix (and feature data matrix) [20], [19]. In contrast, our label compression leverages non-negative matrix factorization and local manifold structure from both label and feature spaces in a coherent fashion. Our experimental results confirm this advantage.

C. Multi-label Predictor Learning

Existing PML solutions focus on inducing a multi-label predictor based on the latent ground-truth label matrix $\mathbf{Y} - \mathbf{E}$. When given a large label space, an even larger number of training samples should be provided to learn the large feature-to-label mapping coefficient matrix $\mathbf{W} \in \mathbb{R}^{q \times d}$ for a linear predictor. However, such a large number of partial multi-label training data is not readily available, and seeking such a coefficient matrix is also time-consuming [6]. Furthermore, for PML solutions working with ranking loss functions, e.g., [8], a quadratic number (q^2) of loss functions should be optimized for every pair of labels. As a result, it is too computationally expensive to induce the multi-label predictor from the latent ground-truth label matrix $\mathbf{Y} - \mathbf{E}$, especially for a large label space.

From our label compression, matrix $\mathbf{U} \in \mathbb{R}^{n \times r}$ can be viewed as the compressed latent label matrix with respect to r new labels. To achieve credible label prediction for unseen samples and to avoid a large number of feature-to-label mapping coefficients, we advocate to learn a coefficient matrix $\mathbf{W} \in \mathbf{R}^{r \times d}$ ($r \ll q$) to map samples to the compressed labels as follows:

$$\min_{\mathbf{W}} \|\mathbf{U} - \mathbf{X}\mathbf{W}^\top\|_F^2 + \lambda_2 \|\mathbf{W}\|_F^2 \tag{5}$$

where λ_2 is the trade-off parameter to control over-fitting. The multi-label predictor defined by the coefficient matrix $\mathbf{W}$ in (5) has three advantages to highlight. First, the number of coefficients to learn is significantly less than that in previous PML models, because r is much smaller than q. Second, the predictor learned in the compressed label space implicitly filter out the irrelevant labels, because the ground-truth label matrix in MLL is intrinsically low-rank. Third, it reduces the run-time as it avoids the complex singular value decomposition, which is used by PML-LRS [11] to gain the low-rank coefficient matrix $\mathbf{W}$.

D. Unified Objective Function and Optimization

Most PML solutions adopt a two-stage protocol that firstly induces the noise-free labels of training samples and then learns the predictor with respect to the induced labels [12], [10], [16]. These two-stage based solutions may face with the inconsistency between the induced labels and the predictor. In other words, the label induction and the predictor learning should collaborate to avoid such inconsistency. Given that, we coordinate label compression and coefficient matrix learning via a unified objective as follows:

$$\begin{aligned} \min_{\mathbf{U}\geq 0, \mathbf{V}\geq 0, \mathbf{E}\geq 0, \mathbf{W}} & \|(\mathbf{Y} - \mathbf{E}) - \mathbf{U}\mathbf{V}^\top\|_F^2 + \|\mathbf{U} - \mathbf{X}\mathbf{W}^\top\|_F^2 \\ & + \lambda_1 \|\mathbf{E}\|_1 + \lambda_2 \|\mathbf{W}\|_F^2 + \lambda_3 tr(\mathbf{U}^\top \mathbf{L}\mathbf{U}) \end{aligned} \tag{6}$$

The first term of (6) seeks the ground-truth labels by removing noisy labels from the observed label matrix and then compresses the labels. The second term computes the empirical squared loss on the training samples in the compressed label subspace. The last trace term uses the local geometric structure among samples to obtain better latent labels estimation. Parameters λ_1, λ_2 and λ_3 are the scalar weights for these terms. By minimizing the above objective function, PML-*LCom* can achieve the compatible compressed label matrix and the coefficient matrix, and thus a robust multi-label predictor.

For a previously unseen sample $\mathbf{x}$, PML-*LCom* predicts its association probability with respect to q labels as follows:

$$f(\mathbf{x}) = \mathbf{x}\mathbf{W}^T\mathbf{V}^T \tag{7}$$

1) Optimization: The proposed objective function of PML-*LCom* in (6) is non-convex for all variables $\mathbf{U}$, $\mathbf{V}$, $\mathbf{E}$, and $\mathbf{W}$ together. Inspired by the idea of multiplicative algorithms for NMF [39], we separately optimize $\mathbf{U}$, $\mathbf{V}$, $\mathbf{E}$ and $\mathbf{W}$ while alternatively fixing the others as constants. (6) is split into:

$$\begin{aligned} \min_{\mathbf{U}\geq 0, \mathbf{V}\geq 0, \mathbf{E}\geq 0} & \|\mathbf{Y} - \mathbf{U}\mathbf{V}^\top - \mathbf{E}\|_F^2 + \|\mathbf{U} - \mathbf{X}\mathbf{W}^\top\|_F^2 \\ & + \lambda_1 \|\mathbf{E}\|_1 + \lambda_3 tr(\mathbf{U}^\top \mathbf{L}\mathbf{U}) \end{aligned} \tag{8}$$

$$\min_{\mathbf{W}} \|\mathbf{U} - \mathbf{X}\mathbf{W}^\top\|_F^2 + \lambda_2 \|\mathbf{W}\|_F^2 \tag{9}$$

The detailed optimization procedure is presented below.

A. Updating $\{\mathbf{U}, \mathbf{V}, \mathbf{E}\}$ with $\mathbf{W}$ fixed.

We optimize (8) in an alternative way which is widely used in standard NMF [39].

1) **Updating $\mathbf{U}$**: When $\mathbf{V}$, $\mathbf{E}$ and $\mathbf{W}$ are fixed, (8) reduces to:

$$\begin{aligned} \min \Theta_1(\mathbf{U}) \quad & \text{s.t. } \mathbf{U} \geq 0 \\ \Theta_1(\mathbf{U}) = & \|\mathbf{Y} - \mathbf{U}\mathbf{V}^\top - \mathbf{E}\|_F^2 + \|\mathbf{U} - \mathbf{X}\mathbf{W}^\top\|_F^2 \\ & + \lambda_3 tr(\mathbf{U}^\top \mathbf{L}\mathbf{U}) \end{aligned} \tag{10}$$

The derivative of $\Theta_1(\mathbf{U})$ w.r.t. $\mathbf{U}$ is:

$$\frac{\partial \Theta_1}{\partial \mathbf{U}} = (\mathbf{U}\mathbf{V}^\top + \mathbf{E} - \mathbf{Y})\mathbf{V} + \mathbf{U} - \mathbf{X}\mathbf{W}^\top + \lambda_3 \mathbf{L}\mathbf{U} \tag{11}$$

Algorithm 1 PML-*LCom*: Partial Multi-label Learning using Label Compression.

Input:
$\mathbf{X}$: $n \times d$ sample-feature data matrix;
$\mathbf{Y}$: $n \times q$ sample-label association matrix;
λ_1, λ_2, λ_3: scalar parameters.
Output: U, V, E, W

```
Construct sample similarity matrix S through (4);
Randomly initialize U, V, E, W;
Do:
    Fix W, update U, V, E by optimizing (8);
    Seek the optimal W by solving (9);
While not converged or within the allowed number of iterations
```

Using the Karush-Kuhn-Tucker (KKT) condition [40] for the non-negativity of $\mathbf{U}$, we can update $\mathbf{U}$ by:

$$\mathbf{U}_{ij} \leftarrow \mathbf{U}_{ij} \frac{(\mathbf{YV} + (\mathbf{XW}^\top)^+ + \lambda_3 \mathbf{M}^-)_{ij}}{(\mathbf{UV}^\top \mathbf{V} + \mathbf{EV} + \mathbf{U} + (\mathbf{XW}^\top)^- + \lambda_3 \mathbf{M}^+)_{ij}} \tag{12}$$

where $\mathbf{M} = \mathbf{LU}$. We define for any matrix $\mathbf{O} = \mathbf{O}^+ - \mathbf{O}^-$, where the matrices with positive and negative symbols are defined as $\mathbf{O}^+ = \frac{|\mathbf{O}|+\mathbf{O}}{2}$, and $\mathbf{O}^- = \frac{|\mathbf{O}|-\mathbf{O}}{2}$.

2) **Updating** $\mathbf{V}$: With $\mathbf{U}$, $\mathbf{E}$ and $\mathbf{W}$ fixed, (8) reduces to:

$$\begin{aligned} &\min \Theta_2(\mathbf{V}) \quad \text{s.t. } \mathbf{V} \geq 0 \\ &\Theta_2(\mathbf{V}) = \|\mathbf{Y} - \mathbf{UV}^\top - \mathbf{E}\|_F^2 \end{aligned} \tag{13}$$

The derivative of $\Theta_2(\mathbf{V})$ w.r.t. $\mathbf{V}$ is:

$$\frac{\partial \Theta_2}{\partial \mathbf{V}} = (\mathbf{VU}^\top - \mathbf{Y}^\top + \mathbf{E}^\top)\mathbf{U} \tag{14}$$

Using the KKT condition, we can update $\mathbf{V}$ using the following rule:

$$\mathbf{V}_{ij} \leftarrow \mathbf{V}_{ij} \frac{(\mathbf{Y}^\top \mathbf{U})_{ij}}{(\mathbf{VU}^\top \mathbf{U} + \mathbf{E}^\top \mathbf{U})_{ij}} \tag{15}$$

3) **Updating** $\mathbf{E}$: When $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{W}$ are fixed, (8) reduces to:

$$\begin{aligned} &\min \Theta_3(\mathbf{E}) \quad \text{s.t. } \mathbf{E} \geq 0 \\ &\Theta_3(\mathbf{E}) = \|\mathbf{Y} - \mathbf{UV}^\top - \mathbf{E}\|_F^2 + \lambda_1 \|\mathbf{E}\|_1 \end{aligned} \tag{16}$$

Let $f(\mathbf{E}) = \|\mathbf{Y} - \mathbf{UV}^\top - \mathbf{E}\|_F^2$, then we can use the shrinkage operator [41] to find the optimal parameter $\mathbf{E}^*$:

$$\mathbf{E}^* = argmin_{\mathbf{E}} \lambda_1 \|\mathbf{E}\|_1 + f(\mathbf{E}) \tag{17}$$

Let $\mathbf{L} = \mathbf{E} - \frac{1}{l}\nabla f(\mathbf{E})$, where l is the Lispschitz constant, then the closed solution can be obtained as [42]:

$$\forall i \in [n], j \in [q] \quad E_{ij}^* = \begin{cases} L_{ij} - \lambda_1/l, & L_{ij} > \lambda_1/l \\ 0, & |L_{ij}| \leq \lambda_1/l \end{cases} \tag{18}$$

B. Updating $\mathbf{W}$ with $\{\mathbf{U}, \mathbf{V}$ and $\mathbf{E}\}$ are fixed.

The optimization of (9) is a typical ridge regression problem which can be effectively computed by the Accelerated Gradient Decent (AGD) algorithm [43]. The gradient of the objective function w.r.t. $\mathbf{W}$ is:

$$\nabla_{\mathbf{W}} = \lambda_2 \mathbf{W} - \mathbf{U}^\top \mathbf{X} + \mathbf{W}\mathbf{X}^\top \mathbf{X} \tag{19}$$

The overall procedure of PML-*LCom* is summarized in Algorithm 1.

The computational costs of PML- *LCom* are dominated by matrix multiplication. In each iteration, given the sample matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the sample-label association matrix $\mathbf{Y} \in \{0,1\}^{n \times q}$, the computational complexity for updating $\mathbf{U}$ is $\mathcal{O}(nqr + ndr + n^2 r)$. Similarly, the computation complexity for updating $\mathbf{V}$ and $\mathbf{E}$ are $\mathcal{O}(nqr)$ and $\mathcal{O}(nqr)$, respectively. The computation complexity for updating $\mathbf{W}$ is $\mathcal{O}(ndr)$. Hence, the total complexity for optimizing PML-*LCom* in t iterations is $\mathcal{O}(t(nqr + ndr + n^2 r))$.

TABLE I: Statistical information of multi-label datasets used for evaluation.

Dataset	#samples	#Features	#Labels	Domain
delicious	16105	500	983	text
bibtex	7395	1836	159	text
medical	978	1449	45	text
Corel5k	5000	499	374	images
CAL500	502	68	174	music
slashdot	3782	1079	22	biology
YeastCC	6139	6139	50	biology
YeastMF	6139	6139	39	biology
mirflickr	10433	100	7	image

IV. Experiments

A. Experimental Setup

We assess the effectiveness of PML-*LCom* on nine public and benchmark multi-label datasets, including synthetic as well as real-world PML ones [9], [10]. Table I summarizes the number of samples, of classes, and the domains for these datasets. Note that datasets with at most 15 labels were used for experiments in [8], [11], since they can not deal with too many labels. For the first six datasets, we assume the collected labels are noise-free and follow the typical setting in [8], [11] to create partial multi-label annotations for the training samples. Specifically, for each training sample $\mathbf{x}_i$, we randomly inject $\lfloor a \times |\mathbf{y}_i| \rfloor$ noisy labels with $a \in \{0.5, 1, 2\}$, where $\lfloor \cdot \rfloor$ means round down to the nearest integer. We adopt the widely-used multi-label learning evaluation metrics: *hamming loss*, *ranking loss*, *one error*, *coverage* and *average precision*. Their formal definitions can be found in [2], [1]. For the first four evaluation metrics, the smaller the value is, the better the performance is. While for the last evaluation metric, the opposite trend holds.

To reach a comprehensive comparison, we compare the performance of PML-*LCom* against three state-of-the-art PML methods, PML-fp [8], PML-LRS [11], and DRAMA [16]; two label compression based MLL methods, REML [21] and CPLST [19]); and the classical ML-KNN [22]. For the compared algorithms, we adopt the parameters suggested in the original papers and/or codes. Specifically, for PML-fp C_1 is set to 1, and C_3 is set to 10. For PML-LRS, we take $\gamma = 0.01$, $\beta = 0.1$ and $\eta = 1$ as suggested in their paper.

For DRAMA, we adopt $\delta_1 = 0.01$ and $\delta_2 = 0.5$ for each experiment dataset. For REML, we set $\lambda_1 = 0.1$, $\lambda_2 = 0.1$ and $\lambda_3 = 0.01$. We take the first 5 principal components for CPLST and the number of nearest neighbors of ML-KNN is set to default 10. To simplify the implementation, we fix $r = \lfloor 0.6 * q \rfloor$, $\lambda_1 = 5$, $\lambda_2 = 10$, and $\lambda_3 = 0.1$ for the proposed PML-*LCom*. The sensitivity of these parameters will be studied later. PML-*LCom* may achieve a better performance on these datasets when the parameters are fined tuned. The codes of PML-*LCom* will be made public later.

B. Experimental Results of Classification Performance

We randomly partition samples of each dataset into a training set (70%) and a testing set (30%), and independently repeat the above partition 10 times. Due to page limit, we only report the average results and standard deviations in terms of *ranking loss* and *average precision* in Table II-III. The similar results can be observed in other evaluation metrics, and the global comparison is summarized in Table IV. The computation of PML-fp on the delicious and Corel5k datasets are too time-consuming to be completed, and thus the results are not reported.

From the Table, we have the following observations:
(i) **Compressed** vs. **Original** labels: CPLST, REML and PML-*LCom* resort to different techniques to compress labels, and they generally get better results than PML-LRS, which also adopts a linear classifier, similar as these label compression based solutions. This facts the effectiveness of label compression. Even with a similar linear predictor, PML-*LCom* achieves a better performance than CPLST and REML, which only use feature information via a sample-wise least square loss (alike (5)), while PML-*LCom* uses the local manifold structure of samples to guide the label compression. For the similar reasons, PML-fp and DRAMA, which train the predictor in the original label space, are often lost to PML-*LCom*.
(ii) **With** vs. **Without modeling noisy labels**: ML-KNN, CPLST and REML do not explicitly consider noisy labels of training data, they all lose to PML-*LCom*, which explicitly accounts for noisy labels. This contrast implies the necessity of modeling noisy labels. Although PML-LRS, DRAMA and PML-fp consider noisy labels of training data, they do not manifest a clear advantage to ML-KNN, CPLST and REML. One possible cause is that ML-KNN has a flavor of averaging-based disambiguation of noisy labels, CPLST factorizes the label matrix, while REML adopts two low-rank linear predictors for frequent labels and long tail labels, respectively. These techniques can alleviate the impact of noisy labels.
(iii) **Separate** vs. **Unified** models: DRAMA follows a two-stage strategy that firstly elicits the ground-truth labels and then induces the multi-label predictor on elicited labels. It often loses to the proposed PML-*LCom*, which simultaneously achieves the elicited labels and predictor by a unified objective function. This comparison shows the risk of separating label elicitation from the follow-up predictor of two-stage based approaches. However, there is no clear pattern between PML-LRS and PML-fp, which also elicit labels and induce a multi-label predictor in a unified objective function. A possible explanation is that these compared one-stage solutions operate in the original space, and they need a large (often under-deterministic) feature-to-label coefficient matrix for a large label space, while PML-*LCom* learns a smaller feature-to-label coefficient matrix.

Table IV summarizes the win/tie/loss counts of PML-*LCom* against each other compared method over 21 (6 datasets $\times$ 3 settings + 3 real datasets) settings across 5 evaluation metrics. We can observe that among 105 pairwise comparisons, PML-*LCom* significantly outperforms the popular MLL approaches ML-KNN, CPLST and REML in 92.38%, 90.48% and 96.19% cases. PML-*LCom* also achieves much better results compared with popular PML methods PML-LRS, DRAMA, and PML-fp in 92.38%, 88.57%, 85.33% cases. Overall, these results prove the necessity to account for noisy labels and the effectiveness of label compression for learning on PML data with large label spaces.

C. Identification of noisy labels

We conduct additional experiments to investigate the performance of PML-*LCom* and compared methods on identifying the noisy labels of training data. We report the results of each compared method on three real-world PML datasets with five evaluation metrics in Table VI. From Table VI, we can observe the followings: (i) PML-*LCom* outperforms REML on all real-world datasets under across evaluation metrics, as REML does not consider the negative impact of noisy labels. (ii) PML-*LCom* performs better than ML-KNN, CPLST and PML-LRS in most cases, CPLST achieves better results than PML-*LCom* in terms of *one error*, this is because CPLST learns the encoder matrix via the shifted label matrix resulting the most plausible label ahead of other labels, while can not manage a good ranking of relevant labels. (iii) PML-fp and DRAMA give better results than PML-$LCom$ on the Mirflickr dataset, this is because PML-fp and DRAMA use more complex strategies to mine the feature information for denoiseing labels. However, PML-fp has a lower performance than our proposed method on the other two real-world datasets as the biological data with high noisy information. What is more, the smoothness assumption of DRAMA is broken on YeastCC and YeastMF datasets, which are sparse but noisy protein-protein interaction networks. For this reason, DRAMA can not find a solution on these two datasets and its resutls on these two datsets are also not reported. Overall, this study confirms that PML-*LCom* can effectively identify noisy labels concealed in the candidate label set.

D. Model Efficiency by Run-time Analysis

Table V presents the run-time for model training and for predicting unlabeled instances of compared methods[1]. We observe that: (i) ML-KNN, CPLST, and REML are faster than PML-*LCom*. That is mainly because they do not explicitly account for noisy labels but directly perform k nearest neighborhood

[1]All methods are implemented with Matlab2014a on a server (CentOS 6.9, Intel E5-2678v3, 256GB RAM).

TABLE II: Experiment results on different datasets in terms of *ranking loss*. •/○ indicates whether PML-*LCom* is statistically (according to pairwise t-test at 95% significance level) superior/inferior to the other method.

Datasets	$a*100\%$	ML-KNN	CPLST	REML	PML-LRS	DRAMA	PML-fp	PML-*LCom*
delicious	0.5	0.135±0.000•	0.122±0.003•	0.186±0.002•	0.186±0.002•	0.148±0.000•	–	0.118±0.001
	1	0.139±0.000•	0.132±0.001•	0.192±0.003•	0.191±0.002•	0.151±0.000•	–	0.124±0.000
	2	0.144±0.001•	0.133±0.000•	0.201±0.002•	0.201±0.001•	0.162±0.001•	–	0.129±0.000
bibtex	0.5	0.227±0.006•	0.206±0.004•	0.155±0.005•	0.174±0.005•	0.136±0.000•	0.209±0.007•	0.119±0.004
	1	0.239±0.004•	0.210±0.003•	0.168±0.002•	0.145±0.003•	0.153±0.000•	0.219±0.004•	0.116±0.006
	2	0.245±0.004•	0.216±0.003•	0.171±0.003•	0.160±0.002•	0.173±0.000•	0.409±0.003•	0.130±0.004
medical	0.5	0.060±0.006•	0.129±0.014•	0.083±0.009•	0.284±0.043•	0.043±0.002•	0.026±0.003○	0.039±0.011
	1	0.063±0.007•	0.124±0.005•	0.077±0.008•	0.312±0.020•	0.051±0.000•	0.031±0.004	0.035±0.006
	2	0.086±0.009•	0.132±0.012•	0.075±0.010•	0.253±0.006•	0.054±0.001•	0.033±0.006	0.034±0.004
Corel5k	0.5	0.141±0.000○	0.168±0.003○	0.211±0.007	0.253±0.003•	0.276±0.001•	–	0.209±0.003
	1	0.148±0.003○	0.163±0.004○	0.220±0.003•	0.243±0.005•	0.301±0.001•	–	0.216±0.002
	2	0.160±0.004○	0.181±0.003○	0.242±0.005•	0.229±0.004	0.299±0.002•	–	0.226±0.004
CAL500	0.5	0.201±0.004•	0.207±0.006•	0.250±0.007•	0.192±0.005•	0.300±0.016•	0.196±0.007•	0.183±0.004
	1	0.219±0.004•	0.221±0.005•	0.267±0.004•	0.253±0.006•	0.340±0.006•	0.222±0.007•	0.186±0.002
	2	0.244±0.006•	0.249±0.002•	0.308±0.008•	0.308±0.003•	0.362±0.007•	0.222±0.003•	0.197±0.005
slashdot	0.5	0.189±0.006•	0.193±0.004•	0.183±0.005•	0.199±0.007•	0.110±0.000○	0.142±0.004•	0.122±0.007
	1	0.201±0.006•	0.212±0.007•	0.203±0.007•	0.227±0.004•	0.162±0.000•	0.159±0.006	0.143±0.008
	2	0.216±0.007•	0.252±0.012•	0.248±0.013•	0.273±0.013•	0.174±0.001•	0.181±0.004•	0.168±0.002
YeastCC		0.309±0.009•	0.384±0.012•	0.320±0.010•	0.315±0.010•	0.283±0.008•	0.308±0.006•	0.198±0.006
YeastMF		0.326±0.008•	0.401±0.010•	0.425±0.013•	0.398±0.011•	0.298±0.006•	0.342±0.008•	0.276±0.009
mirflickr		0.177±0.016•	0.159±0.006•	0.250±0.010•	0.218±0.004•	0.098±0.006○	0.115±0.008	0.111±0.007

TABLE III: Experiment results on different datasets in terms of *average precision*. •/○ indicates whether PML-*LCom* is statistically (according to pairwise t-test at 95% significance level) superior/inferior to the other method.

Datasets	$a*100\%$	ML-KNN	CPLST	REML	PML-LRS	DRAMA	PML-fp	PML-*LCom*
delicious	0.5	0.323±0.002•	0.329±0.003•	0.344±0.002•	0.345±0.003•	0.334±0.000•	–	0.364±0.002
	1	0.321±0.001•	0.296±0.003•	0.341±0.004•	0.344±0.002•	0.330±0.000•	–	0.362±0.002
	2	0.315±0.002•	0.296±0.002•	0.332±0.002•	0.332±0.002•	0.326±0.000•	–	0.359±0.001
bibtex	0.5	0.314±0.002•	0.297±0.004•	0.360±0.008•	0.474±0.007•	0.422±0.002•	0.393±0.002•	0.545±0.004
	1	0.301±0.004•	0.295±0.004•	0.356±0.005•	0.509±0.004•	0.371±0.001•	0.319±0.003•	0.533±0.009
	2	0.281±0.005•	0.293±0.006•	0.365±0.005•	0.474±0.007•	0.319±0.002•	0.245±0.006•	0.508±0.004
medical	0.5	0.770±0.015•	0.599±0.021•	0.759±0.026•	0.487±0.049•	0.837±0.005•	0.854±0.004	0.853±0.012
	1	0.763±0.015•	0.613±0.011•	0.746±0.014•	0.433±0.012•	0.831±0.004•	0.846±0.004•	0.863±0.012
	2	0.733±0.014•	0.599±0.013•	0.754±0.012•	0.524±0.006•	0.801±0.006•	0.832±0.007•	0.865±0.010
Corel5k	0.5	0.238±0.001•	0.267±0.007•	0.266±0.009•	0.269±0.004•	0.203±0.001•	–	0.289±0.004
	1	0.231±0.005•	0.264±0.003•	0.262±0.006•	0.266±0.003•	0.176±0.000•	–	0.281±0.004
	2	0.229±0.006•	0.256±0.004•	0.262±0.002•	0.256±0.004•	0.171±0.004•	–	0.276±0.003
CAL500	0.5	0.477±0.007•	0.483±0.009•	0.441±0.002•	0.473±0.006•	0.387±0.021•	0.441±0.006•	0.497±0.004
	1	0.463±0.004•	0.478±0.006•	0.410±0.006•	0.436±0.010•	0.338±0.005•	0.404±0.003•	0.498±0.006
	2	0.436±0.008•	0.452±0.004•	0.354±0.006•	0.358±0.009•	0.317±0.006•	0.416±0.005•	0.489±0.009
slashdot	0.5	0.461±0.012•	0.546±0.006•	0.598±0.010•	0.559±0.009•	0.685±0.000○	0.631±0.004•	0.675±0.011
	1	0.445±0.007•	0.528±0.009•	0.546±0.009•	0.499±0.006•	0.610±0.000•	0.609±0.003•	0.635±0.009
	2	0.438±0.017•	0.492±0.015•	0.467±0.011•	0.412±0.012•	0.588±0.001•	0.554±0.004•	0.595±0.009
YeastCC		0.374±0.008•	0.274±0.013•	0.321±0.011•	0.385±0.009•	0.190±0.004•	0.414±0.003•	0.544±0.011
YeastMF		0.319±0.007•	0.249±0.010•	0.211±0.007•	0.301±0.006•	0.165±0.009•	0.385±0.006•	0.418±0.013
mirflickr		0.693±0.004•	0.810±0.007•	0.679±0.006•	0.674±0.003•	0.831±0.000○	0.806±0.013•	0.823±0.007

TABLE IV: Win/Tie/Lose counts (pairwise t-test at 95% signification level) between PML-*LCom* and each compared method.

	PML-*LCom* against					
	ML-KNN	CPLST	REML	PML-LRS	DARAM	PML-fp
hamming loss	18/1/2	17/2/2	18/0/3	15/2/4	15/1/5	11/2/2
ranking loss	18/0/3	20/0/1	20/1/0	20/1/0	19/0/2	10/4/1
one error	21/0/0	18/1/3	21/0/0	21/0/0	20/0/1	15/0/0
coverage	19/1/1	20/0/1	21/0/0	20/0/1	20/0/1	14/0/1
average precision	21/0/0	21/0/0	21/0/0	21/0/0	19/0/2	14/1/0
Total (Win/Tie/Lose)	97/2/6	95/3/7	101/1/3	97/3/5	93/1/11	64/7/4

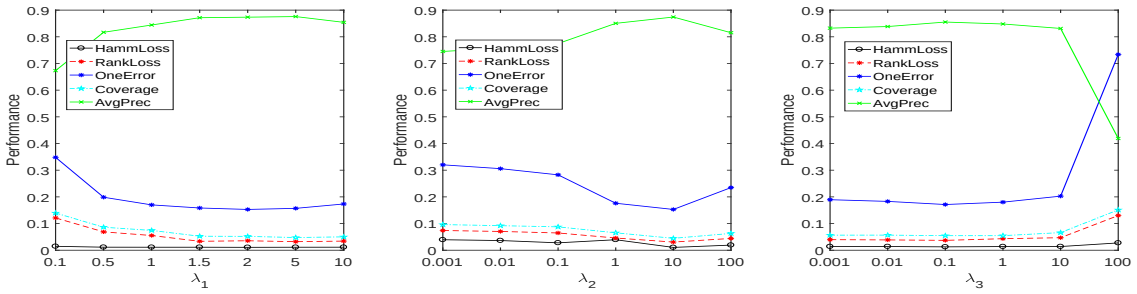
classification or matrix factorization, while PML-*LCom* needs to calculate the feature similar matrix and to model the noisy information explicitly. (ii) Although both PML-LRS and PML-*LCom* adopt a linear predictor, PML-LRS is more time-consuming than PML-*LCom*, as it works in the original label space to learn the coefficient matrix and applies singular value

TABLE V: Comparison of run-time on different datasets (in seconds). The best performer among all PML methods is bolded.

	Methods	delicious	bibtex	medical	Corel5k	CAL500	slashdot	YeastCC	YeastMF	mirflickr	Total
MLL	ML-KNN	415	21	1	31	2	3	12	12	25	522
	CPLST	341	16	1	18	2	1	16	16	3	414
	REML	1016	65	21	78	8	29	19	19	13	1268
PML	PML-LRS	**2201**	998	276	1624	285	582	9765	11480	53	27264
	DRAMA	19502	4323	25	747	235	312	3846	7521	115	36626
	PML-fp	–	62667	408	–	6394	9664	305389	253789	7704	>646015
	PML-*LCom*	3169	**49**	**3**	**116**	**11**	**4**	**95**	**98**	**26**	**3571**

TABLE VI: Transductive experimental performance of all methods on five evaluation metrics. •/○ indicates whether PML-*LCom* is statistically (according to pairwise t-test at 95% significance level) superior/inferior to the other method.

Metric	ML-KNN	CPLST	REML	PML-LRS	DRAMA	PML-fp	PML-*LCom*
	YeastCC						
hamming loss	0.023±0.000•	0.015±0.000•	0.027±0.000•	0.003±0.000	–	0.018±0.006•	0.002±0.000
ranking loss	0.036±0.015•	0.685±0.006•	0.281±0.009•	0.024±0.000•	–	0.267±0.004•	0.001±0.000
one error	0.135±0.021•	0.002±0.000○	0.709±0.010•	0.076±0.000•	–	0.023±0.004	0.022±0.006
coverage	0.024±0.007•	0.206±0.004•	0.140±0.004•	0.026±0.000•	–	0.212±0.006•	0.008±0.000
average precision	0.874±0.018•	0.363±0.004•	0.365±0.010•	0.932±0.000•	–	0.635±0.009•	0.984±0.003
	YeastMF						
hamming loss	0.025±0.000•	0.014±0.000•	0.026±0.000•	0.004±0.000	–	0.017±0.008•	0.003±0.000
ranking loss	0.041±0.006•	0.683±0.007•	0.359±0.003•	0.075±0.000•	–	0.157±0.008•	0.001±0.000
one error	0.268±0.053•	0.007±0.011○	0.800±0.010•	0.176±0.000•	–	0.039±0.021•	0.027±0.006
coverage	0.022±0.003•	0.163±0.004•	0.160±0.002•	0.037±0.000•	–	0.127±0.003•	0.001±0.000
average precision	0.793±0.032•	0.382±0.007•	0.271±0.007•	0.842±0.000•	–	0.773±0.005•	0.983±0.003
	mirflickr						
hamming loss	0.212±0.004	0.217±0.001	0.258±0.000•	0.226±0.000•	0.167±0.000○	0.226±0.001•	0.214±0.001
ranking loss	0.171±0.012•	0.306±0.001•	0.246±0.010•	0.130±0.000•	0.083±0.004○	0.105±0.006○	0.115±0.015
one error	0.448±0.135•	0.145±0.001○	0.473±0.064•	0.243±0.000○	0.188±0.009○	0.277±0.034○	0.301±0.014
coverage	0.259±0.001•	0.301±0.000•	0.307±0.002•	0.251±0.000•	0.196±0.005○	0.214±0.003○	0.225±0.012
average precision	0.700±0.003•	0.651±0.001•	0.647±0.003•	0.793±0.000	0.850±0.008○	0.810±0.012○	0.792±0.023



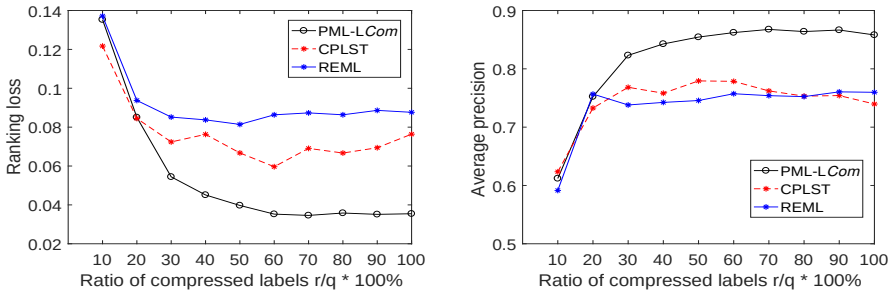
(a) Performance vs. λ_1. (b) Performance vs. λ_2. (c) Performance vs. λ_3.

Fig. 2: Performance under different input values of λ_1, λ_2 and λ_3 on the *medical* dataset.

decomposition on the coefficient matrix, both of which are quite costly, while PML-*LCom* only needs the more efficient matrix multiplication. (iii) As expected, DRAMA and PML-fp are much more time-consuming than PML-*LCom*, since PML-fp needs to maintain pair-wise label correlations (rankings) in the original label space during optimization. DRAMA adopts CART (Classification And Regression Trees) as basic regressor and maintains label correlations among regressors by augmenting the feature space with the elicited label confidence matrix in each boosting round. While our PML-*LCom* can capture the global label correlations via compressed labels and optimize the predictor with respect to compressed labels, with a much smaller coefficient matrix. (iv) PML-*LCom* not only manifests an efficient PML method on datasets with a large label space (> 150 labels, e.g., *Corel5k* and *delicious*), but also on datasets with a moderate label space ($20 \sim 50$ labels, e.g., *slashdot* and *medical*). In summary, the results in Tables II-VI confirm the advantage of label compression for improving the efficiency and effectiveness of PML with large label spaces.

E. Parameter Sensitivity and Convergence Analysis

We first study the sensitivity of parameter λ_1, λ_2, and λ_3 in (6) on 'medical' dataset with $a = 0.5$, by varying one parameter while fixing the other two. The similar results can be found on other datasets. The experiment results of five evaluation metrics are shown in Fig. 2. From Fig. 2(a), we find that a too small λ_1 down-grades the performance. This



(a) Ranking loss vs. r. (b) Average precision vs. r.

Fig. 3: Performance vs. r (size of compressed labels) on the *medical* dataset.

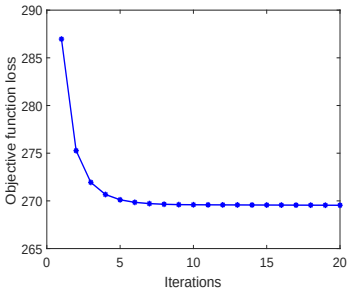


Fig. 4: The convergence trend of PML-*LCom* on the *medical* dataset.

is because when λ_1 is too small (i.e. $\lambda_1 = 0.1$), we do not fully consider the noisy information hidden in the sample-label association matrix, thus, the performance is decreased. This observation indicates the benefits of using the l_1-norm to regularize the sparse noisy label matrix. Parameter λ_2 controls the complexity a linear predictor, when λ_2 is too small/large, PML-*LCom* manifests a degraded performance. That is because a too small λ_2 under-regularizes the predictor, while a too large λ_2 over-regularizes the linear predictor. Similarly, a too large value of λ_3 leads to a lower performance by over-weighting the similarity between samples to coordinate the label compression. In practice, we also observed that when λ_3 is too small, the performance of PML-*LCom* is declined. Overall, the terms balanced by these three parameters indeed help PML-*LCom* to handle the multiplicity of PML on datasets. PML-*LCom* can be effective in a wide-range of input parameters. Based on the above observation, PML-*LCom* adopts $\lambda_1 = 5$, $\lambda_2 = 10$ and $\lambda_3 = 0.1$ in the evaluation experiments.

We investigate the impact of dimension (r) of compressed labels with $a = 0.5$. Furthermore, as CPLST and REML also use the idea of label compression, we study the impact of r on them as well. Fig. 3 shows the effect of varying r on medical dataset. We can observe that, when the labels are over compressed, the performance of all compared based methods are poor as the heavily compressed label matrix cannot capture enough label information. The performance is improved as r increases among all three compared methods. PML-*LCom* usually achieves better results than the other two compared methods as it directly models the noisy information and utilizes feature information to guide the compression process. When the ratio reaches to 60%($r \approx 0.6q$), the performance becomes stable and even better (or comparable) than $r \approx q$. This facts that label compression does not compromise the performance but improves the efficiency. Given the above observation, we adopt $r = \lfloor 0.6 * q \rfloor$ in our study.

We also empirically study the convergence trend of PML-*LCom*. Fig. 4 reveals the convergence curve of PML-*LCom* on the *medical* dataset. We can observe the objective function loss of PML-*LCom* gradually decreases as the number of iterations increases, and it reaches to convergence within fifteen iterations. In practice, PML-*LCom* also quickly converges on other used datasets.This observation shows our alternative optimization procedure for solving the objective function of PML-*LCom* can quickly convege.

V. Conclusions and Future Work

Existing PML learning methods generally suffer from the high computational cost in large label spaces. To address this issue, we proposed a label compression based approach PML-*LCom* that simultaneously compresses multiple labels and induces a multi-label predictor to these compressed labels. Experiments on diverse datasets show that PML-*LCom* can more effectively and efficiently learn from multi-label data with noisy labels than existing methods. We will study PML on multi-view data with unlabeled samples in the future.

References

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

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

[3] Z.-M. Chen, X.-S. Wei, P. Wang, and Y. Guo, "Multi-label image recognition with graph convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5177–5186.

[4] G. Yu, C. Domeniconi, H. Rangwala, G. Zhang, and Z. Yu, "Transductive multi-label ensemble classification for protein function prediction," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2012, pp. 1077–1085.

[5] G. Yu, H. Rangwala, C. Domeniconi, G. Zhang, and Z. Yu, "Protein function prediction with incomplete annotations," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 11, no. 3, pp. 579–591, 2013.

[6] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang, "Deep learning for extreme multi-label text classification," in *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 115–124.

[7] G. Yu, C. Lu, and J. Wang, "Nogoa: predicting noisy go annotations using evidences and sparse representation," *BMC Bioinformatics*, vol. 18, p. 350, 2017.

[8] M.-K. Xie and S.-J. Huang, "Partial multi-label learning," in *AAAI Conference on Artificial Intelligence*, 2018, pp. 4302–4309.

[9] G. Yu, X. Chen, C. Domeniconi, J. Wang, Z. Li, Z. Zhang, and X. Wu, "Feature-induced partial multi-label learning," in *IEEE International Conference on Data Mining*, 2018, pp. 1398–1403.

[10] J.-P. Fang and M.-L. Zhang, "Partial multi-label learning via credible label elicitation," in *AAAI Conference on Artificial Intelligence*, 2019, pp. 3518–3525.

[11] L. Sun, S. Feng, T. Wang, C. Lang, and Y. Jin, "Partial multi-label learning by low-rank and sparse decomposition," in *AAAI Conference on Artificial Intelligence*, 2019, pp. 5016–5023.

[12] Z.-S. Chen, X. Wu, Q.-G. Chen, Y. Hu, and M.-l. Zhang, "Multi-view partial multi-label learning with graph-based disambiguation," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 3553–3560.

[13] M.-K. Xie and S.-J. Huang, "Partial multi-label learning with noisy label identification," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 6454–6461.

[14] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *Journal of Machine Learning Research*, vol. 12, no. 5, pp. 1501–1536, 2011.

[15] M.-L. Zhang, B.-B. Zhou, and X.-Y. Liu, "Partial label learning via feature-aware disambiguation," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2016, pp. 1335–1344.

[16] H. Wang, W. Liu, Y. Zhao, C. Zhang, T. Hu, and G. Chen, "Discriminative and correlative partial multi-label learning," in *International Joint Conference on Artificial Intelligence*, 2019, pp. 3691–3697.

[17] N. Xu, Y.-P. Liu, and X. Geng, "Partial multi-label learning with label distribution," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 1–8.

[18] T. Yu, G. Yu, J. Wang, and M. Guo, "Partial multi-label learning with label and feature collaboration," in *International Conference on Database Systems for Advanced Applications*, 2020, pp. 1–17.

[19] Y.-N. Chen and H.-T. Lin, "Feature-aware label space dimension reduction for multi-label classification," in *Advances in Neural Information Processing Systems*, 2012, pp. 1529–1537.

[20] F. Tai and H.-T. Lin, "Multilabel classification with principal label space transformation," *Neural Computation*, vol. 24, no. 9, pp. 2508–2542, 2012.

[21] C. Xu, D. Tao, and C. Xu, "Robust extreme multi-label learning," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD*, 2016, pp. 1275–1284.

[22] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

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

[24] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079–1089, 2010.

[25] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, "Multi-label learning with weak label," in *AAAI Conference on Artificial Intelligence*, 2010, pp. 593–598.

[26] L. Xu, Z. Wang, Z. Shen, Y. Wang, and E. Chen, "Learning low-rank label correlations for multi-label classification with missing labels," in *IEEE International Conference on Data Mining*, 2014, pp. 1067–1072.

[27] G. Yu, C. Domeniconi, H. Rangwala, and G. Zhang, "Protein function prediction using dependence maximization," in *European Conference on Machine Learning and Knowledge Discovery in Databases*, 2013, pp. 574–589.

[28] H.-F. Yu, P. Jain, P. Kar, and I. Dhillon, "Large-scale multi-label learning with missing labels," in *International Conference on Machine Learning*, 2014, pp. 593–601.

[29] Z. Li and J. Tang, "Weakly supervised deep matrix factorization for social image understanding," *IEEE Transactions on Image Processing*, vol. 26, no. 1, pp. 276–288, 2016.

[30] Q. Tan, G. Yu, C. Domeniconi, J. Wang, and Z. Zhang, "Incomplete multi-view weak-label learning." in *International Joint Conference on Artificial Intelligence*, 2018, pp. 2703–2709.

[31] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma, "Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising," in *World Wide Web Conference*, 2018, pp. 993–1002.

[32] R. Babbar and B. Schölkopf, "Dismec: Distributed sparse machines for extreme multi-label classification," in *ACM International Conference on Web Search and Data Mining*, 2017, pp. 721–729.

[33] L. Liu and T. G. Dietterich, "A conditional multinomial mixture model for superset label learning," in *Advances in Neural Information Processing Systems*, 2012, pp. 548–556.

[34] F. Yu and M.-L. Zhang, "Maximum margin partial label learning," in *Asian Conference on Machine Learning*, 2016, pp. 96–111.

[35] C. Lu, J. Wang, Z. Zhang, P. Yang, and G. Yu, "Noisygoa: Noisy go annotations prediction using taxonomic and semantic similarity," *Computational Biology and Chemistry*, vol. 65, pp. 203–211, 2016.

[36] J. Tu, G. Yu, C. Domeniconi, J. Wang, G. Xiao, and M. Guo, "Multi-label answer aggregation based on joint matrix factorization," in *IEEE International Conference on Data Mining*, 2018, pp. 517–526.

[37] K. Konstantinides, B. Natarajan, and G. S. Yovanof, "Noise estimation and filtering using block-based singular value decomposition," *IEEE Transactions on Image Processing*, vol. 6, no. 3, pp. 479–483, 1997.

[38] D. Meng and F. De La Torre, "Robust matrix factorization with unknown noise," in *IEEE International Conference on Computer Vision*, 2013, pp. 1337–1344.

[39] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, 2000, pp. 535–541.

[40] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[41] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[42] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[43] Y. Nesterov, "Gradient methods for minimizing composite functions," *Mathematical Programming*, vol. 140, no. 1, pp. 125–161, 2013.