A Label Distribution Manifold Learning Algorithm

Chao Tan[a], Sheng Chen[b], Xin Geng[c], Genlin Ji[a]

[a]*School of Computer Science and Technology, Nanjing Normal University, Nanjing 210023, China*
[b]*School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK*
[c]*School of Computer Science and Engineering, Southeast University, Nanjing 210096, China*

Abstract

In this paper, we propose a novel label distribution manifold learning (LDML) method for solving the multilabel distribution learning problem. First, using manifold learning, we extract the accurate and reduced-dimension features of the training data. Second, we estimate the unknown label distributions associated with the extracted reduced-dimension features based on multi-output kernel regression. Third, we use the extracted reduced-dimension features and their associated estimated label distributions to form an enhanced maximum entropy model, which enables us to accurately and efficiently estimate the unknown true label distributions for the training data. We refer to this algorithm as the LDML. We also propose to apply the tangent space alignment regression in the second stage, and the resulting algorithm is called the LDML-R. The LDML-R has better label distribution learning performance than the LDML but imposes higher complexity than the latter. We evaluate the proposed LDML and LDML-R algorithms on 15 real-world data sets with ground-truth label distributions, and the experimental results obtained show that our method has advantages in terms of learning accuracy compared to the latest multi-label distribution learning approaches. We also use another 10 real-world multi-class data sets, which do not have the ground-truth label distributions, to demonstrate the

Email addresses: `tutu_tanchao@163.com` (Chao Tan), `sqc@ecs.soton.ac.uk` (Sheng Chen), `xgeng@seu.edu.cn` (Xin Geng), `glji@njnu.edu.cn` (Genlin Ji)

superior multilabel classification performance of our LDML-R algorithm over the existing state-of-the-art multi-label classification algorithms.

Keywords: Multi-label learning, label distribution learning, manifold learning, dimension reduction, linear regression

1. Introduction

Multi-label learning (MLL) [1] handles the case where one instance corresponds to multiple labels, with goal of learning a mapping from examples to related label sets. MLL is widely used for classification, recognition and retrieval in many areas, such as multi-label text classification [2], aircraft heading changes to resolve conflicts [3], chest radiography classification [4], multi-label image classification [5], and multi-label clinical document classification [6], etc. The data in these applications are often rich in semantics, and hence suitable for modeling using MLL. Traditional methods of MLL generally adopt the uniform label distribution assumption, i.e., the importance of each related label to the example is considered equal. However, for many real-world applications, the multi labels for a sample do not have the same importance to the sample. Rather some labels have primary importance to the sample, while the others have secondary importance.

Label distribution learning (LDL) [7] is a related machine learning paradigm in which each instance is annotated by a label distribution covering the importance of its labels. The emergence of LDL makes it possible to learn richer semantics from data other than multiple labels. Applications of LDL include video parsing [8], scene classification [9], and indoor crowd counting [10]. LDL can characterize the relative importance of the multiple labels related to the same example more accurately [11]. However, in real-world multi-label applications, the training data are usually labeled by multiple logical labels (uniform label distribution), and the true label distribution information is unavailable. Nevertheless, the supervised information in these data essentially follows some kind of label distribution, which is often implicitly contained in the training samples. If this label distribution can be recovered by a suitable method, the advantages of mining more semantic information by MLL can be realized.

The process of promoting the original logical label to the label distribution is called the label enhancement (LE) in LDL, and the concept of LE was proposed by Geng *et al.* [11]. An application of LE to large-scale retrieval was given in [12]. However, it is difficult to obtain the label distributions directly from the logical labels in the training set. To solve this problem, an effective approach is to recover the label distributions from the logical labels in the training set by utilizing the correlation between the topological information in the feature space and the labels. More specifically, by mining the relevant information of the labels hidden in the training samples to establish the relationship between the examples' correlation and the labels' correlation, the logical labels of the examples are enhanced to the label distributions. After the label distributions are recovered, more effective supervised learning can be achieved by using the label distributions, instead of the logical labels.

The label distributions cannot be obtained explicitly from the training examples. In order to reconstruct the label manifold that is necessary for LDL, the key is the topology. According to the smoothness assumption [13], local topology can be shared between feature manifold and the labels. Moreover, examples or points close to each other are more likely to share the same labels. Based on this smoothness property, in this paper, we propose a label distribution manifold learning (LDML) approach with the two algorithms for multi-label distribution learning by reconstructing and utilizing the label manifold. With the expansion from the logical label space onto the Euclidean label space, we can naturally utilize the smoothness property to transfer the local topology from the manifold space onto the label space. The feature vectors in the manifold space and the label vectors in the label space will guide LDL. To our best knowledge, this is an earliest attempt to explore manifold in the label space for multi-label distribution learning. More specifically, the proposed LDML method consists of the following three components.

1. Manifold space enhanced feature extraction: With the nonlinear dimensionality reduction using the locally linear embedding manifold learning (LLEML) algorithm [14], we extract accurate and reduced-dimension features in the feature manifold space construction.

2. Regression to estimate the label distributions of the extracted features: The unknown label distributions associated with the extracted reduced-dimension features are estimated based on multi-output kernel regression.

   Alternatively, the local tangent space alignment (LTSA) [15] based regression can be adopted to learn the unknown label distributions associated with the extracted reduced-dimension features.

3. Enhanced maximum entropy model based LDL: By substituting the full-dimensional data and their corresponding logical labels in the standard maximum entropy model with the reduced-dimension features extracted in step 1) and their associated enriched label distribution estimates acquired in step 2), we form the enhanced maximum entropy model. A gradient-descent iterative optimization is then performed to estimate the unknown true label distributions.

The resulting algorithm by adopting regression in step 2 is referred to as the LDML, while the resulting algorithm by applying the LTSA regression in step 2 is called the LDML-R. Extensive experiments show that our LDML approach significantly improve the performance of multi-label distribution learning. Experimental results also demonstrate that our method has better multi-label classification performance compared with the latest multi-label learning algorithms.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the related work. The proposed LDML approach is detailed in Section 3. Extensive experimental results are reported in Section 4, and our conclusions are offered in Section 5.

2. Related Work

LDL has been successfully applied to various problems, such as facial age estimation [16, 17], head pose estimation [18] and multi-label ranking of natural scene images [19]. Formally, the goal of LDL is to learn the conditional probability of the label vector conditioned on the input sample. According to [16], a well-known approach for learning multi-label distributions is known as the algorithm adaptation (AA) strategy, which adapts existing learning algorithms to process label assignments directly. Two typical algorithms based on this strategy are the AA with backpropagation (AA-BP) and with k-nearest neighbor (AA-kNN) [16]. For the AA-kNN, the average value of the label distributions of k nearest neighbors is calculated as the predicted label distribution, while for the AA-BP, the backpropagation (BP) algorithm is used to training a single layer neural network with multiple outputs as the predicted label distribution.

An alternative strategy is to adopt the maximum entropy model which turns the problem of estimating the unknown label distributions into the problem of estimating the label distributions' parameter vectors. Specifically, by substituting the logical labels for the unknown label distributions and using the full-dimensional input samples as the features in the maximum entropy model, the label distributions' parameter vectors can be estimated via iterative optimization procedures. The two representative algorithms of this strategy are the IIS-LLD and BFGS-LLD [16], The difference between these two algorithms is that the IIS-LLD is a gradient descent iterative method while the BFGS-LLD adopts a quasi-Newton iterative method.

In addition, Geng and Hou [20] regard LDL as a regression problem and proposed the label distribution support vector regression (LDSVR), which applies support vector regression (SVR) to process label assignment. Another well-known LDL algorithm is called the conditional probabilistic neural network (CPNN) [7]. Hou *et al.* [13] proposed an LE algorithm based on manifold learning, and this method relies on the assumption that each data point can be optimally reconstructed by using a linear combination of its neighbors [21].

In this paper, we use manifold in the label space to improve the performance of multi-label distribution learning. To our best knowledge, this is the first attempt to explore label manifolds in multi-label distribution learning.

3. The Proposed Algorithms

3.1. Problem description and maximum entropy model

For the generic multi-label problem, let $\boldsymbol{x} \in \mathbb{R}^q$ be an instance, and $\boldsymbol{y} = \left[y^1\ y^2 \cdots y^c\right]^{\mathrm{T}} \in \{0,1\}^c$ be its logical class label vector, where q is the feature dimension and c is the number of classes. The degree to which the label y^j, $1 \le j \le c$, describes the example $\boldsymbol{x}$ is defined by the conditional probability $d_{\boldsymbol{x}}^{y^j} = \Pr\left(y^j|\boldsymbol{x}\right)$, where $d_{\boldsymbol{x}}^{y^j} \in [0,\ 1]$, $1 \le j \le c$, and $\sum_{j=1}^{c} d_{\boldsymbol{x}}^{y^j} = 1$. It can be seen that for each example, the descriptiveness of all the labels in the label set builds a data form similar to a probability distribution, hence the name label distribution. This label distribution however is unknown. The process of learning the label distribution of a labeled example is known as LDL.

Given the labeled training data set $\left\{\boldsymbol{x}_i, \boldsymbol{y}_i\right\}_{i=1}^{n}$, where n is the sample size, $\boldsymbol{x}_i = \left[x_i^1\ x_i^2 \cdots x_i^q\right]^{\mathrm{T}} \in \mathbb{R}^q$ and $\boldsymbol{y}_i = \left[y_i^1\ y_i^2 \cdots y_i^c\right]^{\mathrm{T}} \in \{0,1\}^c$ are the i sample and its associated logical label vector, respectively, the task of LDL is to learn the unknown underlying label distributions $\left\{d_{\boldsymbol{x}_i}^{y_i^1}, d_{\boldsymbol{x}_i}^{y_i^2}, \cdots, d_{\boldsymbol{x}_i}^{y_i^c}\right\}_{i=1}^{n}$, where

$$d_{\boldsymbol{x}_i}^{y_i^j} \in [0,\ 1], 1 \le j \le c, \text{and } \sum\nolimits_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} = 1, 1 \le i \le n. \tag{1}$$

An effective LDL approach is to express the estimate of $d_{\boldsymbol{x}_i}^{y_i^j}$ in the form of the parameterized conditional probability model

$$\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} = \Pr\left(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j}\right),\ 1 \le j \le c, 1 \le i \le n, \tag{2}$$

where $\boldsymbol{w}_{i,j} = \left[w_{i,j}^1\ w_{i,j}^2 \cdots w_{i,j}^q\right]^{\mathrm{T}} \in \mathbb{R}^q$ is a parameter vector. Thus, learning the label distributions is turned into the problem of estimating $\boldsymbol{w}_{i,j}$ for every $\{\boldsymbol{x}_i, y_i^j\}$, $1 \le i \le n$ and $1 \le j \le c$.

A well-known parameterized conditional probability model is the maximum entropy model [7, 16], in which $\Pr\left(y_i^j|\boldsymbol{x}_i;\boldsymbol{w}_{i,j}\right)$ takes the exponential form

$$\Pr\left(y_i^j|\boldsymbol{x}_i;\boldsymbol{w}_{i,j}\right) = \frac{1}{Z_i}\exp\left(\sum_{k=1}^{q} w_{i,j}^k f_k\big(\boldsymbol{x}_i, y_i^j\big)\right), \tag{3}$$

with the normalization factor

$$Z_i = \sum_{j=1}^{c}\exp\left(\sum_{k=1}^{q} w_{i,j}^k f_k\big(\boldsymbol{x}_i, y_i^j\big)\right), \tag{4}$$

where $f_k(\boldsymbol{x}_i, y_i^j) \in \mathbb{R}$ is known as the kth feature function that relies on both instance $\boldsymbol{x}_i$ and label y_i^j, for $1 \le k \le q$. The features are further expressed as $f_k\big(\boldsymbol{x}_i, y_i^j\big) = y_i^j g_k(\boldsymbol{x}_i)$, with $g_k(\boldsymbol{x}_i)$ denoting the class-independent kth feature function. This simplification allow (3) to be rewritten as

$$\Pr\left(y_i^j|\boldsymbol{x}_i;\boldsymbol{w}_{i,j}\right) = \frac{1}{Z_i}\exp\left(\sum_{k=1}^{q} \big(w_{i,j}^k \cdot y_i^j\big) g_k\big(\boldsymbol{x}_i\big)\right). \tag{5}$$

Recognizing $\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} = 1$ yields the target function for all the parameter vectors $\boldsymbol{w} = \big\{\boldsymbol{w}_{i,j}, 1 \le j \le c, 1 \le i \le n\big\}$:

$$\begin{aligned} T(\boldsymbol{w}) = & \sum_{i=1}^{n}\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} \ln P\big(y_i^j|\boldsymbol{x}_i;\boldsymbol{w}_{i,j}\big) = \sum_{i=1}^{n}\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} \sum_{k=1}^{q}\big(w_{i,j}^k \cdot y_i^j\big) g_k\big(\boldsymbol{x}_i\big) \\ & - \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{c}\exp\left(\sum_{k=1}^{q}\big(w_{i,j}^k \cdot y_i^j\big) g_k\big(\boldsymbol{x}_i\big)\right)\right). \end{aligned} \tag{6}$$

If all the true label distributions $d_{\boldsymbol{x}_i}^{y_i^j}$ and the feature functions $g_k\big(\boldsymbol{x}_i\big)$ were available, the target function (6) could be optimized using the improved iterative scaling (IIS) [22]. Specifically, the IIS finds the optimal parameters $\boldsymbol{w}$ by solving the nonlinear equation associated with the lower bound of $T(\boldsymbol{w} + \Delta\boldsymbol{w}) - T(\boldsymbol{w})$ based on an iterative procedure, such as the Gauss-Newton method. This is of course impractical, as $d_{\boldsymbol{x}_i}^{y_i^j}$ are unknown and they are to be estimated.

Since $g_k\big(\boldsymbol{x}_i\big)$ and in particular $d_{\boldsymbol{x}_i}^{y_i^j}$ are unknown, a practical solution is to construct an empirical target function by substituting the unknown true label distributions $d_{\boldsymbol{x}_i}^{y_i^j}$ with the known logical labels y_i^j as well as by substituting $g_k\big(\boldsymbol{x}_i\big)$ with x_i^k. That is, the following empirical target function is adopted

$$T_{\mathrm{e}}(\boldsymbol{w})=\sum_{i=1}^{n}\sum_{j=1}^{c} y_i^j \sum_{k=1}^{q}\big(w_{i,j}^k\cdot y_i^j\big)x_i^k-\sum_{i=1}^{n}\ln\left(\sum_{j=1}^{c}\exp\left(\sum_{k=1}^{q}\big(w_{i,j}^k\cdot y_i^j\big)x_i^k\right)\right). \quad (7)$$

The IIS-LLD and BFGS-LLD [7, 16] are in fact the iterative optimization algorithms that find the label distributions' parameters $\boldsymbol{w}$ by solving the nonlinear equation associated with the lower bound of $T_{\mathrm{e}}(\boldsymbol{w}+\Delta\boldsymbol{w})-T_{\mathrm{e}}(\boldsymbol{w})$ using gradient descent method and Gauss-Newton method, respectively.

This maximum entropy model based approach has some drawbacks. First it does not calculate the features $g_k(\boldsymbol{x}_i)$, and using the kth element of $\boldsymbol{x}_i$ as its kth feature is clearly heuristic. Furthermore, the logical label y_i^j contains far less information than the associated label distribution. These two substitutions or approximations inherently limit the accuracy of the empirical model (7). Additionally, since the dimension q for many practical applications is very large, the IIS-LLD and BFGS-LLD impose high computational cost. Searching the solution to these problems motivate our work.

3.2. Manifold space construction for feature extraction

We extract the class-independent features in the maximum entropy model (6) based on the LLEML algorithm [14]. According to the smoothness property [13], the topology of the feature space can be transferred locally to a local label space. To maintain locality, we rely on the property that each data point can be reconstructed optimally using a linear combination of its neighbors. Specifically, a graph $\mathcal{G}=(\mathcal{V},\mathcal{E},\boldsymbol{\Omega})$ is used to represent the topology of the multilabel training data set, where $\mathcal{V}$ denotes the set of vertices composed of examples, $\mathcal{E}$ denotes the set of edges, and $\boldsymbol{\Omega}=\big[\omega_{i,j}\big]\in\mathbb{R}^{n\times n}$ is the graph weight matrix with $\omega_{i,j}$ representing the coefficient of the ith point reconstructed by the jth point. First, in the feature space, we need to use the local neighborhood information of each point to construct $\mathcal{G}$, that is, any example $\boldsymbol{x}_i$ can be reconstructed by the linear combination of its k nearest neighbors $\big\{\boldsymbol{x}_{i_1},\cdots,\boldsymbol{x}_{i_k}\big\}$. The reconstruction weight matrix $\boldsymbol{\Omega}$ can be obtained by solving the optimization [14]

$$\begin{aligned} \min_{\boldsymbol{\Omega}} \quad & \Xi(\boldsymbol{\Omega}) = \textstyle\sum\limits_{i=1}^{n} \left\| \boldsymbol{x}_i - \textstyle\sum\limits_{j=1}^{n} \omega_{i,j} \boldsymbol{x}_j \right\|^2, \\ \text{s.t.} \quad & w_{i,j} = 0 \text{ if } \boldsymbol{x}_j \notin \{\boldsymbol{x}_{i_1},\cdots,\boldsymbol{x}_{i_k}\} \text{ and } \textstyle\sum_{j=1}^{n} \omega_{i,j} = 1, \quad 1 \le i \le n. \end{aligned} \tag{8}$$

The optimization problem (8) has a closed-form solution [14].

In order to find the low-dimensional embedded coordinates $\boldsymbol{G} = \big[\boldsymbol{g}_1 \cdots \boldsymbol{g}_n\big]$ that can optimally maintain the weight matrix $\boldsymbol{\Omega}$, we define the cost:

$$\Phi(\boldsymbol{G}) = \sum_{i=1}^{n} \left\| \boldsymbol{g}_i - \sum_{j=1}^{n} \omega_{i,j} \boldsymbol{g}_j \right\|^2, \tag{9}$$

where $\boldsymbol{g}_i \in \mathbb{R}^d$ with $d < q$. Since $\big\{\boldsymbol{g}_i\big\}$ has the same local topology as $\big\{\boldsymbol{x}_i\big\}$, any $\boldsymbol{g}_i$ can be reconstructed by the linear combination of its k nearest neighbors $\big\{\boldsymbol{g}_{i_1},\cdots,\boldsymbol{g}_{i_k}\big\}$. Given $\boldsymbol{\Omega}$, we extract $\boldsymbol{G}$ by minimizing (9). To make the optimization well posed and without changing the cost value, we constrain the d-dimensional embedded coordinates to be centered on the origin $\sum_{i=1}^{n} \boldsymbol{g}_i = \boldsymbol{0}_d$ and to have unit covariance matrix $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{g}_i \boldsymbol{g}_i^{\mathrm{T}} = \boldsymbol{I}_d$, where $\boldsymbol{0}_d$ is the d-dimensional vector whose elements are all zero and $\boldsymbol{I}_d$ is the $d \times d$ identity matrix. Then the cost can be expressed in the following quadratic form [14]

$$\Phi(\boldsymbol{G}) = \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=1}^{n} m_{i,j} \boldsymbol{g}_i^{\mathrm{T}} \boldsymbol{g}_j, \tag{10}$$

with the symmetric, positive semi-definite and sparse matrix $\boldsymbol{M} = \big[m_{i,j}\big] \in \mathbb{R}^{n \times n}$ given by

$$\boldsymbol{M} = \big(\boldsymbol{I}_n - \boldsymbol{\Omega}\big)^{\mathrm{T}} \big(\boldsymbol{I}_n - \boldsymbol{\Omega}\big). \tag{11}$$

The solution of $\boldsymbol{G}$ is obtained by the decomposition of $\boldsymbol{M}$ [23]. The eigenvectors corresponding to the second to $(d+1)$th smallest eigenvalues of $\boldsymbol{M}$ constitute the d-dimensional embedded coordinates $\boldsymbol{G}$, which minimizes the cost function (9). Specifically, grouping these d eigenvectors as the $n \times d$ matrix $\boldsymbol{E_M}$, then $\boldsymbol{G} = \boldsymbol{E}_{\boldsymbol{M}}^{\mathrm{T}}$. This dimension reduction algorithm maintains the local neighborhood structure and restores the entire data set by stitching each neighborhood structure. The maintenance of the local structure information of high-dimensional data set is the notable feature of the LLEML algorithm [14]. The two algorithmic parameters or hyperparameters are k and d.

According to the smoothing property that the labels of examples with similar features are also likely to be similar, the topology of the feature space can be transferred into the label space, i.e., sharing the same local linear reconstruction of the d-dimensional embedded coordinates $\boldsymbol{G}$. Hence, we can use the d-dimensional $\boldsymbol{g}_i = \left[g_i^1 \cdots g_i^d\right]^{\mathrm{T}}$ to replace the q-dimensional unknown features $\boldsymbol{g}(\boldsymbol{x}_i) = \left[g_1(\boldsymbol{x}_i) \cdots g_q(\boldsymbol{x}_i)\right]^{\mathrm{T}}$ in the maximum entropy model (6).

3.3. Regression to estimate features' label distributions

Since $\boldsymbol{g}_i$ is the low-dimensional feature vector of $\boldsymbol{x}_i$, it will share the same logical label vector $\boldsymbol{y}_i$ with $\boldsymbol{x}_i$. It can also be envisaged that there exists an underlying label distribution vector $\boldsymbol{d}_{\boldsymbol{g}_i} = \left[d_{\boldsymbol{g}_i}^{y_i^1} \cdots d_{\boldsymbol{g}_i}^{y_i^c}\right]^{\mathrm{T}}$ associated with the feature vector $\boldsymbol{g}_i$. Clearly, $\{\boldsymbol{d}_{\boldsymbol{g}_i}\}$ contain richer supervised information than the logical label vectors $\{\boldsymbol{y}_i\}$. Therefore, if we can obtain the unknown label distributions $\{\boldsymbol{d}_{\boldsymbol{g}_i}\}$, we can use them, instead of $\{\boldsymbol{y}_i\}$, in the maximum entropy model (6).

To estimate $\{\boldsymbol{d}_{\boldsymbol{g}_i}\}$ is to construct a Euclidean label or label distribution manifold, which can readily be reconstructed using the local topology from the feature manifold established in Subsection 3.2 and the known logical labels. Hence, we can represent $\{\boldsymbol{d}_{\boldsymbol{g}_i}\}$ by the following linear-in-the-parameter kernel regression model

$$d_{\boldsymbol{g}_i}^{y_i^j} = \boldsymbol{\theta}_{i,j}^{\mathrm{T}} \boldsymbol{\psi}(\boldsymbol{x}_i) + e_{i,j}, \ 1 \leq i \leq n, 1 \leq j \leq c, \tag{12}$$

where $\boldsymbol{\psi}(\boldsymbol{x}_i) \in \mathbb{R}^d$ are the kernel feature vectors associated with the features $\boldsymbol{g}_i$, and $\boldsymbol{\theta}_{i,j} \in \mathbb{R}^d$ are the regression weight vectors. That is, we estimate $d_{\boldsymbol{g}_i}^{y_i^j}$ with

$$\widehat{d}_{\boldsymbol{g}_i}^{y_i^j} = \boldsymbol{\theta}_{i,j}^{\mathrm{T}} \boldsymbol{\psi}(\boldsymbol{x}_i), \tag{13}$$

Define the $n \times n$ kernel matrix as [24, 25]

$$\boldsymbol{K} = (\lambda_{\max} \boldsymbol{I}_n - \boldsymbol{M}), \tag{14}$$

where the matrix $\boldsymbol{M}$ is given in (11) and $\lambda_{\max}$ is the maximum eigenvalue of $\boldsymbol{M}$. The kernel feature matrix $\boldsymbol{\Psi} = \left[\boldsymbol{\psi}(\boldsymbol{x}_1) \cdots \boldsymbol{\psi}(\boldsymbol{x}_n)\right]$ are the eigenvectors corresponding to the largest d eigenvalues of $\boldsymbol{K}$ [23].

To estimate $\boldsymbol{\Theta} = \left\{\boldsymbol{\theta}_{i,j}, 1 \le i \le n, 1 \le j \le c\right\}$, consider the standard SVR technique with the loss function

$$L(\boldsymbol{\Theta}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{c} \|\boldsymbol{\theta}_{i,j}\|^2 + \sum_{i=1}^{n} L_2(r_i), \tag{15}$$

where $r_i = \|\boldsymbol{e}_i\|$ and $\boldsymbol{e}_i = \left[e_{i,1} \cdots e_{i,c}\right]^{\mathrm{T}}$ with

$$e_{i,j} = y_i^j - \boldsymbol{\theta}_{i,j}^{\mathrm{T}} \boldsymbol{\psi}(\boldsymbol{x}_i), \ 1 \le j \le c, \tag{16}$$

while the L_2 loss is given by

$$L_2(r) = \begin{cases} 0, & r < \varepsilon, \\ (r - \varepsilon)^2, & r \ge \varepsilon. \end{cases} \tag{17}$$

175 After obtaining $\boldsymbol{\Theta}$ with the SVR, we arrive at the estimates of the label distributions $\widehat{d}_{\boldsymbol{g}_i}^{y_i^j}$, $1 \le j \le c$, $1 \le i \le n$, associated with the feature vectors $\boldsymbol{g}_i$.

3.4. LTSA regression to estimate features' label distributions

As aforementioned, it can be visualized that there exists a set of the label distributions $\{d_i^j\}_{j=1}^c$, which contains more supervisory information than the logical label set $\{y_i^j\}_{j=1}^c$ for $\boldsymbol{g}_i$. According to the smooth assumption [26], samples close to each other in the feature space are likely to have the same labels. Let us determine $\boldsymbol{X}_i$ as the k nearest neighbors for each $\boldsymbol{x}_i$. By transferring the closeness in the feature manifold space to the closeness in the label manifold space, we can reconstruct the label manifold to align the reduced-dimensional feature $\boldsymbol{g}_i$ to the label space with c labels, that is, to align the dimension k to c. This can be formulated as the following optimization

$$\min \left\| \boldsymbol{T}_i \Big(\boldsymbol{I}_k - \frac{1}{k} \mathbf{1}_k \mathbf{1}_k^{\mathrm{T}} \Big) - \boldsymbol{L}_i \boldsymbol{U}_i \right\| . \tag{18}$$

Here $\boldsymbol{U}_i = \boldsymbol{Q}_i^{\mathrm{T}} \boldsymbol{X}_i \left(\boldsymbol{I}_k - \frac{1}{k} \mathbf{1}_k \mathbf{1}_k^{\mathrm{T}} \right)$, $\mathbf{1}_k$ is the k-dimensional vector whose elements are all 1, $\boldsymbol{T}_i$ denotes the low-dimensional global coordinates with respect to the local geometry determined by the $\boldsymbol{U}_i$, and $\boldsymbol{L}_i = \boldsymbol{T}_i \left(\boldsymbol{I}_k - \frac{1}{k} \mathbf{1}_k \mathbf{1}_k^{\mathrm{T}} \right) \boldsymbol{U}_i^{\dagger}$, while $\boldsymbol{Q}_i$ denotes the matrix formed by the eigenvectors corresponding to the first c maximum eigenvalues of the neighborhood covariance matrix of point $\boldsymbol{x}_i$. Using the LTSA algorithm [15], we obtain the global alignment matrix $\boldsymbol{W}_i \boldsymbol{W}_i^{\mathrm{T}}$, with

$$\boldsymbol{W}_i = \left(\boldsymbol{I}_c - \frac{1}{c} \mathbf{1}_c \mathbf{1}_c^{\mathrm{T}} \right) \left(\boldsymbol{I}_c - \boldsymbol{U}_i^{\dagger} \boldsymbol{U}_i \right) . \tag{19}$$

The global low-dimensional coordinates $\bar{\boldsymbol{g}}_i \in \mathbb{R}^c$ are composed of the eigenvectors corresponding to the first c small eigenvalues of the global matrix.

We can model $\left\{ d_i^j \right\}_{j=1}^{c}$ by the linear regression model

$$d_i^j = \bar{\boldsymbol{g}}_i^{\mathrm{T}} \bar{\boldsymbol{\theta}}_{i,j} + \bar{e}_{i,j}, \ 1 \le j \le c, 1 \le i \le n, \tag{20}$$

namely, we estimate d_i^j by

$$\widehat{d_i^j} = \bar{\boldsymbol{g}}_i^{\mathrm{T}} \bar{\boldsymbol{\theta}}_{i,j}, \tag{21}$$

where $\bar{\boldsymbol{\theta}}_{i,j} \in \mathbb{R}^c$ is the parameter vector of the label distribution estimate $\widehat{d_i^j}$. After estimating all the $\widehat{d_i^j}$, i.e., all the $\bar{\boldsymbol{\theta}}_{i,j}$, for $1 \le j \le c$ and $1 \le i \le n$, we need to perform the normalization

$$\widetilde{d}_i^j = \frac{\widehat{d}_i^j}{\sum_{l=1}^{c} \widehat{d}_i^l},\ 1 \le i \le n. \tag{22}$$

Then $\widetilde{d}_i^j$ is the estimate of d_i^j.

To estimate the parameters $\bar{\boldsymbol{\Theta}} = \big\{\bar{\boldsymbol{\theta}}_{i,j}, 1 \le j \le c, 1 \le i \le n\big\}$, the regression cost function similar to (15) can be adopted

$$L\big(\bar{\boldsymbol{\Theta}}\big) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{c}\big\|\bar{\boldsymbol{\theta}}_{i,j}\big\|^2 + \sum_{i=1}^{n} L_2\big(\bar{r}_i\big), \tag{23}$$

in which $\bar{r}_i = \big\|\bar{\boldsymbol{e}}_i\big\|$ and $\bar{\boldsymbol{e}}_i = \big[\bar{e}_{i,1} \cdots \bar{e}_{i,c}\big]^{\mathrm{T}}$ with

$$\bar{e}_{i,j} = y_i^j - \bar{\boldsymbol{g}}_i^{\mathrm{T}}\bar{\boldsymbol{\theta}}_{i,j},\ 1 \le j \le c, \tag{24}$$

while the L_2 loss is specified in (17). With the constraints $\bar{\boldsymbol{g}}_i^{\mathrm{T}}\bar{\boldsymbol{\theta}}_{i,j} \ge 0$, the iterative reweighed least squares (IRWLS) [27] can readily be used to solve this multi-output regression problem.

Clearly, this LTSA regression for estimating the features' label distributions imposes significantly higher complexity than the regression of Subsection 3.3, but it potentially offers more accurate estimates, which will be investigated in the experimental evaluation section.

3.5. Enhanced maximum entropy model and LDML/LDML-R algorithms

1) LDML: By using the extracted reduced-dimensional features $\boldsymbol{g}_i = \big[g_i^1 \cdots g_i^d\big]^{\mathrm{T}}$ and the associated label distribution estimates $\widehat{\boldsymbol{d}}_{\boldsymbol{g}_i} = \big[\widehat{d}_{\boldsymbol{g}_i}^{y_i^1} \cdots \widehat{d}_{\boldsymbol{g}_i}^{y_i^c}\big]^{\mathrm{T}}$ in the maximum entropy model (6), we arrive at the enhanced empirical target function

$$\widehat{T}_{\mathrm{e}}(\boldsymbol{w}) = \sum_{i=1}^{n}\sum_{j=1}^{c}\widehat{d}_{\boldsymbol{g}_i}^{y_i^j}\sum_{k=1}^{d}\big(w_{i,j}^k \cdot \widehat{d}_{\boldsymbol{g}_i}^{y_i^j}\big)g_i^k - \sum_{i=1}^{n}\ln\left(\sum_{j=1}^{c}\exp\left(\sum_{k=1}^{d}\big(w_{i,j}^k \cdot \widehat{d}_{\boldsymbol{g}_i}^{y_i^j}\big)g_i^k\right)\right). \tag{25}$$

```
Algorithm 1 Label Distribution Manifold Learning (LDML)
Input: Multilabel sample set of size $n$: $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$, where samples $\boldsymbol{x}_i \in \mathbb{R}^q$ and
    the label vectors $\boldsymbol{y}_i = \left[y_i^1 \cdots y_i^c\right]^{\mathrm{T}} \in \{0,1\}^c$.
Output: Label distribution estimates $\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} = \Pr\left(y_i^j | \boldsymbol{x}_i; \boldsymbol{w}_{i,j}\right)$ of unknown label
    distributions $d_{\boldsymbol{x}_i}^{y_i^j}$, $1 \le j \le c$ and $1 \le i \le n$.
Step 1. Extract features: Use manifold learning method of Subsection 3.2
    to extract reduced dimension features $\boldsymbol{G} = \left[\boldsymbol{g}_1 \cdots \boldsymbol{g}_n\right] \in \mathbb{R}^{n \times d}$ as eigenvectors
    corresponding to second to $(d+1)$ smallest eigenvalues of $\boldsymbol{M}$ in (11).
Step 2. Estimate label distributions for features: Use kernel regression of
    Subsection 3.3 to estimate label distributions $\widehat{\boldsymbol{d}}_{\boldsymbol{g}_i}$ of extracted features $\boldsymbol{g}_i$.
Step 3. Label enhancement based on manifold: Use $\{\boldsymbol{g}_i, \widehat{\boldsymbol{d}}_{\boldsymbol{g}_i}\}$ to form
    enhanced maximum entropy model (25), and use gradient-descent iterative
    optimization to find parameters $\boldsymbol{w}_{i,j}$, $1 \le i \le n$, $1 \le j \le c$.
return $\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} \leftarrow \frac{1}{Z_i} \exp\left(\sum_{k=1}^{d} \left(w_{i,j}^k \cdot \widehat{d}_{\boldsymbol{g}_i}^{y_i^j}\right) g_i^k\right)$, $1 \le i \le n$, $1 \le j \le c$.
```

Note that the parameter vector of the conditional probability model is now d-dimensional rather than q-dimensional, that is, $\boldsymbol{w}_{i,j} \in \mathbb{R}^d$.

A gradient descent iterative optimization can be applied to find the parameters $\boldsymbol{w}_{i,j}$, $1 \le i \le n$ and $1 \le j \le c$, of the conditional probability models (2) by solving the nonlinear equation associated with the lower bound of $\widehat{T}_{\mathrm{e}}(\boldsymbol{w} + \Delta\boldsymbol{w}) - \widehat{T}_{\mathrm{e}}(\boldsymbol{w})$ to yield the estimates $\widehat{d}_{\boldsymbol{x}_i}^{y_i^j}$ of (2) for all the unknown label distributions $d_{\boldsymbol{x}_i}^{y_i^j}$. Our proposed LDML algorithm is summarized in Algorithm 1.

2) LDML-R: Similarly, by using the extracted reduced-dimensional features $\boldsymbol{g}_i = \left[g_i^1 \cdots g_i^d\right]^{\mathrm{T}}$ and the associated label distribution estimates $\widetilde{\boldsymbol{d}}_i = \left[\widetilde{d}_i^1 \cdots \widetilde{d}_i^c\right]^{\mathrm{T}}$ in the maximum entropy model (6), we arrive at the alternative enhanced empirical target function

$$\widetilde{T}_{\mathrm{e}}(\boldsymbol{w}) = \sum_{i=1}^{n} \sum_{j=1}^{c} \widetilde{d}_i^j \sum_{k=1}^{d} \left(w_{i,j}^k \cdot \widetilde{d}_i^j\right) g_i^k - \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{d} \left(w_{i,j}^k \cdot \widetilde{d}_i^j\right) g_i^k\right)\right). \tag{26}$$

It is clear that the LDML-R algorithm differs from the LDML of Algorithm 1 in **Step 2**. Since the LTSA regression of Subsection 3.4 imposes higher computational complexity than the regression of Subsection 3.3, the LDML-R algorithm

Table 1: Multilabel datasets with known ground-truth label distributions [28] used in experimental evaluation with LDL metrics

Dataset	Examples (n)	Features (q)	Labels (c)
Yeast-alpha	2465	24	18
Yeast-cdc	2465	24	15
Yeast-cold	2465	24	4
Yeast-diau	2465	24	7
Yeast-dtt	2465	24	4
Yeast-elu	2465	24	14
Yeast-heat	2465	24	6
Yeast-spo	2465	24	6
Yeast-spo5	2465	24	3
Yeast-spoem	2465	24	2
Human Gene	30542	36	68
Natural Scene	2000	294	9
Movie	7755	1869	5
SJAFFE	213	243	6
SBU_3DFE	2500	243	6

imposes higher complexity than the LDML algorithm. However, because $\widetilde{d}_i^j$ estimated in Subsection 3.4 may contain more label information than $\widehat{d}_{\boldsymbol{g}_i}^{y_i^j}$ estimated in Subsection 3.3, the LDML-R may outperform the LDML, in terms of LDL accuracy. This will be further demonstrated in the experimental study.

4. Experimental Evaluation

4.1. Experiment setup

4.1.1. Datasets

1) Our primary objective is to evaluate the label distribution estimation accuracy of the proposed LDML and LDML-R algorithms, namely, how close their label distribution estimates to the ground-truth label distributions. We select 15 real-world multilabel datasets from Mulan website [28], whose ground-truth label distributions are provided. Table 1 summarizes the features of these datasets. Half of these datasets are regular-sized and half of them are large-scale. These datasets therefore cover a wide range of multilabel attributes.

2) It is also crucial to evaluate the multilabel classification capability of the proposed LDML and LDML-R algorithms using various MLL metrics. For

Table 2: Characteristics of 10 real-world datasets from [28] with unknown ground-truth label distributions used in experimental evaluation with MLL metrics

Dataset	S	T	$dim(S)$	$L(S)$	$LCard(S)$	$LDen(S)$	$DL(S)$	$F(S)$
Emotions	415	178	72	6	1.869	0.311	27	numeric
Medical	645	333	1449	45	1.245	0.028	94	nominal
Cal500	250	252	68	174	26.044	0.150	502	numeric
Birds	320	325	260	19	1.014	0.053	133	numeric
Enron	1123	579	1001	53	3.378	0.064	753	nominal
Yeast	1200	1217	103	14	4.237	0.303	198	numeric
Image	1000	1000	294	5	1.236	0.247	20	numeric
Scene	1211	1196	294	6	1.074	0.179	15	numeric
Corel5k	2500	2500	499	374	3.522	0.009	3175	nominal
Bibtex	3700	3695	1836	159	2.402	0.015	2856	nominal

this purpose, we select another 10 real-world multilabel datasets from Mulan website [28], which do not have ground-truth label distributions, for performance evaluation. Table 2 summarizes the features of these 10 real-world datasets from [28], with unknown ground-truth label distributions. These datasets cover a wide range of multilabel attributes. In Table 2, S: the number of examples, T: the number of testing samples, $dim(S)$: the feature dimensions, $L(S)$: the number of class labels, $LCard(S)$: the label cardinality, $LDen(S)$: the label density, $DL(S)$: the distinct label sets, and $F(S)$: the feature type.

4.1.2. Comparison algorithms

1) In the experimental evaluation of LDL accuracy, we choose six well-established multilabel distribution learning algorithms, the AA-BP [16], the BFGS-LLD [16], CPNN [7], AA-KNN [16], IIS-LLD [16] and LDSVR [20], as the benchmarks for comparison with our LDML and LDML-R algorithms.

2) In the experimental evaluation of multilabel classification performance, we first compare our LDML and LDML-R algorithms with the 5 existing state-of-the-art LDL algorithms, the AA-BP, CPNN, AA-KNN, IIS-LLD and LDSVR.

3) Next, we select five up-to-date MLL algorithms, namely, backpropagation for multilabel learning (BP-MLL) [29], multi-label manifold learning (ML^2) [13], multi-label lazy learning approach (ML-kNN) [30], multi-label naive Bayes classifier (MLNB) [31], and multi-label learning with feature-induced labeling information enrichment (MLFE) [32], as the benchmarks for comparison with our LDML and LDML-R in the multilabel classification experiments.

4.1.3. Evaluation metrics

1) The output of an LDL algorithm is the label distribution, which is different from the single label output of the single-label learning (SLL) and the label set output of the MLL. Therefore, the evaluation measures for an LDL algorithm are different from the evaluation measures for the SLL and MLL algorithms. The natural choice of evaluation metric for an LDL algorithm is the average distance or similarity between the estimated label distributions obtained by the LDL algorithm and the true label distributions. Thus, we use the following six measures of LDL accuracy for comparing different LDL algorithms:

Chebyshev distance (Cheb) ↓ Clark distance (Clark) ↓
Canberra metric (Canber) ↓ Kullback-Leibler divergence (KL-div) ↓
cosine coefficient (Cosine) ↑ intersection similarity (Intersec) ↑

The first four are distance metrics and the last two are similarity metrics, where the symbol '↓' after the metrics indicates 'the smaller the better', while the symbol '↑' after the metrics indicates 'the larger the better'. How to calculate these metrics and the motivation of using them can be found in [33].

2) To evaluate multilabel classification performance, we choose five widely used MLL metrics, and they are: Hamming loss ↓, ranking loss ↓, one error ↓, coverage ↓, and average precision ↑.

4.1.4. Hyperparameter setting

We set the two algorithmic parameters of the LDML/LDML-R as follows. The number of nearest neighbors in feature extraction is set to $k = 10$. This value is chosen simply to be consistent with the value of k used in the benchmark algorithms of [16]. The influence of the reduced feature dimension d turns out to be not significant. In fact, we have tested the values of d from 1 to 9, and the results obtained are all similar. Hence, we simply choose $d = 8$.

For the comparing algorithms, we use the original algorithmic settings provided by the authors in their publications.

4.2. Experiments for multilabel distribution learning

The 15 datasets with ground-truth label distributions of Table 1 are used in this first set of experiments.

4.2.1. Label distribution learning experimental results

Quantitative experimental results of the eight LDL algorithms applied to these 15 real-world datasets are compared in Tables 3 to 8 for the six evaluation metrics, respectively. In each of these six tables, each row presents the metric values attained by the 8 LDL algorithms together with the rankings achieved in brackets for the corresponding dataset. We also calculate the corresponding algorithms' average ranking performance over the 15 datasets in the last row of each table, where the numerical value before the bracket is the average ranking value, i.e., the sum of the ranks over the 15 datasets divided by 15, and the number in the bracket is again the rank. To indicate the overall performance, Table 9 summarizes the ranking performance of the 8 LDL algorithms averaging over these 15 datasets and the 6 estimation accuracy measures.

From Tables 3 to 9, it can be seen that on average the LDML-R attains the best performance, and the LDML achieves the second best performance, followed by the IIS-LLD as the third best algorithm. The reason for the LDML to outperform the IIS-LLD is that by extracting the features $\boldsymbol{g}_i$ of samples $\boldsymbol{x}_i$ and using the estimated label distributions $\widehat{d}_{\boldsymbol{g}_i}^{y_i^j}$, rather than the binary labels y_i^j, for the extracted features in the maximum entropy model, the LDML is provided with better information than the IIS-LLD. The results also confirm our analysis that the LDML-R outperforms the LDML, in terms of LDL accuracy.

The runtime performance of the 8 LDL algorithms on the 15 datasets of [28] are compared in Table 10. For these 15 datasets, the LDSVR and AA-kNN are clear winners on average, in terms of runtime performance. Our algorithm LDML ranks the third. However, the LDML-R imposes higher computational complexity than the LDML, and it ranks the fifth on average, in terms of runtime complexity, which confirms our previous analysis. Of particular interest is to compare the runtimes of our LDML and LDML-R with that of the IIS-

Table 3: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by Chebyshev distance ↓

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	0.0185(4)	0.0257(5.5)	0.0257(5.5)	0.0487(8)	0.0182(3)	0.0260(7)	0.0151(2)	0.0124(1)
Yeast-cdc	0.0152(6)	0.0147(5)	0.0170(8)	0.0142(4)	0.0156(7)	0.0100(3)	0.0071(1)	0.0091(2)
Yeast-cold	0.0409(3)	0.0442(5)	0.0542(8)	0.0485(7)	0.0427(4)	0.0457(6)	0.0219(2)	0.0164(1)
Yeast-diau	0.0245(3)	0.0313(6.5)	0.0313(6.5)	0.0282(5)	0.0203(2)	0.0357(8)	0.0251(4)	0.0195(1)
Yeast-dtt	0.0310(8)	0.0176(4)	0.0209(6)	0.0204(5)	0.0143(3)	0.0216(7)	0.0082(2)	0.0070(1)
Yeast-elu	0.0118(5)	0.0099(3.5)	0.0093(2)	0.0138(7)	0.0099(3.5)	0.0188(8)	0.0119(6)	0.0079(1)
Yeast-heat	0.0411(7)	0.0308(3)	0.0375(6)	0.0310(4)	0.0304(2)	0.0414(8)	0.0323(5)	0.0121(1)
Yeast-spo	0.0380(6)	0.0342(4)	0.0357(5)	0.0485(8)	0.0339(2)	0.0389(7)	0.0340(3)	0.0274(1)
Yeast-spo5	0.0664(4)	0.1012(7)	0.0969(6)	0.0744(5)	0.0591(3)	0.1156(8)	0.0472(1)	0.0498(2)
Yeast-spoem	0.0099(2.5)	0.0597(8)	0.0099(2.5)	0.0272(6)	0.0431(7)	0.0125(4)	0.0175(5)	0.0046(1)
Human Gene	0.0284(6)	0.0323(8)	0.0125(1)	0.0140(3)	0.0187(4)	0.0130(2)	0.0245(5)	0.0300(7)
Natural Scene	0.1526(6)	0.1388(5)	0.1355(3)	0.2473(8)	0.1892(7)	0.0132(1)	0.1375(4)	0.1257(2)
Movie	0.0876(5)	0.0742(2)	0.0629(1)	0.0975(8)	0.0767(3)	0.0930(6)	0.0816(4)	0.0967(7)
SJAFFE	0.0907(8)	0.0661(5)	0.0828(7)	0.0694(6)	0.0658(4)	0.0613(3)	0.0474(2)	0.0375(1)
SBU_3DFE	0.0984(5)	0.0830(3)	0.1170(7)	0.1008(6)	0.1295(8)	0.0871(4)	0.0739(2)	0.0697(1)
Average rank	5.2333(6)	4.9667(4.5)	4.9667(4.5)	6.0000(8)	4.1667(3)	5.4667(7)	3.2000(2)	2.0000(1)

Table 4: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by Clark distance ↓

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	0.3292(7)	0.3067(4)	0.3109(5)	0.4898(8)	0.3004(3)	0.3111(6)	0.2608(2)	0.1805(1)
Yeast-cdc	0.2031(7)	0.2001(6)	0.2660(8)	0.1397(3)	0.1899(5)	0.1404(4)	0.1170(1)	0.1186(2)
Yeast-cold	0.1248(3)	0.1383(6)	0.1422(8)	0.1390(7)	0.1253(4)	0.1324(5)	0.0529(2)	0.0402(1)
Yeast-diau	0.1680(7)	0.1481(6)	0.1974(8)	0.1323(4)	0.1278(3)	0.1461(5)	0.1145(2)	0.0991(1)
Yeast-dtt	0.0755(8)	0.0507(5)	0.0627(7)	0.0491(4)	0.0398(3)	0.0542(6)	0.0244(2)	0.0168(1)
Yeast-elu	0.1541(6)	0.1251(2.5)	0.1313(4)	0.1592(7)	0.1251(2.5)	0.1931(8)	0.1393(5)	0.1087(1)
Yeast-heat	0.2009(8)	0.1438(2)	0.1730(5)	0.1761(6)	0.1514(3)	0.1851(7)	0.1634(4)	0.0556(1)
Yeast-spo	0.1738(7)	0.1619(4)	0.1712(5)	0.1736(6)	0.1793(8)	0.1561(2)	0.1614(3)	0.1215(1)
Yeast-spo5	0.1323(4)	0.1943(7)	0.1908(6)	0.1504(5)	0.1177(3)	0.2057(8)	0.0991(2)	0.0961(1)
Yeast-spoem	0.0140(2.5)	0.0846(8)	0.0140(2.5)	0.0386(6)	0.0632(7)	0.0176(4)	0.0250(5)	0.0065(1)
Human Gene	3.0756(5)	3.4892(7)	0.9650(1)	1.3913(4)	1.0162(2)	1.0485(3)	3.1507(6)	3.5121(8)
Natural Scene	2.1240(6)	2.1327(7)	2.1043(5)	1.8009(2)	2.2530(8)	1.7982(1)	2.0915(4)	2.0674(3)
Movie	0.4607(6)	0.3387(1)	0.3931(4)	0.4724(7)	0.3861(3)	0.4079(5)	0.3543(2)	0.5201(8)
SJAFFE	0.3215(8)	0.2729(7)	0.2511(6)	0.2174(3)	0.2474(5)	0.2375(4)	0.2152(2)	0.1847(1)
SBU_3DFE	0.3368(7)	0.3112(5)	0.2943(3)	0.3363(6)	0.3509(8)	0.2807(2)	0.2420(1)	0.2967(4)
Average rank	6.1000(8)	5.1667(5.5)	5.1667(5.5)	5.2000(7)	4.5000(3)	4.6667(4)	2.8667(2)	2.3333(1)

Table 5: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by Canberra distance ↓

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	1.0239(5)	1.0452(6)	1.0234(4)	1.4548(8)	1.0085(3)	1.0573(7)	0.8769(2)	0.4740(1)
Yeast-cdc	0.6542(6)	0.6556(7)	0.8938(8)	0.3801(3)	0.5645(5)	0.4443(4)	0.3588(2)	0.3219(1)
Yeast-cold	0.2273(7)	0.2108(3)	0.2198(4)	0.2228(5)	0.2256(6)	0.2387(8)	0.0875(2)	0.0665(1)
Yeast-diau	0.3899(7)	0.3005(6)	0.4733(8)	0.2991(5)	0.2980(4)	0.2742(3)	0.2277(2)	0.1991(1)
Yeast-dtt	0.1267(8)	0.0797(4)	0.1203(7)	0.0829(5)	0.0677(3)	0.0889(6)	0.0470(2)	0.0277(1)
Yeast-elu	0.4645(6)	0.3226(2.5)	0.4069(5)	0.4904(7)	0.3226(2.5)	0.5928(8)	0.4050(4)	0.3167(1)
Yeast-heat	0.4816(8)	0.2935(2)	0.3357(4)	0.3732(6)	0.3376(5)	0.3965(7)	0.3147(3)	0.1181(1)
Yeast-spo	0.3938(8)	0.3318(5)	0.3650(6)	0.3127(3)	0.3838(7)	0.2942(2)	0.3129(4)	0.2365(1)
Yeast-spo5	0.1941(4)	0.3121(7)	0.2825(6)	0.2270(5)	0.1823(3)	0.3409(8)	0.1471(2)	0.0886(1)
Yeast-spoem	0.0198(2.5)	0.1195(8)	0.0198(2.5)	0.0545(6)	0.0883(7)	0.0249(4)	0.0352(5)	0.0093(1)
Human Gene	20.7807(5)	23.3088(7)	6.4936(3)	9.6774(4)	6.3145(1)	6.4525(2)	21.8772(6)	23.6378(8)
Natural Scene	5.3662(7)	5.2818(4)	5.3364(5)	4.6644(2)	5.8775(8)	4.5593(1)	5.3472(6)	5.0009(3)
Movie	0.8623(7)	0.7367(5)	0.7194(3)	0.8318(6)	0.7291(4)	0.6882(1)	0.7190(2)	0.9982(8)
SJAFFE	0.6150(8)	0.5797(7)	0.4754(3)	0.3949(2)	0.5041(5)	0.5339(6)	0.4921(4)	0.3822(1)
SBU_3DFE	0.7412(8)	0.6210(5)	0.6169(4)	0.5703(2)	0.7260(7)	0.5790(3)	0.4902(1)	0.6410(6)
Average rank	6.4333(8)	5.2333(7)	4.8333(6)	4.6000(3)	4.7000(5)	4.6667(4)	3.1333(2)	2.4000(1)

Table 6: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by Kullback-Leibler divergence ↓

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	0.0114(3.5)	0.0114(3.5)	0.0116(5)	0.0317(7)	0.5645(8)	0.0117(6)	0.0076(2)	0.0044(1)
Yeast-cdc	0.0055(6.5)	0.0055(6.5)	0.0092(8)	0.0027(3.5)	0.0049(5)	0.0027(3.5)	0.0018(1.5)	0.0018(1.5)
Yeast-cold	0.0073(3)	0.0095(7.5)	0.0095(7.5)	0.0092(6)	0.0077(4)	0.0082(5)	0.0014(2)	0.0007(1)
Yeast-diau	0.0079(7)	0.0063(5)	0.0108(8)	0.0053(4)	0.0044(3)	0.0065(6)	0.0038(2)	0.0028(1)
Yeast-dtt	0.0027(8)	0.0012(4.5)	0.0020(7)	0.0012(4.5)	0.0008(3)	0.0014(6)	0.0002(2)	0.0001(1)
Yeast-elu	0.0033(6)	0.0023(2.5)	0.0024(4)	0.0037(7)	0.0023(2.5)	0.0055(8)	0.0028(5)	0.0017(1)
Yeast-heat	0.0138(8)	0.0067(2)	0.0099(5)	0.0100(6)	0.0075(3)	0.0116(7)	0.0088(4)	0.0010(1)
Yeast-spo	0.0103(6)	0.0082(2)	0.0098(5)	0.0108(8)	0.0107(7)	0.0083(3)	0.0087(4)	0.0049(1)
Yeast-spo5	0.0119(4)	0.0246(6)	0.0252(7)	0.0147(5)	0.0089(3)	0.0301(8)	0.0062(2)	0.0058(1)
Yeast-spoem	0.0001(2.5)	0.0072(8)	0.0001(2.5)	0.0015(6)	0.0038(7)	0.0003(4)	0.0006(5)	0.00003(1)
Human Gene	0.3242(5)	0.4361(8)	0.0283(1)	0.0594(4)	0.0314(2)	0.0330(3)	0.3309(6)	0.4064(7)
Natural Scene	0.4782(5)	0.4162(4)	0.1398(2)	0.6874(7)	0.7376(8)	0.0109(1)	0.5819(6)	0.3367(3)
Movie	0.0578(6)	0.0409(2)	0.0375(1)	0.0617(7)	0.0450(5)	0.0420(4)	0.0411(3)	0.0736(8)
SJAFFE	0.0412(8)	0.0270(7)	0.0249(6)	0.0191(3)	0.0233(5)	0.0204(4)	0.0162(2)	0.0115(1)
SBU_3DFE	0.0433(6)	0.0361(4)	0.0395(5)	0.0455(7)	0.0565(8)	0.0309(2)	0.0221(1)	0.0326(3)
Average rank	5.6333(7)	4.8333(4)	4.9333(6)	5.6667(8)	4.9000(5)	4.7000(3)	3.1667(2)	2.1667(1)

Table 7: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by cosine coefficient ↑

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	0.9895(4)	0.9882(5)	0.9880(6)	0.9686(8)	0.9903(3)	0.9879(7)	0.9927(2)	0.9943(1)
Yeast-cdc	0.9945(6.5)	0.9945(6.5)	0.9912(8)	0.9973(3.5)	0.9952(5)	0.9973(3.5)	0.9982(1.5)	0.9982(1.5)
Yeast-cold	0.9931(3)	0.9909(7)	0.9908(8)	0.9911(6)	0.9924(4)	0.9922(5)	0.9986(2)	0.9992(1)
Yeast-diau	0.9925(7)	0.9939(5)	0.9897(8)	0.9945(4)	0.9958(3)	0.9936(6)	0.9962(2)	0.9973(1)
Yeast-dtt	0.9974(8)	0.9989(4.5)	0.9981(7)	0.9989(4.5)	0.9992(3)	0.9987(6)	0.9997(2)	0.9999(1)
Yeast-elu	0.9967(6)	0.9977(3)	0.9977(3)	0.9963(7)	0.9977(3)	0.9946(8)	0.9972(5)	0.9984(1)
Yeast-heat	0.9863(8)	0.9937(2)	0.9903(5.5)	0.9903(5.5)	0.9928(3)	0.9884(7)	0.9916(4)	0.9990(1)
Yeast-spo	0.9897(6)	0.9923(2)	0.9903(5)	0.9886(8)	0.9894(7)	0.9917(3)	0.9913(4)	0.9951(1)
Yeast-spo5	0.9880(4)	0.9770(6)	0.9746(7)	0.9857(5)	0.9915(3)	0.9704(8)	0.9941(2)	0.9943(1)
Yeast-spoem	0.9998(2.5)	0.9929(8)	0.9998(2.5)	0.9985(6)	0.9965(7)	0.9997(4)	0.9994(5)	0.9999(1)
Human Gene	0.7972(5)	0.7647(7)	0.9718(1)	0.9420(4)	0.9678(2)	0.9673(3)	0.7690(6)	0.7468(8)
Natural Scene	0.8128(6)	0.8792(4)	0.9953(2)	0.8278(5)	0.7244(8)	1.0000(1)	0.7446(7)	0.8905(3)
Movie	0.9620(7)	0.9669(5)	0.9771(1)	0.9589(8)	0.9731(3)	0.9746(2)	0.9666(6)	0.9723(4)
SJAFFE	0.9547(8)	0.9723(7)	0.9733(6)	0.9784(4)	0.9761(5)	0.9792(3)	0.9838(2)	0.9890(1)
SBU_3DFE	0.9566(5)	0.9623(4)	0.9555(6)	0.9521(7)	0.9382(8)	0.9677(2)	0.9769(1)	0.9657(3)
Average rank	5.7333(8)	5.0667(5.5)	5.0667(5.5)	5.7000(7)	4.4667(3)	4.5667(4)	3.4333(2)	1.9667(1)

Table 8: Experimental results of 8 LDL algorithms on 15 real-world datasets with ground-truth label distributions [28] measured by intersectional similarity ↑

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	0.9453(3)	0.9408(6)	0.9421(5)	0.9173(8)	0.9443(4)	0.9402(7)	0.9516(2)	0.9710(1)
Yeast-cdc	0.9565(6)	0.9559(7)	0.9416(8)	0.9746(3)	0.9622(5)	0.9702(4)	0.9760(2)	0.9787(1)
Yeast-cold	0.9448(6)	0.9486(3)	0.9458(4)	0.9452(5)	0.9429(7)	0.9415(8)	0.9781(2)	0.9836(1)
Yeast-diau	0.9452(7)	0.9570(5)	0.9338(8)	0.9559(6)	0.9585(4)	0.9596(3)	0.9670(2)	0.9720(1)
Yeast-dtt	0.9690(8)	0.9814(4)	0.9700(7)	0.9796(5)	0.9830(3)	0.9784(6)	0.9883(2)	0.9930(1)
Yeast-elu	0.9672(6)	0.9768(2.5)	0.9713(4)	0.9649(7)	0.9768(2.5)	0.9578(8)	0.9709(5)	0.9773(1)
Yeast-heat	0.9191(8)	0.9521(2)	0.9441(5)	0.9387(6)	0.9447(4)	0.9334(7)	0.9460(3)	0.9803(1)
Yeast-spo	0.9338(8)	0.9468(5)	0.9394(6)	0.9489(3)	0.9359(7)	0.9509(2)	0.9476(4)	0.9906(1)
Yeast-spo5	0.9336(4)	0.8988(7)	0.9031(6)	0.9256(5)	0.9409(3)	0.8844(8)	0.9528(2)	0.9700(1)
Yeast-spoem	0.9901(2.5)	0.9403(8)	0.9901(2.5)	0.9728(6)	0.9569(7)	0.9875(4)	0.9825(5)	0.9954(1)
Human Gene	0.7043(5)	0.6785(7)	0.9040(2.5)	0.8567(4)	0.9068(1)	0.9040(2.5)	0.6794(6)	0.6660(8)
Natural Scene	0.6573(5)	0.7171(4)	0.8645(2)	0.6142(6)	0.5522(8)	0.9868(1)	0.5651(7)	0.7257(3)
Movie	0.8566(6)	0.8615(5)	0.8891(2)	0.8550(7)	0.8764(3)	0.8930(1)	0.8696(4)	0.8501(8)
SJAFFE	0.8850(8)	0.8989(7)	0.9144(4)	0.9251(2)	0.9124(5)	0.9081(6)	0.9163(3)	0.9344(1)
SBU_3DFE	0.8718(7)	0.8901(4)	0.8830(6)	0.8942(3)	0.8631(8)	0.8972(2)	0.9144(1)	0.8885(5)
Average rank	5.9667(8)	5.1000(7)	4.8000(5)	5.0667(6)	4.7667(4)	4.6333(3)	3.3333(2)	2.3333(1)

LLD, as all the these algorithms are based on solving similar maximum entropy targets with a similar gradient descent optimization technique. Observe that

Table 9: Ranking performance of 8 LDL algorithms average over 15 datasets with ground-truth label distributions [28] and 6 multilabel distribution estimation accuracy measures

Algorithm	Average rank
LDML-R	1.00 (1)
LDML	2.00 (2)
IIS-LLD	3.83 (3)
LDSVR	4.17 (4)
CPNN	5.42 (5)
BFGS-LLD	5.58 (6)
AA-kNN	6.50 (7)
AA-BP	7.50 (8)

Table 10: Experimental results of 8 LDL algorithms on 15 datasets with ground-truth label distributions [28] measured by runtime [s] ↓

Algorithms	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	LDML	LDML-R
Yeast-alpha	25.3371 (8)	20.6198 (7)	18.2403 (4)	1.0531 (1)	19.9805(6)	1.1045 (2)	15.0913 (3)	19.0666(5)
Yeast-cdc	25.7166 (8)	21.9220 (7)	16.2724 (4)	0.8836 (1)	18.9836 (6)	1.0490 (2)	12.8269(3)	18.1193(5)
Yeast-cold	20.7718 (7)	21.5407 (8)	5.2897 (4)	0.9374 (2)	17.5568 (6)	0.3171 (1)	4.8367 (3)	7.2280(5)
Yeast-diau	23.6153 (7)	35.7157 (8)	9.2031 (5)	0.9131 (1)	18.7653 (6)	1.0043 (2)	6.8951 (3)	7.2450(4)
Yeast-dtt	21.8427 (8)	21.4975 (7)	6.0809 (4)	0.9139 (2)	19.0879 (6)	0.8812 (1)	4.3748 (3)	7.1093(5)
Yeast-elu	25.5688 (7)	26.4238 (8)	14.8539 (4)	0.9085 (1)	19.8465 (6)	1.0313 (2)	11.7568 (3)	18.2103(5)
Yeast-heat	24.1267 (7)	31.2775 (8)	8.7874 (5)	0.9475 (1)	18.5278 (6)	0.9593 (2)	5.8102 (3)	7.2924(4)
Yeast-spo	23.3896 (7)	31.7739 (8)	7.9834 (5)	0.9444 (2)	18.9764 (6)	0.3705 (1)	5.8481 (3)	7.1597(4)
Yeast-spo5	21.9315 (8)	20.1524 (7)	4.3023 (4)	0.8912 (1)	18.0636 (6)	1.1629 (2)	3.6275 (3)	7.2504(5)
Yeast-spoem	21.8737 (8)	17.9856 (7)	3.5101 (4)	1.0004 (2)	17.1523 (6)	0.1556 (1)	2.7987 (3)	7.0937(5)
Human Gene	228.4788 (5)	621.8826 (6)	627.7233 (7)	45.1855 (1)	187.1719 (3)	120.2555 (2)	199.3687 (4)	952.6166(8)
Natural Scene	33.4654 (3)	521.4446 (7)	582.3852 (8)	1.5061 (2)	187.0668 (6)	0.3724 (1)	82.7652 (4)	167.5837(5)
Movie	200.3475 (2)	302.7946 (6)	247.1239 (4)	256.6008 (5)	224.6542 (3)	24.3013 (1)	311.7948(7)	446.2196(8)
SJAFFE	15.6128 (6)	79.1725 (8)	2.4305 (3)	0.0231 (2)	21.9615 (7)	0.0192 (1)	6.6878 (5)	5.9159(4)
SBU_3DFE	32.9427 (7)	118.8989 (8)	19.5154 (5)	1.7065 (2)	25.9362 (6)	0.4548 (1)	8.9834(4)	8.8391(3)
Average rank	6.5333 (7)	7.3333 (8)	4.6667 (4)	1.7333 (2)	5.6667 (6)	1.4667 (1)	3.6000 (3)	5.0000(5)

Table 11: Friedman statistics F_F, in terms of each LDL evaluation metric and the critical value at a significance level of 0.05 (comparing algorithms: 8, datasets: 15)

Evaluation metric	F_F	Critical value
Chebyshev distance	5.7381	2.104
Clark distance	5.1718	
Canberra distance	4.8107	
Kullback-Leibler divergence	4.5878	
cosine coefficient	5.0694	
intersectional similarity	3.8544	
Runtime [s]	41.8214	

the proposed LDML and LDML-R consistently impose lower runtimes than the IIS-LLD except for Human Gene and Movie datasets. As discussed previously, compared with the IIS-LLD, our LDML and LDML-R introduce additional complexity in feature extraction and regression. However, owing to its capability of extracting reduced dimensional features, our LDML and LDML-R impose dramatically lower computational complexity in the iterative maximum entropy based optimization than the IIS-LLD. Consequently, the LDML and LDML-R impose lower overall computational complexity than the IIS-LLD. This is significant, as we already know that the IIS-LLD outperforms the other well-established LLD algorithms on average, in terms of estimation accuracy. Our LDML and LDML-R not only consistently outperform the IIS-LLD, in terms of estimation accuracy, but also impose lower computational complexity.

4.2.2. Statistical validation of label distribution learning experimental results

Friedman test statistically compares relative performance among multiple algorithms over multiple datasets [34]. We first use this test to validate the statistical significance of the performance of various algorithms given in Tables 3 to 8 and 10. Table 11 shows the Friedman statistic F_F and the critical value on each evaluation metric at a significance level of 0.05, among the 8 comparing algorithms and 15 datasets.

As seen from Table 11, the F_F values for all the seven metrics are greater than the critical value, and Nemenyi test [34] can be used as a post hoc test to show the algorithms' relative performances. Specifically, based on Table 11, we use Nemenyi test [34] to check the average ordering comparison between two algorithms. Figures 1 to 7 represent these results with a critical difference (CD) graph for each evaluation metric, respectively. When the significance level is 0.05, the number of comparing algorithms is 8, and the number of datasets is 15, the CD value is $\mathrm{CD} = 2.7110$ for Nemenyi test. In the CD diagram, the average ordering of each algorithm is marked on the same coordinate axis. If the difference between the average order of the two algorithms is less than the CD value, then there exists no significant difference between the two algorithms

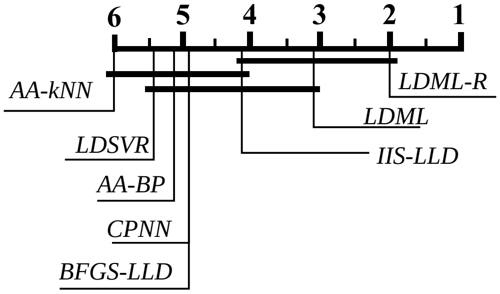

Figure 1: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for Chebyshev distance evaluation metric

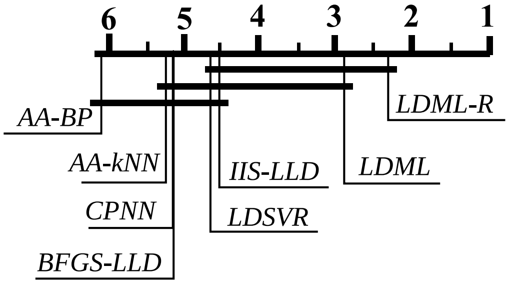

Figure 2: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for Clark distance evaluation metric

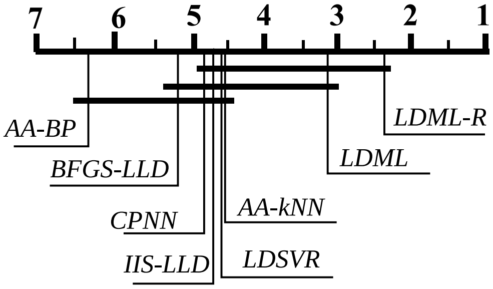

Figure 3: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for Canberra distance evaluation metric

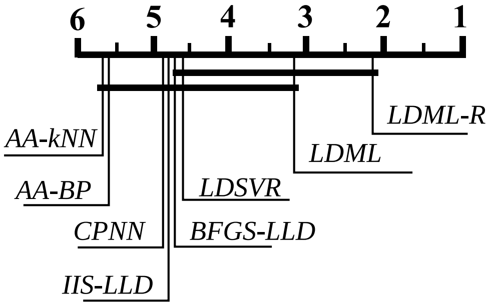

Figure 4: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for Kullback-Leibler divergence metric

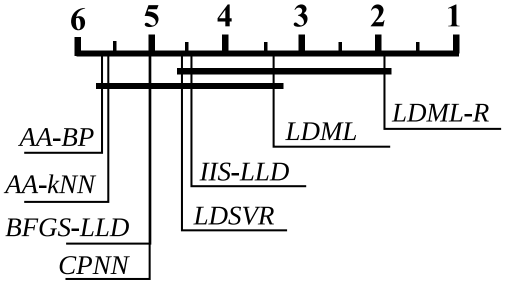

Figure 5: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for cosine coefficient evaluation metric

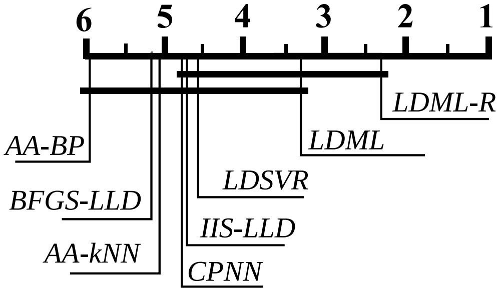

Figure 6: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for intersectional similarity evaluation metric

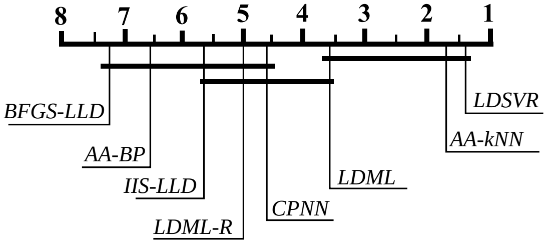

Figure 7: CD diagrams given CD = 2.7110 of Nemenyi tests on the 8 algorithms and 15 datasets for run time (s) evaluation metric

and they are connected by a line segment in the CD graph. Algorithms not connected with the LDML-R in the CD diagrams are considered to have significant performance difference from the control algorithm, given the CD value of 2.7110 at a significance level of 0.05.

The CD diagram of Nemenyi test for Chebyshev distance metric in Figure 1 indicates that the differences for the top three ranking algorithms, the LDML-R, LDML and IIS-LLD, may not be statistically significant, but it is statistically significant that the best LDML-R outperforms the BFGS-LLD, CPNN, AA-BP, LDSVR and AA-kNN. According to the CD diagram of Nemenyi test for the

Table 12: Summary of Nemenyi test results for label distribution learning experiments

Evaluation metric	Statistically significant
Cheb	Top ranking LDML-R superior over BFGS-LLD, CPNN, AA-BP, LDSVR, AA-kNN
Clark	Top ranking LDML-R superior over BFGS-LLD, CPNN, AA-kNN, AA-BP
Canber	Top ranking LDML-R superior over BFGS-LLD, AA-BP
KL-div	Top ranking LDML-R superior over IIS-LLD, CPNN, AA-BP, AA-kNN
Cosine	Top ranking LDML-R superior over CPNN, BFGS-LLD, AA-kNN, AA-BP
Intersec	Top ranking LDML-R superior over AA-kNN, BFGS-LLD, AA-BP

Clark distance metric depicted in Figure 2, it is statistically significant that the top ranking LDML-R outperforms the BFGS-LLD, CPNN, AA-kNN and AA-BP, while for the Canberra distance metric, it is statistically significant that the best LDML-R outperforms the BFGS-LLD and AA-BP, as can be seen from Figure 3. For the Kullback-Leibler divergence, the difference between the top four algorithms may not be statistically significant, but it is statistically significant that the top LDML-R outperforms the IIS-LLD, CPNN, AA-BP and AA-kNN, as seen in Figure 4. Likewise, for the cosine coefficient metric, it is statistically significant that thebest LDML-R outperforms the CPNN, BFGS-LLD, AA-kNN and AA-BP, as confirmed in Figure 5. For the intersectional similarity metric, it is statistically significant that the best LDML-R outperforms the AA-kNN, BFGS-LLD and AA-BP, as confirmed in the CD diagram of Nemenyi test in Figure 6. Table 12 summarizes the statistical Nemenyi test results for label distribution learning experiments. Lastly, the CD diagram of Nemenyi test for the run time of Figure 7 confirms that the differences between the best LDSVR, the second best AA-kNN and the third best LDML may not be statistically significant, but it is statistically significant that the LDML-R imposes higher runtime than the LDSVR.

In addition, Wilcoxon signed-ranks test [34] is employed as the statistical test to show whether the LDML-R performs significantly better than other 7 algorithms, in terms of each evaluation metric. Table 13 summarizes the statistical test results where the p-values for the corresponding tests are given in the brackets. From Table 13, the following observations can be made. The LDML-R achieves statistically better performance than the AA-BP in terms of all the six metrics, and better than the BFGS-LLD, AA-kNN, IIS-LLD in

Table 13: Wilcoxon signed-ranks test among 8 algorithms in terms of Chebyshev distance, Clark distance, Canberra distance, Kullback-Leibler divergence, cosine coefficient and intersectional similarity (significance level $\alpha = 0.05$; p-values shown in the brackets)

LDML-R versus	Evaluation metric					
	Cheb	Clark	Canber	KL-div	cosine	Intersec
AA-BP	WIN[1.53E-3]	WIN[3.53E-2]	WIN[2.15E-2]	WIN[6.71E-3]	WIN[6.71E-3]	WIN[4.27E-3]
BFGS-LLD	WIN[3.36E-3]	WIN[1.50E-2]	TIE[5.53E-2]	WIN[6.71E-3]	WIN[8.36E-3]	WIN[4.27E-3]
CPNN	WIN[2.15E-2]	TIE[7.30E-2]	TIE[7.30E-2]	TIE[3.30E-1]	TIE[1.35E-1]	TIE[2.77E-1]
AA-kNN	WIN[8.54E-4]	TIE[1.69E-1]	TIE[3.59E-1]	WIN[4.79E-2]	WIN[8.36E-3]	WIN[2.15E-2]
IIS-LLD	WIN[1.51E-2]	TIE[7.30E-2]	TIE[6.37E-2]	WIN[4.79E-2]	WIN[1.51E-2]	WIN[3.02E-2]
LDSVR	TIE[5.54E-2]	TIE[3.30E-1]	TIE[3.89E-1]	TIE[3.00E-1]	TIE[2.29E-1]	TIE[3.59E-1]
LDML	WIN[6.23E-1]	TIE[3.03E-1]	TIE[2.08E-1]	TIE[3.00E-1]	WIN[8.42E-2]	TIE[9.46E-2]

majority of the metrics. The LDML-R performs better than the LDML in Cheb and cosine metrics, and is better than the CPNN in Cheb metric, while it is similar with the LDSVR in all the metrics. Wilcoxon signed-ranks statistical test results clearly validate the superior performance of our proposed LDML-R algorithm, in terms of LDL accuracy.

We also employ Bayesian signed-rank test [35] as the statistical test to validate whether the LDML-R performs significantly better than the other 7 algorithms. Table 14 lists the statistical test results, in terms of each evaluation metric, where the values of a, b, c in the bracket $[a, b, c]$ are the probabilities of [WIN, TIE, LOSE] for the corresponding test. Compared with Nemenyi test or Wilcoxon signed-ranks test, Bayesian signed-rank test provides more statistical details. Observe from Table 14 that the LDML-R achieves statistically better performance than the 5 existing LDL benchmark algorithms, including LDSVR, in 4 metrics and it is similar (TIE) with these methods in Kullback-Leibler divergence and cosine coefficient. Also the LDML-R performs better than the LDML in 3 metrics and it is similar with the LDML in the other 3 metrics.

Table 14: Bayesian signed-rank test among 8 algorithms in terms of Chebyshev distance, Clark distance, Canberra distance, Kullback-Leibler divergence, cosine coefficient and intersectional similarity (rope = 0.01; Default prior strength: 0.6)

LDML-R versus	Evaluation metric					
	Cheb↓	Clark↓	Canber↓	KL-div↓	cosine↑	Intersec↑
AA-BP	[0.81994,0.18006,0.0]	[0.98728,0.0,0.01272]	[0.98856,0.0,0.01144]	[0.0729,0.91052,0.01658]	[0.08252,0.90656,0.01092]	[0.9983,0.00132,0.00038]
BFGS-LLD	[0.86354,0.13642,0.00004]	[0.99122,0.0,0.00878]	[0.97166,0.0,0.02834]	[0.19312,0.80148,0.0054]	[0.00886,0.99114,0.0]	[0.97928,0.02072,0.0]
CPNN	[0.85898,0.13526,0.00576]	[0.96268,6e-05,0.03726]	[0.96984,0.0,0.03016]	[0.01672,0.7523,0.23098]	[0.01636,0.89426,0.08938]	[0.87768,0.00112,0.1212]
AA-kNN	[0.97736,0.02264,0.0]	[0.93434,8e-05,0.06558]	[0.84968,0.0,0.15032]	[0.07052,0.9134,0.01608]	[0.17522,0.79884,0.02594]	[0.96672,0.01624,0.01704]
IIS-LLD	[0.64526,0.35474,0.0]	[0.96464,0.0,0.03536]	[0.96538,0.0,0.03462]	[0.26464,0.67342,0.06194]	[0.0611,0.9204,0.0185]	[0.98112,0.00128,0.0176]
LDSVR	[0.85962,0.11018,0.0302]	[0.83664,0.0,0.16336]	[0.82926,0.0,0.17074]	[0.01144,0.78422,0.20434]	[0.01286,0.91122,0.07592]	[0.82536,0.0063,0.16834]
LDML	[0.0022,0.9978,0.0]	[0.80474,0.00522,0.19004]	[0.90786,0.0,0.09214]	[0.01988,0.91868,0.06144]	[0.0093,0.98926,0.00144]	[0.61836,0.37936,0.00228]

4.3. Experiments for multilabel classification performance

The 10 datasets without ground-truth label distributions of Table 2 are used in this second set of experiments with LDL benchmarks.

4.3.1. Multilabel classification experimental results

Half the examples in each dataset are selected randomly as a training set, and the remaining half are used to form a test set. We use 10-fold cross-validation on each dataset and record each algorithm's average performance on the five MLL evaluation metrics in Tables 15 to 19, respectively. The overall ranking performance on multilabel classification, averaged over the ten datasets and the five MLL metrics, are listed in Table 20. It can be seen that on average, our LDML holds the top rank position with our LDML-R in the close second, compared with the other five existing LDL algorithms, AA-kNN, LDSVR, IIS-LLD, CPNN and AA-BP.

4.3.2. Statistical validation of multilabel classification experimental results

Table 21 lists the Friedman statistics F_F and the critical value on the five multilabel classification metrics at a significance level of 0.05, among 7 algorithms and 10 datasets. Figures. 8 to 12 show the results of Nemenyi test [34] with a CD graph for each of the five MLL metrics, respectively. For the significance level 0.05, 7 comparing algorithms and 10 datasets, the CD value is CD = 2.8490 for Nemenyi test. Table 22 summarizes the Nemenyi test results for this set of experiments.

Table 15: Performance comparison of 7 LDL algorithms on 10 real-world datasets without ground-truth label distributions [28] using Hamming loss ↓

Algorithms	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	LDML	LDML-R
Yeast	1.0000 (6.5)	0.3037 (4)	0.6964 (5)	0.2297 (3)	1.0000 (6.5)	0.1939 (1)	0.1950(2)
Emotions	1.0000 (6.5)	0.2996 (3)	0.7097 (5)	0.3006 (4)	1.0000 (6.5)	0.2388 (2)	0.2350(1)
Medical	1.0000 (7)	0.9721 (4)	0.9732 (5)	0.0184 (1)	0.9959 (6)	0.0279(3)	0.0277(2)
Cal500	1.0000 (6.5)	0.1488 (1.5)	0.8522 (5)	0.1814 (4)	1.0000 (6.5)	0.1488 (1.5)	0.1489(3)
Birds	1.0000 (6.5)	0.0517 (2.5)	0.9491 (5)	0.0748 (4)	1.0000 (6.5)	0.0517 (2.5)	0.0510(1)
Image	1.0000 (6.5)	0.7516 (4)	0.7522 (5)	0.2158 (2)	1.0000 (6.5)	0.2054 (1)	0.2484(3)
Scene	1.0000 (6.5)	0.1810 (4)	0.8194 (5)	0.1134 (1)	1.0000 (6.5)	0.1559 (2)	0.1809(3)
Enron	1.0000 (7)	0.0677 (2.5)	0.9339 (5)	0.0705 (4)	0.9919 (6)	0.0677 (2.5)	0.0668(1)
Corel5k	1.0000 (6.5)	0.9907 (4.5)	0.9907 (4.5)	0.0114 (3)	1.0000 (6.5)	0.0093 (2)	0.0092(1)
Bibtex	1.0000 (6.5)	0.0149 (2.5)	0.9853 (5)	0.0165 (4)	1.0000 (6.5)	0.0149 (2.5)	0.0125(1)
Average rank	6.6000 (7)	3.2500 (4)	4.9500 (5)	3.0000 (3)	6.4000 (6)	2.0000 (2)	1.8000(1)

Table 16: Performance comparison of 7 LDL algorithms on 10 real-world datasets without ground-truth label distributions [28] using ranking loss ↓

Algorithms	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	LDML	LDML-R
Yeast	0.5915 (6)	0.4974 (4)	0.9708 (7)	0.5054 (5)	0.4809 (3)	0.2945 (1)	0.3038(2)
Emotions	0.9453 (7)	0.5899 (5)	0.8511 (6)	0.4283 (4)	0.3877 (3)	0.1814 (1)	0.2345(2)
Medical	0.8245 (6)	0.5000 (4)	0.8982 (7)	0.5039 (5)	0.3082 (2)	0.1059 (1)	0.4970(3)
Cal500	0.5126 (5)	0.5005 (4)	0.8621 (7)	0.7750 (6)	0.4937 (3)	0.4617 (1)	0.4836(2)
Birds	0.6485 (6)	0.4374 (4)	0.3132 (1)	0.7335 (7)	0.4157 (3)	0.3159 (2)	0.4858(5)
Image	0.7320 (6)	0.5000 (5)	0.8892 (7)	0.3139 (2)	0.3819 (3)	0.1402 (1)	0.4623(4)
Scene	0.7328 (6)	0.6556 (4)	0.8609 (7)	0.1838 (2)	0.6753 (5)	0.0612 (1)	0.4766(3)
Enron	0.6236 (5)	0.4741 (3)	0.9621 (7)	0.8563 (6)	0.5165 (4)	0.3126 (2)	0.3124(1)
Corel5k	0.5134 (6)	0.5000 (5)	0.4990 (4)	0.9444 (7)	0.4954 (2)	0.4436 (1)	0.4982(3)
Bibtex	0.5243 (5)	0.5012 (4)	0.6954 (6)	0.7416 (7)	0.0000 (1)	0.1017 (2)	0.4994(3)
Average rank	5.8000 (6)	4.2000 (4)	5.9000 (7)	5.1000 (5)	2.9000 (3)	1.3000 (1)	2.8000 (2)

Table 17: Performance comparison of 7 LDL algorithms on 10 real-world datasets without ground-truth label distributions [28] using one error ↓

Algorithms	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	LDML	LDML-R
Yeast	0.7857 (6.5)	0.4286 (4)	0.0714 (1.5)	0.4999 (5)	0.7857 (6.5)	0.0714 (1.5)	0.2857(3)
Emotions	0.0000 (1)	0.6667 (7)	0.3333 (3)	0.4899 (5)	0.3333 (3)	0.3333 (3)	0.5000(6)
Medical	1.0000 (7)	0.5000 (5)	0.4290 (4)	0.1579 (2)	0.5789 (6)	0.3684 (3)	0.1421(1)
Cal500	0.8678 (7)	0.8563 (6)	0.3333 (2)	0.5862 (3)	0.8046 (5)	0.7471 (4)	0.1494(1)
Birds	0.8421 (6)	0.4990 (3)	0.8421 (4.5)	0.4737 (2)	0.9474 (7)	0.8421 (4.5)	0.0526(1)
Image	1.0000 (7)	0.5000 (5)	0.5470 (6)	0.4990 (4)	0.2000 (2.5)	0.0000 (1)	0.2000(2.5)
Scene	1.0000 (6.5)	0.4999 (4)	0.3333 (3)	0.5000 (5)	1.0000 (6.5)	0.0000 (1)	0.1667(2)
Enron	0.9804 (7)	0.9615 (6)	0.5050 (3)	0.4808 (2)	0.9423 (5)	0.6923 (4)	0.0566(1)
Corel5k	0.9865 (7)	0.4890 (4)	0.4400 (2)	0.4419 (3)	0.9797 (6)	0.9390 (5)	0.0116(1)
Bibtex	0.9937 (7)	0.9497 (5)	0.9874 (6)	0.7688 (3)	0.8428 (4)	0.3396 (2)	0.0063(1)
Average rank	6.2000 (7)	4.9000 (5)	3.5000 (4)	3.4000 (3)	5.1500 (6)	2.9000 (2)	1.9500 (1)

Table 18: Performance comparison of 7 LDL algorithms on 10 real-world datasets without ground-truth label distributions [28] using coverage ↓

Algorithms	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	LDML	LDML-R
Yeast	1.4925 (7)	0.8982 (6)	0.8845 (4)	0.8794 (3)	0.8976 (5)	0.8447 (1)	0.8629(2)
Emotions	0.4125 (7)	0.1568 (2)	0.1703 (5)	0.1661 (4)	0.1643 (3)	0.1523 (1)	0.1723(6)
Medical	0.5036 (7)	0.2087 (4.5)	0.2081 (3)	0.1374 (1)	0.1562 (2)	0.2087 (4.5)	0.3398(6)
Cal500	0.2332 (7)	0.2284 (1)	0.2316 (6)	0.2315 (5)	0.2286 (2)	0.2288 (3)	0.2302(4)
Birds	0.2702 (2)	0.3014 (7)	0.2309 (1)	0.2899 (6)	0.2771 (4)	0.2809 (5)	0.2704(3)
Image	0.9980 (7)	0.9608 (1.5)	0.9648 (4)	0.9644 (3)	0.9872 (6)	0.9686 (5)	0.9608(1.5)
Scene	1.2093 (7)	1.0843 (5)	1.0773 (4)	1.0505 (2)	1.1597 (6)	0.9900 (1)	1.0690(3)
Enron	1.0148 (7)	0.4936 (3)	0.5028 (5)	0.4956 (4)	0.4891 (2)	0.4405 (1)	0.7529(6)
Corel5k	1.6825 (7)	1.5023 (4.5)	1.5023 (4.5)	1.5126 (6)	0.1876 (3)	0.1836 (1)	0.1866(2)
Bibtex	0.3562 (5)	0.3382 (3)	0.3598 (7)	0.3585 (6)	0.2068 (1)	0.2632 (2)	0.3477(4)
Average rank	6.3000 (7)	3.7500 (3.5)	4.3500 (6)	4.0000 (5)	3.4000 (2)	2.4500 (1)	3.7500 (3.5)

Table 19: Performance comparison of 7 LDL algorithms on 10 real-world datasets without ground-truth label distributions [28] using average precision ↑

Algorithms	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	LDML	LDML-R
Yeast	0.2675 (7)	0.3965 (4)	0.3064 (6)	0.4779 (3)	0.3125 (5)	0.5123 (2)	0.6910(1)
Emotions	0.3422 (6)	0.4900 (4)	0.3123 (7)	0.4926 (3)	0.4220 (5)	0.6496 (2)	0.6743(1)
Medical	0.0186 (7)	0.0480 (5)	0.0467 (6)	0.3692 (3)	0.2035 (4)	0.9520 (2)	0.9597(1)
Cal500	0.1655 (6)	0.1676 (5)	0.1598 (7)	0.1705 (3)	0.1687 (4)	0.8417 (1)	0.8416(2)
Birds	0.0653 (7)	0.0759 (6)	0.1013 (5)	0.1131 (4)	0.1151 (3)	0.9353 (1)	0.9348(2)
Image	0.1650 (7)	0.2729 (5)	0.2645 (6)	0.5954 (3)	0.3663 (4)	0.7219 (2)	0.7271(1)
Scene	0.1247 (7)	0.7859 (3)	0.2954 (5)	0.7649 (4)	0.1615 (6)	0.8354 (1)	0.7892(2)
Enron	0.0522 (7)	0.0747 (6)	0.0828 (4)	0.1201 (3)	0.0812 (5)	0.9175 (2)	0.9217(1)
Corel5k	0.0140 (6)	0.0141 (4.5)	0.0141 (4.5)	0.0252 (3)	0.0137 (7)	0.9859 (2)	0.9873(1)
Bibtex	0.0155 (6)	0.0226 (4)	0.0182 (5)	0.1111 (3)	NaN (7)	0.9829 (1)	0.9824(2)
Average rank	6.6000 (7)	4.6500 (4)	5.5500(6)	3.2000 (3)	5.0000 (5)	1.6000 (2)	1.4000 (1)

Table 20: Ranking performance of 7 LDL algorithms averaged over 10 datasets without ground-truth label distributions [28] and 5 multilabel classification measures

Algorithm	Average rank
LDML	2.05 (1)
LDML-R	2.34 (2)
AA-kNN	3.74 (3)
LDSVR	4.15 (4)
IIS-LLD	4.57 (5)
CPNN	4.85 (6)
AA-BP	6.30 (7)

Table 21: Friedman statistics F_F, in terms of each MLL evaluation metric and the critical value at a significance level of 0.05 (comparing algorithms 7, datasets 10)

Evaluation metric	F_F	Critical value
Hamming loss	51.3593	2.272
ranking loss	16.3012	
one error	7.7944	
coverage	3.7919	
average precision	50.2244	

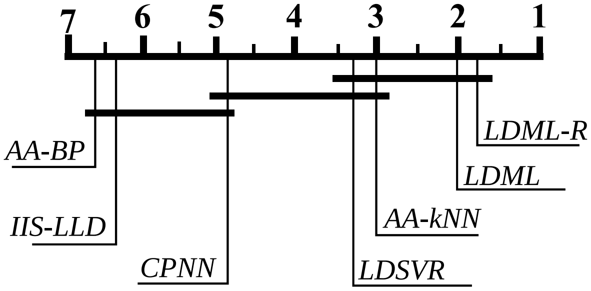

Figure 8: CD diagrams given CD = 2.8490 of Nemenyi tests on the 7 algorithms for Hamming loss evaluation metric

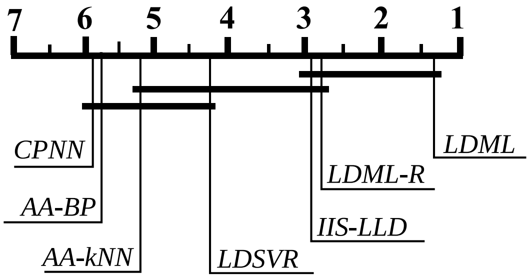

Figure 9: CD diagrams given CD = 2.8490 of Nemenyi tests on the 7 algorithms for ranking loss evaluation metric

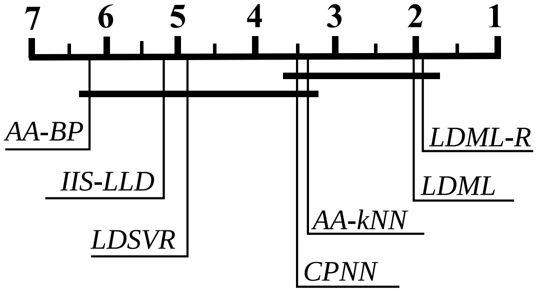

Figure 10: CD diagrams given CD = 2.8490 of Nemenyi tests on the 7 algorithms for one error evaluation metric

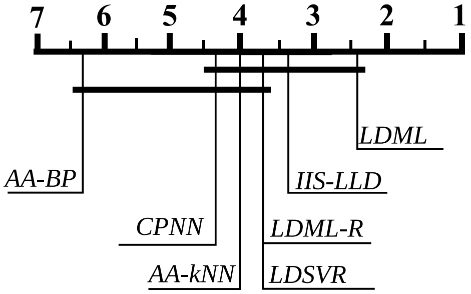

Figure 11: CD diagrams given CD = 2.8490 of Nemenyi tests on the 7 algorithms for coverage evaluation metric

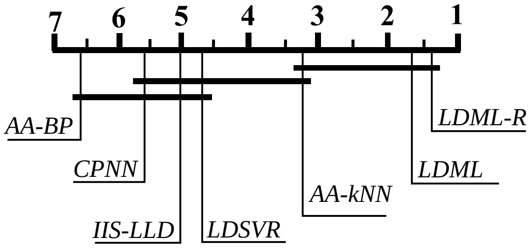

Figure 12: CD diagrams given CD = 2.8490 of Nemenyi tests on the 7 algorithms for average precision evaluation metric

Table 22: Summary of Nemenyi test results for multilabel classification experiments

Evaluation metric	Statistically significant
Hamming loss	Top ranking LDML-R superior over CPNN, IIS-LLD, AA-BP
ranking loss	Top ranking LDML superior over LDSVR, AA-kNN, AA-BP, CPNN
one error	Top ranking LDML-R superior over LDSVR, IIS-LLD, AA-BP
coverage	Top ranking LDML superior over BP-MLL, AA-BP
average precision	Top ranking LDML-R superior over LDSVR, IIS-LLD, CPNN, AA-BP

Table 23: Wilcoxon signed-ranks test among 7 algorithms in terms of Hamming loss, ranking loss, one error, coverage and average precision (significance level $\alpha = 0.05$; p-values shown in the brackets)

LDML-R versus	Evaluation metric				
	Hamming loss	ranking loss	one error	coverage	average precision
AA-BP	WIN[1.95E-3]	WIN[1.95E-3]	WIN[5.86E-3]	WIN[3.91E-3]	WIN[1.95E-3]
CPNN	WIN[1.95E-3]	WIN[5.86E-3]	WIN[2.73E-2]	TIE[9.22E-1]	WIN[1.95E-3]
AA-kNN	WIN[5.57E-1]	TIE[8.40E-2]	WIN[3.91E-3]	TIE[9.22E-1]	WIN[1.95E-3]
IIS-LLD	WIN[1.95E-3]	TIE[8.46E-1]	WIN[1.09E-2]	TIE[6.25E-1]	WIN[3.91E-3]
LDSVR	WIN[5.86E-3]	WIN[2.73E-2]	WIN[1.95E-3]	TIE[9.53E-1]	WIN[1.95E-3]
LDML	TIE[1.0]	WIN[3.91E-3]	TIE[8.40E-2]	WIN[3.71E-2]	TIE[2.32E-1]

Wilcoxon signed-ranks test [34] is next used as the statistical test to show whether the LDML-R performs significantly better than the comparing algorithms. Table 23 summarizes the statistical test results and the p-values for the corresponding tests are also shown in the brackets. As shown in Table 23, the LDML-R achieves statistically better performance than the AA-BP for all the five metrics, and it is better than the CPNN, AA-kNN, IIS-LLD and LDSVR in majority of the metrics. The LDML-R is also better then the LDML in 2 metrics, and they are similar (TIE) in the other 3 metrics.

In addition, Bayesian signed-rank test [35] results are given in Table 24. It can be seen that that the LDML-R is statistically better than the AA-BP, CPNN, AA-kNN, IIS-LLD and LDSVR in almost all the cases with only two exceptions. For the coverage metric, the AA-kNN seems similar with the LDML-R and the IIS-LLD seems to perform better than the LDML-R. Compared with LDML, the LDML-R has one WIN, two TIEs and two LOSEs.

Table 24: Bayesian signed-rank test among 7 algorithms in terms of Hamming loss, ranking loss, one error, coverage and average precision (rope = 0.01; Default prior strength: 0.6)

LDML-R versus	Evaluation metric				
	Hamming loss↓	ranking loss↓	one error↓	coverage↓	average precision↑
AA-BP	[1.0,0.0,0.0]	[1.0,0.0,0.0]	[1.0,0.0,0.0]	[0.99356,0.00644,0.0]	[1.0,0.0,0.0]
CPNN	[0.99998,2e-05,0.0]	[0.9996,0.0001,0.0003]	[0.9965,4e-05,0.00346]	[0.25768,0.24178,0.50054]	[1.0,0.0,0.0]
AA-kNN	[0.65222,0.15744,0.19034]	[0.97668,0.00026,0.02306]	[0.99992,6e-05,2e-05]	[0.3038,0.36444,0.33176]	[1.0,0.0,0.0]
IIS-LLD	[1.0,0.0,0.0]	[0.57546,0.00348,0.42106]	[0.99976,0.0001,0.00014]	[0.25142,0.12512,0.62346]	[0.96306,0.03694,0.0]
LDSVR	[0.87944,0.12056,0.0]	[0.94478,0.04542,0.0098]	[1.0,0.0,0.0]	[0.4565,0.21146,0.33204]	[0.99994,6e-05,0.0]
LDML	[0.0,0.81986,0.18014]	[0.0,0.00108,0.99892]	[0.96696,6e-05,0.03298]	[4e-05,0.09082,0.90914]	[0.23364,0.70158,0.06478]

4.4. Performance comparison with state-of-the-art MLL algorithms

In the previous set of multilabel classification experiments, we compare our the LDML-R and LDML with some existing state-of-the-art LDL algorithms. We now further compare our LDML-R and LDML with the five algorithms, namely, BP-MLL [29], ML^2 [13], ML-kNN [30], MLNB [31], and MLFE [32], using the same 10 real-world datasets of Table 2 with the same five MLL metrics.

The experimental results are listed in Table 25. In terms of average ranking, MLFE ranks the first with the best performance in 13 cases, our LDML ranks the second achieving the best performance in 11 cases, and our LDML-R ranks the third achieving the best performance in 10 cases. In terms of multilabel classification performance, it seems that our proposed LDL algorithms may have lost their edge over best MLL algorithms. This is because for classification the label enhancement algorithms, such as our LDML and LDML-R, have to convert the predicted label distribution results back to logical labels by binarization,

Table 25: MLL performance comparison of 6 algorithms on 10 real-world datasets of Table 2

	Yeast	Emotions	Medical	Cal500	Birds	Image	Scene	Enron	Corel5k	Bibtex
Algorithm					Hamming Loss↓					
ML^2	0.2073	0.2388	0.0114	0.1578	0.0636	0.1642	**0.0847**	0.0546	0.0098	0.0126
ML-kNN	0.1980	0.2706	0.0153	0.1416	0.0546	0.1862	0.0989	0.0620	0.0094	0.0136
MLNB	0.2166	0.2804	0.0339	**0.1395**	0.0779	0.2300	0.1299	0.1145	0.0145	0.0824
MLFE	0.2038	0.2434	**0.0112**	0.1549	0.0615	**0.1616**	0.0903	**0.0543**	0.0101	**0.0124**
BP-MLL	0.4500	0.2987	0.0290	0.1472	0.0683	0.3056	0.2904	0.0682	0.0094	0.0160
LDML	**0.1939**	0.2388	0.0279	0.1488	0.0517	0.2054	0.1559	0.0677	0.0093	0.0149
LDML-R	0.1950	**0.2350**	0.0277	0.1489	**0.0510**	0.2484	0.1809	0.0668	**0.0092**	0.0125
Algorithm					ranking loss↓					
ML^2	0.3022	0.2228	0.1084	0.4721	0.3288	0.1467	**0.0580**	0.3210	0.4177	**0.0897**
ML-kNN	**0.1715**	0.2724	0.0540	0.1928	0.3070	0.1927	0.0931	0.1220	0.2663	0.2234
MLNB	0.2323	0.2150	0.0599	**0.1927**	**0.2157**	0.2420	0.1124	0.1768	**0.1267**	0.1584
MLFE	0.1777	0.2061	**0.0209**	0.2089	0.3210	0.1443	0.0713	**0.0958**	0.3156	0.0914
BP-MLL	0.4450	0.4803	0.2445	0.1996	0.3964	0.7956	0.5992	0.3738	0.2695	0.4764
LDML	0.2945	**0.1814**	0.1059	0.4617	0.3159	**0.1402**	0.0612	0.3126	0.4436	0.1017
LDML-R	0.3038	0.2345	0.4970	0.4836	0.4858	0.4623	0.4766	0.3124	0.4982	0.4994
Algorithm					one error↓					
ML^2	0.2857	0.5000	0.3421	**0.0805**	0.7895	0.2000	**0.0000**	0.6731	0.9360	0.3899
ML-kNN	0.2345	0.4213	0.2492	0.1190	0.7356	0.3600	0.2425	0.3921	0.7892	0.6225
MLNB	0.4170	0.4848	0.4234	0.1190	0.5517	0.4390	0.2851	0.5233	0.8804	0.5876
MLFE	0.2356	0.3708	0.1471	0.1984	0.7471	0.2680	0.2157	0.2608	0.7832	0.3710
BP-MLL	0.7034	0.7022	0.4024	0.1071	0.7989	0.6710	0.8269	0.2642	0.9716	0.4547
LDML	**0.0714**	**0.3333**	0.3684	0.7471	0.8421	**0.0000**	**0.0000**	0.6923	0.9390	0.3396
LDML-R	0.2857	0.5000	**0.1421**	0.1494	**0.0526**	0.2000	0.1667	**0.0566**	**0.0116**	**0.0063**
Algorithm					coverage↓					
ML^2	0.8749	0.1600	0.5236	0.2302	0.2836	0.9510	0.9282	0.4523	**0.1813**	**0.2472**
ML-kNN	0.6414	0.2247	0.3441	**0.1319**	0.3606	1.0420	0.5686	0.1631	0.1978	0.5723
MLNB	**0.2499**	0.2871	0.1925	0.1346	**0.2695**	1.2450	0.6564	0.2313	0.2102	0.3819
MLFE	0.6503	0.1887	**0.1475**	0.1354	0.3763	**0.8410**	**0.4582**	**0.1495**	0.2238	0.2586
BP-MLL	0.8990	0.3089	0.2955	1.3386	0.4415	2.1460	2.0761	0.2369	0.1980	0.7356
LDML	0.8447	**0.1523**	0.2087	0.2288	0.2809	0.9686	0.9900	0.4405	0.1836	0.2632
LDML-R	0.8629	0.1723	0.3398	0.2302	0.2704	0.9608	1.0690	0.7529	0.1866	0.3477
Algorithm					average precision↑					
ML^2	**0.8228**	0.7764	**0.9806**	0.7758	0.6430	0.8555	0.9329	0.9056	0.7059	0.9261
ML-kNN	0.6642	0.7142	0.7695	0.5054	0.6173	0.8187	0.9108	0.5512	0.5233	0.6528
MLNB	0.6936	0.6807	0.5227	0.5120	0.6955	0.7788	0.8993	0.5569	0.3559	0.8209
MLFE	0.6996	**0.7901**	0.8745	0.5377	0.7047	**0.8617**	**0.9385**	0.6581	0.5549	0.8672
BP-MLL	0.4297	0.5161	0.2081	0.4783	0.2460	0.5111	0.4200	0.2057	0.2012	0.0659
LDML	0.5123	0.6496	0.9520	**0.8417**	**0.9353**	0.7219	0.8354	0.9175	0.9859	**0.9829**
LDML-R	0.6910	0.6743	0.9597	0.8416	0.9348	0.7271	0.7892	**0.9217**	**0.9873**	0.9824

Table 26: Wilcoxon signed-ranks test of LDML-R versus 5 state-of-art MLL algorithms and LDML in terms of Hamming loss, ranking loss, one error, coverage and average precision (significance level $\alpha = 0.05$; p-values shown in the brackets)

LDML-R	Evaluation metric				
	Hamming loss	ranking loss	one error	coverage	average precision
ML^2	TIE[6.95E-1]	WIN[5.86E-3]	WIN[6.30E-2]	TIE[4.41E-2]	TIE[1.0]
ML-kNN	TIE[3.75E-3]	WIN[3.91E-3]	WIN[3.71E-2]	TIE[6.95E-1]	TIE[6.45E-2]
MLNB	TIE[2.75E-1]	WIN[1.95E-3]	WIN[9.77E-3]	TIE[3.22E-1]	TIE[8.40E-2]
MLFE	TIE[4.92E-1]	WIN[1.95E-3]	TIE[8.40E-2]	WIN[4.88E-2]	TIE[2.75E-1]
BP-MLL	WIN[1.37E-2]	TIE[1.0]	WIN[3.91E-3]	TIE[8.40E-2]	WIN[1.95E-3]
LDML	TIE[1.0]	WIN[3.91E-3]	TIE[8.40E-2]	WIN[3.71E-2]	TIE[2.32E-1]

which has inherent defects. By contrast, the state-of-the-art MLL algorithms can direct output logical label results for classification.

We also employ Wilcoxon signed-ranks test [34] to test the statistical relationship between LDML-R and the other algorithms, and the corresponding test results are summarized in Table 26. As shown in Table 26, the LDML-R achieves statistically superior performance against the ML^2, ML-kNN and MLNB in the rank loss and one error metrics, and it is better over the BP-MLL in Hamming loss, one error and average precision, while it achieves statistically superior performance over the MLFE ans LDML in the rank loss and coverage metrics. It can be seen that our proposed LDML-R still offers some advantage in multilabel classification over the state-of-the-art MLL algorithms.

Table 27: Bayesian signed-rank test among 7 algorithms in terms of Hamming loss, ranking loss, one error, coverage and average precision (rope = 0.01; Default prior strength: 0.6)

LDML-R versus	Evaluation metric				
	Hamming loss↓	ranking loss↓	one error↓	coverage↓	average precision↑
ML^2	[0.0048,0.69412,0.30108]	[0.0,0.01082,0.98918]	[0.94914,0.01896,0.0319]	[0.10934,0.2716,0.61906]	[0.48756,0.00022,0.51222]
ML-kNN	[0.04556,0.66948,0.28496]	[2e-05,0.0,0.99998]	[0.99406,0.0,0.00594]	[0.30948,0.00336,0.68716]	[0.98676,2e-05,0.01322]
MLNB	[0.81604,0.1186,0.06536]	[0.0,2e-05,0.99998]	[0.99962,2e-05,0.00036]	[0.13698,0.0006,0.86242]	[0.98706,0.0029,0.01004]
MLFE	[0.00014,0.74434,0.25552]	[0.0,0.0,1.0]	[0.98118,0.00022,0.0186]	[0.00704,0.0001,0.99286]	[0.92538,0.0005,0.07412]
BP-MLL	[0.76608,0.23392,0.0]	[0.51472,4e-05,0.48524]	[0.99996,0.0,4e-05]	[0.97966,0.0,0.02034]	[1.0,0.0,0.0]
LDML	[0.0,0.82062,0.17938]	[0.0,0.00072,0.99928]	[0.96578,2e-05,0.0342]	[0.00012,0.0945,0.90538]	[0.23802,0.69812,0.06386]

Bayesian signed-rank test [35] results are given in Table 27. Compared with the ML^2, the LDML-R has 1 WIN, 1 TIE and 3 LOSEs. Compared with the ML-kNN and MLFE, our LDML-R has 2 WINs, 1 TIE and 2 LOSEs. Compared with the MLNB, the LDML-R achieves 3 WINs and 2 LOSEs, while it achieves 5 WINs over the BP-MLL. In addition, the LDML-R has 1 WIN, 2 TIEs and 2 LOSEs when compared with the LDML.

5. Conclusions

We have proposed a new label distribution manifold learning (LDML) approach to learn the unknown label distributions of multilabel data. Our contribution has been three-fold. First, we have proposed a manifold learning based feature extraction to extract the accurate and reduced-dimension features of data. Second, we have proposed two algorithms to estimate the label distributions associated with the extracted features, one being the kernel regression and the other being the LTSA based regression. Third, using the extracted reduced-dimension features and associated label distribution estimates to form the enhanced maximum entropy model has yielded the two algorithms, the LDML associated with the kernel regression and the LDML-R associated with the LTSA regression to enable us to accurately estimate the underlying labels distributions of the multilabel data. Experimental results involving 15 real-life datasets with ground-truth label distributions have demonstrated that our proposed LDML-R algorithm offers the advantages in label distribution estimation accuracy, compared with the latest LDL methods. Experimental results involving 10 real-life datasets without ground-truth label distributions have demonstrated the excellent multilabel classification performance of our LDML-R algorithm compared with the state-of-the-art MLL algorithms.

Acknowledgment

This work was supported by National Natural Science Foundation of China under Grant 61702270 and the China Postdoctoral Science Foundation under grant 2017M621592. Dr Tan would like to thank the sponsorship of Chinese Scholarship Council for funding her research at School of Electronics and Computer Science, University of Southampton, UK.

References

[1] M.-L. Zhang and Z.-H. Zhou, “A review on multi-label learning algorithms,” *IEEE Trans. Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1819-1837, Aug. 2014.

[2] X. Su, R. Wang, and X. Dai, "Contrastive learning-enhanced nearest neighbor mechanism for multi-label text classification," in *Proc. ACL 2022* (Dublin, Ireland), May 22-27, 2022, pp. 672–679.

[3] M. S. Rahman, L. Lapasset, and J. Mothe, "Multi-label classification of aircraft heading changes using neural network to resolve Conflicts," in *Proc. ICAART 2022* (Vienna, Austria), Feb. 3-5, 2022, pp. 403–411.

[4] M.-A. Monshi, J. Poon, and V. Chung, "Distributed deep learning for multi-label chest radiography classification," in *Proc. VISIGRAPP 2022* (Online Streaming), Feb. 6-8, 2022, pp. 949–956.

[5] H.-D. Nguyen, X.-S. Vu, and D.-T. Le, "Modular graph transformer networks for multi-label image classification," in *Proc. AAAI 2021* (virtual conference), Feb. 2-9, 2021, pp. 9092–9100.

[6] Y. Liu, H. Cheng, R. Klopfer, M.-R. Gormley, and T. Schaaf, "Effective convolutional attention network for multi-label clinical document classification," in *Proc. EMNLP 2021* (Punta Cana, Dominican Republic), Nov. 7-11, 2021, pp. 5941–5953.

[7] X. Geng, C. Yin, and Z.-H. Zhou, "Facial age estimation by learning from label distributions," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 10, pp. 2401–2412, Oct. 2013.

[8] X. Geng and M.-G. Ling, "Soft video parsing by label distribution learning," in *Proc. AAAI 2017* (San Francisco, CA, USA), Feb. 4-9, 2017, pp. 1331–1337.

[9] J.-Q. Luo, B. He, Y. Ou, B.-L. Li, and K. Wang, "Topic-based label distribution learning to exploit label ambiguity for scene classification," *Neural Computing and Applications*, vol. 33, pp. 16181–16196, 2021.

[10] M.-G. Ling and X. Geng, "Indoor crowd counting by mixture of Gaussians label distribution learning," *IEEE Trans. Image Processing*, vol. 28, no. 11, pp. 5691–5701, Nov. 2019.

[11] X. Geng, N. Xu, and R.-F. Shao, “Label enhancement for label distribution learning,” *J. Computer Research and Development*, vol. 54, no. 6, pp. 1171–1184, 2017.

[12] P. Liu, X. Wang, S. Wang, W. Ye, X. Xi, and S. Zhang, “Improving embedding-based large-scale retrieval via label enhancement,” in *Proc. EMNLP 2021* (Punta Cana, Dominican Republic), Nov. 7-11, 2021, pp. 133–142.

[13] P. Hou, X. Geng, and M.-L. Zhang, “Multi-label manifold learning,” in *Proc. AAAI 2016* (Phoenix, AZ, USA), Feb. 12-17, 2016, pp. 1680–1686.

[14] S.-T. Roweis and L.-K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[15] Z. Y. Zhang and H. Y. Zha, “Principal manifolds and nonlinear dimensionality reduction via tangent space alignment,” *SIAM J. Scientific Computing*, vol. 26, no. 1, pp. 313–338, 2004.

[16] X. Geng, “Label distribution learning,” *IEEE Trans. Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1734–1748, Jul. 2016.

[17] X. Geng, C. Yin, and Z.-H. Zhou, “Facial age estimation by learning from label distributions,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 10, pp. 2401–2412, Oct. 2013.

[18] X. Geng and Y. Xia, “Head pose estimation based on multivariate label distribution,” in *Proc. CVPR 2014* (Columbus, OH, USA), Jun. 23-28, 2014, pp. 1837-1842.

[19] X. Geng and L.-R. Luo, “Multi-label ranking with inconsistent rankers,” in *Proc. CVPR 2014* (Columbus, OH, USA), June. 23-28, 2014, pp. 3742-3747.

[20] X. Geng and P. Hou, “Pre-release prediction of crowd opinion on movies by label distribution learning,” in *Proc. IJCAI 2015* (Buenos Aires, Argentina), Jul. 23-31, 2015, pp. 3511-3517.

[21] F. Wang and C. Zhang, “Label propagation through linear neighborhoods,” *IEEE Trans. Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55-67, Jan. 2008.

[22] S.-D. Pietra, V.-D. Pietra, and J. Lafferty, “Inducing features of random fields,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 380-393, Apr. 1997.

[23] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf, “A kernel view of the dimensionality reduction of manifolds,” in *Proc. ICML 2004* (Banff, Alberta, Canada), Jul. 4-8, 2004, pp. 1–8.

[24] C. Tan, C. Chen, and J.-H. Guan, “A nonlinear dimension reduction method with both distance and neighborhood preservation,” in *Proc. KSEM 2013* (Dalian, China), Aug. 10-12, 2013, pp. 48–63.

[25] C. Tan and G.-L. Ji, “DKE-RLS: A manifold reconstruction algorithm in label spaces with double kernel embedding-regularized least square,” in *Proc. PRICAI 2018* (Nanjing, China), Aug. 28-31, 2018, pp. 16–28.

[26] X.-J. Zhu, *Semi-Supervised Learning with Graphs.* PhD Thesis CMU-LTI-05-192, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, May 2005.

[27] F. Párez-Cruz, A. Navia-Vázquez, P. L. Alarcón-Diana, and A. Artés-Rodríguez, “An IRWLS procedure for SVR,” in *Proc. 10th European Signal Processing Conf.* (Tampere, Finland), Sep. 4-8, 2000, pp. 1–4.

[28] Mulan: A java library for multi-label learning.
http://mulan.sourceforge.net/datasets-mlc.html, 2018-03-01.

[29] M.-L. Zhang and Z.-H. Zhou, “Multilabel neural networks with applications to functional genomics and text categorization,” *IEEE Trans. Knowledge and Data Engineering*, vol. 18, no. 10, pp. 13381351, Oct. 2006.

[30] 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, Jul. 2007.

[31] M.-L. Zhang, J. M. Peña, and V. Robles, “Feature selection for multi-label naive Bayes classification,” *Information Sciences*, vol. 179, no. 19, pp. 3218–3229, Sep. 2009.

[32] Q.-W. Zhang, Y. Zhong, and M.-L. Zhang, “Feature-induced labeling information enrichment for multi-label learning,” in *Proc. AAAI 2018* (New Orleans, LA, USA), Feb. 2-7, 2018, pp. 4446–4453.

[33] S.-H. Cha, “Comprehensive survey on distance/similarity measures between probability density functions,” *Int. J. Mathematical Models and Methods in Applied Sciences*, vol. 1, no. 4, pp. 300-307, 2007.

[34] J.-Demsar, “Statistical comparisons of classifiers over multiple datasets,” *J. Machine Learning Research*, vol. 7, no. 1, pp. 1–30, 2006.

[35] A. Benavoli, G. Corani, J. Dems̃ar, and M. Zaffalon, “Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis,” *J. Machine Learning Research*, vol. 18, no. 77, pp. 1–36, 2017.