A Novel Probabilistic Label Enhancement Algorithm for Multi-label Distribution Learning

Chao Tan, Sheng Chen, *Fellow, IEEE*, Genlin Ji, and Xin Geng, *Member, IEEE*

Abstract—We propose a novel probabilistic label enhancement algorithm, called PLEA, to solve challenging label distribution learning (LDL) for multi-label classification problems. We adopt the well-known maximum entropy model based label distribution learner. However, unlike the existing LDL algorithms based on the maximum entropy model, we propose to use manifold learning to enhance the label distribution learner. Specifically, the supervised information in the label manifold is utilized in the feature manifold space construction to improve the accuracy of feature extraction, while dramatically reducing the feature dimension. Then the robust linear regression is employed to estimate the label distributions associated with the extracted reduced-dimension features. Using the enhanced reduced-dimension features and their associated estimated label distributions in the maximum entropy model, the unknown true label distributions can be estimated more accurately, while imposing considerably lower computational complexity. We evaluate the proposed PLEA method on a wide-range artificial and high-dimensional real-world datasets. Experimental results obtained demonstrate that our proposed PLEA method has advantages in LDL accuracy and runtime performance, compared to the latest multi-label LDL approaches. The results also show that our PLEA compares favourably with the state-of-the-arts multi-label learning algorithms for classification tasks.

Index Terms—Multi-label classification, label distribution learning, manifold learning, robust linear regression

1 Introduction

Multi-label learning (MLL) [1] is widely used for classification, recognition and retrieval in many areas, such as text [2], voice [3], image [4], and video [5], etc. The data in these applications are often rich in semantics, and hence suitable for modeling using MLL. A known challenging multi-label image classification problem is facial age estimation [6], [7], because aging is a gradually changing random process, exhibiting non-stationary patterns. The work [8] focused on the outlier labels and derived a robust multi-label active learning algorithm based on the maximum correntropy criterion (MCC), while the authors of [9] proposed a robust graph-based semisupervised learning method, where the MCC was used to suppress labeling noise. In the past few years, the deployment of intelligent transport system and, in particular, the research and development of autonomous driving, has become a focus of the scientific and engineering community. One of the many challenges for this grand and complex application is how to fully mine and utilize information from a large number of features hidden in huge amount of vehicular videos. A most common use of vehicular video is to detect and identify an important target in the video, such as target vehicles, traffic scene text, pedestrians, etc. However, comprehensive exploration and practical use of scenes, weather conditions, lane lines and other driving information offer much more value. Since traditional single-label classification is difficult to accurately describe all the information contained in the driving video, MLL has become the research focus in this application.

Traditional methods of MLL generally adopt the uniform label distribution assumption, i.e., the importance of each related label (positive label) to the example is considered equal. However, for many real-world learning problems, the multi labels for describing 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) paradigm [10] was proposed to address this issue. The fundamental assumption of LDL is that each example is represented by a label distribution covering the importance of all its labels.

In most of multi-label applications, the data are usually labeled by multiple logical labels (uniform label distribution), and the true label distribution information is unknown or not provided. Nevertheless, the supervised information in these data essentially follows some kind of label distribution. Although this label distribution is not given explicitly, it is often implicitly contained in the training samples. If it can be recovered by a suitable method, the advantages of mining more semantic information by LDL can be realized. The process of promoting the original logical label into a label distribution is known as label enhancement (LE) [11]. More specifically, by discovering the information of the labels' importance contained in the training samples, logical labels can be transformed into label distributions, and prediction accuracy can be improved. In other words, LE utilizes the correlation between labels hidden in the data to effectively strengthen the supervised information of the examples, which enables LDL to achieve better prediction results. Examples of LE based LDL include the methods of

C. Tan (Corresponding author) and G. Ji are with the School of Computer Science and Technology, Nanjing Normal University, Nanjing 210023, China (E-mails: tutu_tanchao@163.com, glji@njnu.edu.cn).
S. Chen is with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K., and also with King Abdulaziz University, Jeddah 21589, Saudi Arabia (E-mail: sqc@ecs.soton.ac.uk).
X. Geng (Corresponding author) is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China (E-mail: xgeng@seu.edu.cn).
This work was supported by National Natural Science Foundation of China under Grants 61702270 and 62076063, 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.

using prior distributions of head pose and face age [12], [13], the label propagation method commonly used in semi-supervised learning [14], and the manifold learning [15].

Some LE methods assign an identical label distribution to all the examples of the same class. For example, Geng *et al.* [13] proposed an LE algorithm for face age estimation based the prior distribution. From the training examples of the same class, an average Gaussian distribution is learnt, and this label distribution is assigned to every example of the class. However, in practice, the examples of a class will have some subtle differences and this should be reflected in their related label distributions. Therefore, the existing state-of-the-art LE algorithms construct the individual label distributions for the corresponding samples of the same class. These include the algorithm adaptation with backpropagation (AA-BP) and with k-nearest neighbor (AA-kNN) [10], the conditional probabilistic neural network (CPNN) [6], the label distribution SVR (LDSVR) [16], the improved iterative scaling-learning from labeled distribution (IIS-LLD) [6], [10], and the algorithm using the quasi-Newton iterative method called the BFGS [17] to improve the IIS-LLD (BFGS-LLD) [6], [10]. More specifically, these LE algorithms all construct individual label distributions to a sample based on all the original feature vectors of the training samples.

Formally, the goal of LDL is to learn the conditional probability of the label vector conditioned on the input sample. Using the Kullback-Leibler (KL) divergence as a measure of similarity between the two distributions, a reasonable choice for this conditional probability model is the one that achieves the largest conditional entropy while meeting the usual probability constraints [6]. This model is known as the maximum entropy model. The problem of estimating the unknown label distributions is then turned into the problems of estimating the label distributions' parameter vectors. Substituting the logical labels for the unknown label distributions and using all the elements of an input sample as its features in the maximum entropy model enable the estimation of the label distributions' parameter vectors via iterative optimization procedures, such as the IIS-LLD and BFGS-LLD [6], [10]. 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. It can be seen that for the problems with high-dimensional input data, the IIS-LLD and BFGS-LLD methods impose higher computational complexity.

Against the above background, in this paper, we propose a novel probabilistic LE algorithm, referred to as PLEA, for multi-label LDL. Although we also adopt the maximum entropy model, our PLEA algorithm is very different from the IIS-LLD and BFGS-LLD. More specifically, our PLEA consists of the following three components or steps.

1) Manifold space enhanced feature extraction: Based on the local tangent space alignment (LTSA) manifold learning principle [18], we extract accurate and reduced-dimension features in the feature manifold space construction.
2) Robust regression: For the extracted reduced-dimension features, we perform the robust linear regression on the manifold learning enhanced label space to estimate their associated label distributions.
3) Enhanced maximum entropy model based LDL: In the enhanced maximum entropy model, we use the enhanced reduced-dimension features obtained in step 1), rather than the full-dimensional input data as features, and we substitute the logical labels with the estimated enriched label distributions acquired in step 2). A gradient-descent iterative optimization then estimates the unknown true label distributions.

It can be seen that unlike the IIS-LLD and BFGS-LLD which rely only on the original logic label space information and are based on all the original input data vectors, we mine the rich information of the neighbour training samples in the manifold space to better guide the LDL. To the best knowledge of the authors, our PLEA is the first to apply manifold space learning in LDL. The advantages of this manifold space enhanced LDL are elaborated as follows.

- By utilizing the supervisory information of the original label space to guide the feature manifold space learning, the reduced-dimensional principal features of the original samples can be extracted. The selected features are important and can fully reflect the category information of the original samples.
- Using the principal features extracted in the robust linear regression to estimate the associated label distributions effectively exploits the feature manifold space learning to guide the label manifold space learning. The acquired label distribution estimates contain richer supervisory information than their corresponding logical labels.
- By using the reduced-dimensional principal features and their associated label distribution estimates to form the enhanced maximum entropy model, the unknown label distributions can be estimated with enhanced accuracy while potentially imposing lower complexity on the entire LDL procedure.

Extensive experimental results show that our PLEA outperforms a wide range of existing LE learning methods, in terms of both estimation accuracy and run time. The rest of this paper is organized as follows. In Section 2, we briefly introduce the related work. The proposed PLEA method is detailed in Section 3. Extensive experimental results are reported in Section 4. Our conclusions are offered in Section 5.

2 Related work

A large amount of research in the literature have devoted to solving image annotation and multi-label classification problems. In this section, we briefly review the work most relevant to our approach from the perspective of multi-label classification and multi-label distribution.

2.1 Multi-label Classification

In recent years, the academic community has carried out numerous research work on multi-label learning. Zhang and Zhou [19] proposed a back propagation (BP) neural network based method for multi-label learning to classify gene functions and texts, called BP-MLL. Jiang *et al.* [20] proposed a multi-label text classification method based on fuzzy similarity measure and k-nearest neighbor (kNN).

Yu *et al.* [21] proposed a multi-label classification framework based on neighborhood rough set. Liu and Chen [22] advocated an emotion analysis method based on multi-label learning. Ding *et al.* [23] proposed an algorithm for evaluating the majority class cost and the minority class value to deal with multi-label unbalanced data classification problems. A multi-label learning approach was proposed in [24] to learn each label's label-specific function while considering the relevant information in the label space and the related information in the feature space. The multi-label learning method has been widely applied in the fields of text classification [25] and traffic scene text classification for determining the target vehicle's driving trajectory [26].

2.2 Multi-label Distribution

The existing research on multi-label distribution mainly focuses on designing algorithms for LDL. According to [10], there exist three strategies for designing LDL algorithms. The first one is called the problem transformation (PT), which generates a single-label data set based on the label distribution and then uses a single-label learning (SLL) algorithm to learn the converted data set. The algorithms belonging to the first strategy include the PT-support vector machine (PT-SVM) and PT-Bayes [10], which respectively apply SVM and Bayes classifiers. The second one is called the algorithm adaptation (AA), which adapts existing learning algorithms to process label assignments directly. Two representative algorithms of the second strategy are the AA-kNN and AA-BP [10]. 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 BP algorithm is used to training a single layer neural network with multiple outputs as the predicted label distribution.

The last type of algorithms exploits the characteristics of LDL. The two representative algorithms of this strategy are the IIS-LLD and BFGS-LLD [10], which apply the maximum entropy model to learn the label distribution. In addition, Geng and Hou [16] regard LDL as a regression problem and proposed the LDSVR, which applies SVR to process label assignment. Shen *et al.* [27] proposed a LDL forests, which extends the random forest to learn the label distribution. Gao *et al.* [28] provided a deep LDL model, called the deep label distribution learning with label ambiguity.

2.3 Incremental Feature Extraction

Two feature extraction algorithms were presented in [29], [30]. As an effective means of nonlinear dimensionality reduction, manifold learning finds low-dimensional smooth manifold results from high-dimensional observation data. Most manifold learning algorithms process data in batch. That is, all the data must be collected before running the algorithms. These batch-type manifold learning algorithm are ineffective for large data-stream problems in which the data arrives continuously. Many practical applications need to process the real-time data stream, where data are collected sequentially and continuously, such as news text analysis, network data mining, video surveillance and seismic signal detection, etc. These applications require incremental manifold learning algorithms that continuously and efficiently update manifolds on newly arriving data, without performing repeated calculations on the entire data set.

Several incremental manifold learning algorithms exist in the literature. Incremental Isomap algorithm [31] learns the input data stream incrementally. With the incremental Laplacian eigen-mapping algorithm [32], the low-dimensional representation of the dataset is calculated by optimally storing the local neighborhood information, and the sub-manifold analysis by the linear incremental method is used to incrementally learn the new sample. The work [33] proposed the incremental locally linear embedding algorithm to evaluate the mappings of the new samples and re-calculate the projections of the original samples. The incremental LTSA (ILTSA) [34] and the incremental principal component analysis [35] were also proposed.

The aforementioned incremental manifold learning algorithm have certain limitations. For example, the new point may change the local neighborhood and the local distribution of the manifold. Therefore, these algorithms may not guarantee sufficient approximation accuracy. Furthermore, their computational cost may be too high. To mitigate these potential drawbacks, Tan *et al.* [36] proposed a self-adaptive LTSA manifold learning algorithm (SLITSA) based on incremental tangent space to incrementally construct subspaces. The update of local information of sample points is obtained from the feature vectors of existing points and new points. Therefore, there is no need to calculate the entire covariance matrix repeatedly when updating the local tangent space.

3 The Proposed PLEA Algorithm

We now detail our proposed PLEA Algorithm. After providing a brief description of the LDL problem, we discuss the maximum entropy model for the LE learning algorithms, specifically, the IIS-LLD and BFGS-LLD [10]. By highlighting the differences of our approach, it naturally leads to our novel contributions to LE learning, namely, the manifold space enhanced label learning model with robust regression.

3.1 Problem Description

Let $\boldsymbol{x} \in \mathbb{R}^q$ be an input instance, and $\boldsymbol{y} = \left[y^1\ y^2 \cdots y^c\right]^{\mathrm{T}} \in \{-1, +1\}^c$ be its logical class label vector. The degree to which the label y^j, $1 \le j \le c$, describes the example $\boldsymbol{x}$ is defined by the conditional probability $d_{\boldsymbol{x}}^{y^j} = P(y^j|\boldsymbol{x})$. Here, $d_{\boldsymbol{x}}^{y^j} \in [0,\ 1]$, $1 \le j \le c$, and $\sum_{j=1}^{c} d_{\boldsymbol{x}}^{y^j} = 1$. For each example, the descriptiveness of all the labels in the label set builds a data form similar to a probability distribution. Therefore, it is called a label distribution. This label distribution however is unknown. The process of learning the label distribution of a labeled example is called LDL. Formally, given the training dataset $\left\{\boldsymbol{x}_i, \boldsymbol{y}_i\right\}_{i=1}^{n}$, where $\boldsymbol{x}_i = \left[x_i^1\ x_i^2 \cdots x_i^q\right]^{\mathrm{T}}$ and $\boldsymbol{y}_i = \left[y_i^1\ y_i^2 \cdots y_i^c\right]^{\mathrm{T}}$, the goal of LDL is to learn the underlying unknown label distributions $\left\{d_{\boldsymbol{x}_i}^{y_i^1}, d_{\boldsymbol{x}_i}^{y_i^2}, \cdots d_{\boldsymbol{x}_i}^{y_i^c}, \right\}_{i=1}^{n}$. The estimate of $d_{\boldsymbol{x}_i}^{y_i^j}$ can be expressed in the form of the parameterized conditional probability model

$$\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} = P(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j}),\ 1 \le j \le c, 1 \le i \le n, \qquad (1)$$

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 $1 \le j \le c$. For many practical applications, the dimension q can be very large, in thousands or even tens of thousands, and the sample size n is typically very large, while the size of label set c is very small by comparison.

3.2 Maximum Entropy Model

Denote $f_k(\boldsymbol{x}_i, y_i^j) \in \mathbb{R}$ as the kth feature function that relies on both instance $\boldsymbol{x}_i$ and label y_i^j, where $1 \le k \le q$. According to the maximum entropy model [6], [10], $P(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j})$ takes the following exponential form

$$P(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j}) = \frac{1}{Z_i} \exp\left(\sum_{k=1}^{q} w_{i,j}^k f_k(\boldsymbol{x}_i, y_i^j)\right), \quad (2)$$

where the normalization factor

$$Z_i = \sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} w_{i,j}^k f_k(\boldsymbol{x}_i, y_i^j)\right). \quad (3)$$

In [6], [10], the features are further expressed as $f_k(\boldsymbol{x}_i, y_i^j) = y_i^j g_k(\boldsymbol{x}_i)$, where $g_k(\boldsymbol{x}_i)$ is the class-independent kth feature function. Therefore, (2) can be rewritten as follows

$$P(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j}) = \frac{1}{Z_i} \exp\left(\sum_{k=1}^{q} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i)\right). \quad (4)$$

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

$$\begin{aligned} T(\boldsymbol{w}) =& \sum_{i=1}^{n}\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} \ln P(y_i^j|\boldsymbol{x}_i; \boldsymbol{w}_{i,j}) \\ =& \sum_{i=1}^{n}\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_i^j} \sum_{k=1}^{q} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i) \\ & - \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i)\right)\right). \end{aligned} \quad (5)$$

If all the true label distributions $d_{\boldsymbol{x}_i}^{y_i^j}$ and the feature functions $g_k(\boldsymbol{x}_i)$ are available, the target function (5) can be optimized using a strategy similar to the improved iterative scaling (IIS) [37], which is a well-known algorithm that maximizes the possibility of a maximum entropy model. 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 yet to be estimated.

Since $g_k(\boldsymbol{x}_i)$ 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 $g_k(\boldsymbol{x}_i)$ with x_i^k. More specifically, the following 'empirical' target function is adopted [6], [10]

$$\begin{aligned} T_{\mathrm{e}}(\boldsymbol{w}) =& \sum_{i=1}^{n}\sum_{j=1}^{c} y_i^j \sum_{k=1}^{q} (w_{i,j}^k \cdot y_i^j) x_i^k \\ & - \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} (w_{i,j}^k \cdot y_i^j) x_i^k\right)\right). \end{aligned} \quad (6)$$

The IIS-LLD and BFGS-LLD [6], [10] 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.

Clearly, there exists a drawback associated with the aforementioned approach for estimating the label distributions. Since the dimension q for many practical applications is large, say, thousands or tens of thousands, the aforementioned IIS-LLD and BFGS-LLD impose high computational cost. Also, these two algorithms do not really calculate the features $g_k(\boldsymbol{x}_i)$ for $\boldsymbol{x}_i$. Rather, they simply use the kth element of $\boldsymbol{x}_i$ as the kth feature of $\boldsymbol{x}_i$, which is somewhat heuristic. Additionally, the 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 (6).

The main contribution of this paper is to propose the novel PLEA algorithm, the manifold space enhanced label learning with robust regression, which eliminates the aforementioned drawbacks. More specifically, we extract the subset of k_s principal features $g_k(\boldsymbol{x}_i)$, $1 \le k \le k_s$, $k_s \ll q$, for each $\boldsymbol{x}_i$. In particular, based on the smoothness between the label manifold space and feature manifold space, we can perform unsupervised feature manifold space learning to extract these principal features. It can be visualized that each extracted feature vector $\left[g_1(\boldsymbol{x}_i) \cdots g_{k_s}(\boldsymbol{x}_i)\right]^{\mathrm{T}}$ is associated with a set of the unknown label distributions d_i^j, $1 \le j \le c$. Learning these label distributions, i.e., the label manifold space learning, is in turn helped by the previous feature manifold space learning. Specifically, we can employ the robust linear regression to estimate the label distributions d_i^j for the extracted features, and we denote the estimate of d_i^j by $\widetilde{d_i^j}$. Consequently, we have the enhanced reduced-dimensional features $g_k(\boldsymbol{x}_i)$, $1 \le k \le k_s$, and the associated label distribution estimates $\widetilde{d_i^j}$, $1 \le j \le c$, to form an enhanced empirical maximum entropy model. This allow us to estimate the true label distributions, namely, the parameters $\boldsymbol{w}$, with enhanced accuracy and potentially significantly lower computational complexity, based on the gradient-descent iterative optimization.

3.3 Manifold Space Learning Based Feature Extraction

According to the fundamental hypothesis of manifold space [18], each data point can be optimally reconstructed using a linear combination of its neighbors. Ideally, we would like to exploit the label information in the label manifold space for guiding the unsupervised feature extraction in the feature manifold space to improve the accuracy of feature extraction. Strictly speaking, therefore, the way of selecting the best k_s neighbors of $\boldsymbol{x}_i$ is according to the closeness of these neighbors to $\boldsymbol{x}_i$ in the label manifold space, i.e., according to the Hamming distance of these neighbors' labels to the labels of $\boldsymbol{x}_i$. There seems no off-shelf kit available to do this. Fortunately, according to the smoothness property of manifold space [18], the closeness in the label space is transferred to the closeness in the feature space. Therefore, we can find the k_s neighbors of $\boldsymbol{x}_i$ by their closeness in the feature space directly.

Specifically, our aim is to find $\bar{k}_s$ neighbor points $\boldsymbol{x}_{i_1}$, $\cdots$, $\boldsymbol{x}_{i_{\bar{k}_s}}$ for every point $\boldsymbol{x}_i$. The optimal 'average' $\bar{k}_s$ can be determined by minimizing the following cost function

$$\Omega(k) = \sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \sum_{j=1}^{k} \alpha \boldsymbol{x}_{i_j} \right\|^2. \tag{7}$$

According to the statistical validity [35], [36], the coefficient α should be the mean value, i.e., $\alpha = \frac{1}{k}$. The optimal $\bar{k}_s$ can readily be obtained as

$$\bar{k}_s = \arg \min_{c \le k \le \min\{n,q\}} \Omega(k), \tag{8}$$

using an exhaustive search or other optimization algorithms. Clearly, $\boldsymbol{x}_i$ is included in its set of $\bar{k}_s$ neighbors. Note that from practical consideration, we want the dimension of feature vector $\boldsymbol{g}(\boldsymbol{x}_i)$ of (11) to be no smaller than c.

Thus, for each $\boldsymbol{x}_i \in \mathbb{R}^q$, we have its $\bar{k}_s$ neighbors, which can be collected together in the matrix $\boldsymbol{X}_i = \left[\boldsymbol{x}_{i_1} \cdots \boldsymbol{x}_{i_{\bar{k}_s}}\right] \in \mathbb{R}^{q \times \bar{k}_s}$. Clearly, $\boldsymbol{X}_i$ has the full rank $\bar{k}_s$. Next define

$$\boldsymbol{U}_i = \boldsymbol{Q}_i^{\mathrm{T}} \boldsymbol{X}_i \big(\boldsymbol{I}_{\bar{k}_s} - \frac{1}{\bar{k}_s} \mathbf{1}_{\bar{k}_s} \mathbf{1}_{\bar{k}_s}^{\mathrm{T}}\big) \in \mathbb{R}^{\bar{k}_s \times \bar{k}_s}, \tag{9}$$

where $\boldsymbol{I}_k$ is the k-dimensional identity matrix, and $\mathbf{1}_k$ is the k-dimensional vector whose elements are all 1, while $\boldsymbol{Q}_i \in \mathbb{R}^{q \times \bar{k}_s}$ contains the $\bar{k}_s$ left singular vectors of $\boldsymbol{X}_i\big(\boldsymbol{I}_{\bar{k}_s} - \frac{1}{\bar{k}_s} \mathbf{1}_{\bar{k}_s} \mathbf{1}_{\bar{k}_s}^{\mathrm{T}}\big)$ corresponding to its $\bar{k}_s$ positive singular values. Further define

$$\boldsymbol{G}_i = \big(\boldsymbol{I}_{\bar{k}_s} - \frac{1}{\bar{k}_s} \mathbf{1}_{\bar{k}_s} \mathbf{1}_{\bar{k}_s}^{\mathrm{T}}\big)\big(\boldsymbol{I}_{\bar{k}_s} - \boldsymbol{U}_i^{\dagger} \boldsymbol{U}_i\big), \tag{10}$$

where $\boldsymbol{U}_i^{\dagger}$ is the Moore-Penrose generalized inverse of $\boldsymbol{U}_i$. According to [18], the optimal feature vector of $\boldsymbol{x}_i$

$$\boldsymbol{g}(\boldsymbol{x}_i) = \left[g_1(\boldsymbol{x}_i) \cdots g_{k_s}(\boldsymbol{x}_i)\right]^{\mathrm{T}} \in \mathbb{R}^{k_s}, \tag{11}$$

is given by the k_s eigenvectors corresponding to the first k_s smallest eigenvalues of $\boldsymbol{G}_i \boldsymbol{G}_i^{\mathrm{T}}$, where $k_s \le \bar{k}_s$ and $k_s \ll q$. To be more specific, if $\bar{k}_s \ll q$, we simply set $k_s = \bar{k}_s$. Otherwise, we choose a sufficiently small k_s that satisfies $k_s \ll q$.

3.4 Estimating Features' Label distributions

For each extracted feature vector $\boldsymbol{g}(\boldsymbol{x}_i)$, it can be visualized that there exists a set of the c virtual labels. The k_s points from which $\boldsymbol{g}(\boldsymbol{x}_i)$ is extracted are the closest points to $\boldsymbol{x}_i$. Since the closeness in the feature manifold space is transferred to the closeness in the label manifold space, the label sets of these k_s points are the closest to the label set $\{y_i^j\}_{j=1}^c$ of $\boldsymbol{x}_i$. Similarly, 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 $\boldsymbol{g}(\boldsymbol{x}_i)$. That is, $\{d_i^j\}_{j=1}^c$ carry more semantic information to describe $\boldsymbol{g}(\boldsymbol{x}_i)$ more comprehensively than $\{y_i^j\}_{j=1}^c$ [38]. Therefore, there is a need to perform an LDL for $\boldsymbol{g}(\boldsymbol{x}_i)$.

To facilitate this LDL, we propose to model $\{d_i^j\}_{j=1}^c$ by the linear regression model

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

namely, we estimate d_i^j by

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

where $\boldsymbol{\theta}_{i,j} \in \mathbb{R}^{k_s}$ 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 $\boldsymbol{\theta}_{i,j}$, for $1 \le j \le c$ and $1 \le i \le n$, we need to perform the normalization

$$\widetilde{d}_i^j = \frac{\widehat{d}_i^j}{\sum_{l=1}^c \widehat{d}_i^l},\ 1 \le i \le n. \tag{14}$$

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

To reliably estimate the parameters $\boldsymbol{\theta} = \big\{\boldsymbol{\theta}_{i,j}, 1 \le j \le c, 1 \le i \le n\big\}$, we adopt the robust linear regression technique. Specifically, the following robust regression cost function is adopted

$$L(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{c} \|\boldsymbol{\theta}_{i,j}\|^2 + \sum_{i=1}^{n} L_1(r_i), \tag{15}$$

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

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

while the L_1 loss is specified by

$$L_1(r) = \begin{cases} 0, & r < \varepsilon, \\ (r-\varepsilon)^2, & r \ge \varepsilon. \end{cases} \tag{17}$$

The constraints $\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{x}_i)\boldsymbol{\theta}_{i,j} \ge 0$, $\forall i, j$, should be imposed. The standard SVR technique is readily applied to determine $\boldsymbol{\theta}_{i,j}$. More specifically, the iterative reweighed least squares (IRWLS) [39] can readily be used to solve this multi-output robust regression problem.

3.5 Summary of Proposed PLEA

By using the extracted features $g_k(\boldsymbol{x}_i)$ with $1 \le k \le k_s$ and $1 \le i \le n$ as well as the associated label distribution estimates $\widetilde{d}_i^j$ with $1 \le j \le c$ and $1 \le i \le n$ in the maximum entropy model (5), we arrive at the enhanced empirical target function

$$\begin{aligned} \widetilde{T}_{\mathrm{e}}(\boldsymbol{w}) = &\sum_{i=1}^{n} \sum_{j=1}^{c} \widetilde{d}_i^j \sum_{k=1}^{k_s} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i) \\ &- \sum_{i=1}^{n} \ln \left(\sum_{j=1}^{c} \exp \left(\sum_{k=1}^{k_s} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i) \right) \right). \end{aligned} \tag{18}$$

The gradient-descent iterative optimization, IIS [37], can then be applied to find the label distributions' parameters $\boldsymbol{w} = \big\{\boldsymbol{w}_{1,j} \in \mathbb{R}^{k_s}, 1 \le j \le c, 1 \le i \le n\big\}$ by solving the nonlinear equation associated with the lower bound of $\widetilde{T}_{\mathrm{e}}(\boldsymbol{w} + \Delta\boldsymbol{w}) - \widetilde{T}_{\mathrm{e}}(\boldsymbol{w})$, yielding the estimates $P\big(y_i^j | \boldsymbol{x}_i; \boldsymbol{w}_{i,j}\big)$ for all the unknown true label distributions $d_{\boldsymbol{x}_i}^{y_i^j}$.

The proposed PLEA is summarized in Algorithm 1. Since $k_s \ll q$, $\widetilde{d}_i^j$ contains more label information than y_i^j for $g_k(\boldsymbol{x}_i)$, and $g_k(\boldsymbol{x}_i)$, $1 \le k \le k_s$, better represent the features of $\boldsymbol{x}_i$ than x_i^k, $1 \le k \le q$, our PLEA is capable of producing more accurate estimates of label distributions than the IIS-LLD. The computational complexity of the PLEA consists of three parts as summarized below.

Step 1. Feature extraction: The complexity of feature decomposition on $\boldsymbol{G}_i \boldsymbol{G}_i^{\mathrm{T}}$ is on the order of k_s^3, denoted as $\mathsf{O}\big(k_s^3\big)$. Therefore, the complexity of **Step 1.** is $\mathsf{O}\big(n \times k_s^3\big)$.

Algorithm 1: Probabilistic Label Enhancement Algorithm

Require: Multi-label training sample set $\{\boldsymbol{x}_i \in \mathbb{R}^q, \boldsymbol{y}_i = [y_i^1 \cdots y_i^c]^{\mathrm{T}} \in \{0,1\}^c\}_{i=1}^n$.
Ensure: Estimates of label distributions $\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} = P(y_i^j|\boldsymbol{x}_i;\boldsymbol{w}_{i,j}), 1 \le j \le c, 1 \le i \le n$.
1: **Step 1**. Extract features:
2: Use manifold learning based feature extraction of Subsection 3.3 to extract k_s ($k_s \ll q$) features $\boldsymbol{g}(\boldsymbol{x}_i) = [g_1(\boldsymbol{x}_i) \cdots g_{k_s}(\boldsymbol{x}_i)]^{\mathrm{T}}$ of $\boldsymbol{x}_i$ for $1 \le i \le n$.
3: **Step 2**. Estimate label distributions for features:
4: Use IRWLS for solving robust linear regression of Subsection 3.4 to estimate label distributions, $\widetilde{d}_i^j$, $1 \le j \le c, 1 \le i \le n$, of extracted principal features.
5: **Step 3**. Enhanced maximum entropy based LDL:
6: With $g_k(\boldsymbol{x}_i)$, $1 \le k \le k_s$ and $1 \le i \le n$, and $\widetilde{d}_i^j$, $1 \le j \le c$ and $1 \le i \le n$, form enhanced maximum entropy model (18).
7: Use IIS gradient-descent iterative algorithm to find parameters $\boldsymbol{w}_{i,j} = [w_{i,j}^1 \cdots w_{i,j}^{k_s}]^{\mathrm{T}} \in \mathbb{R}^{k_s}, \forall i,j$.
8: **return** $\widehat{d}_{\boldsymbol{x}_i}^{y_i^j} \leftarrow \frac{1}{Z_i} \exp\left(\sum_{k=1}^{k_s} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i)\right), \forall i,j$.

Step 2. Robust linear regression: Let the number of iterations for the IRWLS be upper bounded by I_{irwls}. The complexity per iteration of the IRWLS follows the complexity of SVR, which is $\mathsf{O}(n^3)$. Therefore, the complexity of **Step 2.** is $\mathsf{O}(I_{\mathrm{irwls}} \times n^3)$.
Step 3. Enhanced maximum entropy learner: Let the number of iterations for the IIS be upper bounded by $I_{\mathrm{iis}}^{\mathrm{e}}$. The complexity per iteration of the IIS algorithm is $\mathsf{O}(c \times k_s \times n^2)$. Therefore, the complexity of **Step 3.** is $\mathsf{O}(I_{\mathrm{iis}}^{\mathrm{e}} \times c \times k_s \times n^2)$.

Although the feature selection and in particular the robust regression add additional computational complexity, the complexity of the gradient-descent optimization procedure for estimating label distributions based on the enhanced maximum entropy model (18) is significantly lower than that based on the original maximum entropy model (6). Note that the complexity of the original maximum entropy learner is $\mathsf{O}(I_{\mathrm{iis}}^{\mathrm{o}} \times c \times q \times n^2)$ [6], where $I_{\mathrm{iis}}^{\mathrm{o}}$ denotes the upper bound number of iterations by the IIS to solve (6). Therefore, it is likely that the overall computational complexity of the PLEA is lower than that of the IIS-LLD. This will be further investigated based on the experimental results.

Incidentally, after the manifold space learning based feature extraction, we may construct the following empirical target function

$$\widehat{T}_{\mathrm{e}}(\boldsymbol{w}) = \sum_{i=1}^{n}\sum_{j=1}^{c} y_i^j \sum_{k=1}^{k_s} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i) - \sum_{i=1}^{n} \ln\left(\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{k_s} (w_{i,j}^k \cdot y_i^j) g_k(\boldsymbol{x}_i)\right)\right). \quad (19)$$

The corresponding LDL algorithm is referred to as the PLEA$^-$. Clearly, the PLEA$^-$ has the potential to offer even lower computational complexity than the PLEA, as it does not need to perform the robust linear regression for estimating the features' label distributions. However, because y_i^j contains less label information than $\widetilde{d}_i^j$, the PLEA outperforms the PLEA$^-$, in terms of estimation accuracy. This will be further demonstrated in the experimental study.

4 Experimental Evaluation

4.1 Experiment Setups

As the primary objective is to evaluate the estimation accuracy of the proposed PLEA, namely, how close its label distribution estimates to the ground-true label distributions, we first select 15 multi-label datasets with the known ground-true label distributions from Mulan website [41] for performance evaluation. In this set of experiments, we choose two state-of-the-art MLL algorithms, the ML-KNN [40] and the BP-MLL [19], as well as six well-established LDL algorithms, the AA-BP [10], the BFGS-LLD [10], the CPNN [6], the AA-KNN [10], the IIS-LLD [10] and the LDSVR [16], as the benchmarks for comparison with our algorithm. In addition, we also compare the proposed PLEA with the PLEA$^-$ suggested in Subsection 3.5. As the ground-true label distributions of these datasets are provided, we can evaluate the estimation accuracy by comparing the estimated label distributions with their corresponding ground-true label distributions for these MLL and LDL algorithms.

It is also important to compare the runtime performance of these algorithms, particularly for the datasets with large feature dimensions q. With the exception of Human Gene, the feature dimensions q of the datasets [41] are all larger than their label dimensions. However, except for Movie dataset which has a q close to 2000, most of the 15 datasets do not have large feature dimensions q. To investigate the potential runtime saving of our PLEA over the IIS-LLD, in the second set of experiments, we choose five real-world vehicle video datasets from BRVD [42], which have large feature dimensions q. As these 5 datasets are real-world multi-label datasets, their ground-true label distributions are unknown and we cannot use them to evaluate the label distribution estimation accuracy. But we can use them to compare the runtime performance of various algorithms.

Additionally, it is crucial to evaluate the multi-label classification capability of the proposed PLEA using various

TABLE 1
15 Multi-label datasets with known ground-true label distributions [41] used in experimental evaluation

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

MLL metrics. For this purpose, in the third set of experiments, we select another 10 real-world multi-label datasets from Mulan website [41], which do not have ground-true label distributions, for performance evaluation. The three MLL algorithms, BP-MLL [19], MLNB [43] and ML-kNN [40], as well as the seven LDL algorithms, AA-BP [10], LDSVR [16], CPNN [6], AA-kNN [10], IIS-LLD [10], PLEA, and PLEA$^-$, are used in the performance evaluation.

All the experiments are carried out on Matlab 2019b, running on a PC with i5-6200 2.30 GHz processor of 4 cores and 8GB of RAM.

4.2 Evaluation Using Mulan Datasets with Ground-True Label Distributions

Table 1 summarizes the basic attributes of the 15 datasets from [41]. Because the ground-true label distributions for these multi-label datasets are provided, they are particularly suitable for evaluating the estimation accuracy of an algorithm by comparing the estimated label distributions with their corresponding ground-true label distributions. Specifically, we can evaluate the performance based on a metric that measures the average distance or similarity between the estimated label distributions and the ground-true label distributions. We use the following six metrics [10] to evaluate the estimation accuracy performance:

Chebyshev distance (Cheb) ↓
Clark distance (Clark) ↓
Canberra metric(Canber) ↓
Kullback-Leibler divergence (KL-div) ↓
cosine coefficient (Cosine) ↑
intersection similarity [44] (Intersec) ↑

The first four metrics are distance metrics and the last

TABLE 2
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by Chebyshev distance ↓

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	0.0393 (8)	0.1061 (10)	0.0185 (4)	0.0257 (5.5)	0.0257 (5.5)	0.0487 (9)	0.0182 (3)	0.0260 (7)	0.0165 (2)	0.0150 (1)
Yeast-cdc	0.0297 (9)	0.1073 (10)	0.0152 (6)	0.0147 (5)	0.0170 (8)	0.0142 (4)	0.0156 (7)	0.0100 (3)	0.0081 (1.5)	0.0081 (1.5)
Yeast-cold	0.1678 (10)	0.1259 (9)	0.0409 (2)	0.0442 (5)	0.0542 (8)	0.0485 (7)	0.0427 (4)	0.0457 (6)	0.0423 (3)	0.0180 (1)
Yeast-diau	0.0644 (9)	0.1155 (10)	0.0245 (4)	0.0313 (6.5)	0.0313 (6.5)	0.0282 (5)	0.0203 (3)	0.0357 (8)	0.0196 (2)	0.0194 (1)
Yeast-dtt	0.1749 (10)	0.1257 (9)	0.0310 (8)	0.0176 (4)	0.0209 (6)	0.0204 (5)	0.0143 (3)	0.0216 (7)	0.0136 (2)	0.0088 (1)
Yeast-elu	0.0261 (9)	0.1079 (10)	0.0118 (6)	0.0099 (4.5)	0.0093 (1)	0.0138 (7)	0.0099 (4.5)	0.0188 (8)	0.0096 (2)	0.0098 (3)
Yeast-heat	0.0876 (9)	0.1179 (10)	0.0411 (7)	0.0308 (3)	0.0375 (6)	0.0310 (4)	0.0304 (2)	0.0414 (8)	0.0318 (5)	0.0299 (1)
Yeast-spo	0.1025 (9)	0.1193 (10)	0.0380 (6)	0.0342 (4)	0.0357 (5)	0.0485 (8)	0.0339 (2.5)	0.0389 (7)	0.0339 (2.5)	0.0338 (1)
Yeast-spo5	0.2731 (10)	0.1364 (9)	0.0664 (4)	0.1012 (7)	0.0969 (6)	0.0744 (5)	0.0591 (3)	0.1156 (8)	0.0567 (2)	0.0506 (1)
Yeast-spoem	0.4216 (10)	0.1513 (9)	0.0099 (2)	0.0597 (8)	0.0099 (2)	0.0272 (6)	0.0431 (7)	0.0125 (5)	0.0093 (4)	0.0099 (2)
Human Gene	0.0823 (9)	0.1028 (10)	0.0284 (6)	0.0323 (7)	0.0125 (2)	0.0140 (4)	0.0187 (5)	0.0130 (3)	0.0431 (8)	0.0088 (1)
Natural Scene	0.1493 (7)	0.1240 (3)	0.1526 (8)	0.1388 (5)	0.1355 (4)	0.2473 (10)	0.1892 (9)	0.0132 (1)	0.1445 (6)	0.1106 (2)
Movie	0.2591 (10)	0.1350 (9)	0.0876 (5)	0.0742 (2)	0.0629 (1)	0.0975 (8)	0.0767 (4)	0.0930 (6)	0.0932 (7)	0.0750 (3)
SJAFFE	0.1334 (10)	0.1224 (9)	0.0907 (8)	0.0661 (5)	0.0828 (7)	0.0694 (6)	0.0658 (4)	0.0613 (3)	0.0493 (2)	0.0412 (1)
SBU_3DFE	0.1551 (10)	0.1246 (8)	0.0984 (5)	0.0830 (3)	0.1170 (7)	0.1008 (6)	0.1295 (9)	0.0871 (4)	0.0730 (2)	0.0684 (1)
Average rank	9.27 (10)	9.00 (9)	5.40 (6)	4.97 (4)	5.00 (5)	6.27 (8)	4.67 (3)	5.60 (7)	3.40 (2)	**1.43 (1)**

TABLE 3
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by Clark distance ↓

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	1.0306 (10)	0.4742 (8)	0.3292 (7)	0.3067 (4)	0.3109 (5)	0.4898 (9)	0.3004 (3)	0.3111 (6)	0.2897 (2)	0.2597 (1)
Yeast-cdc	0.6098 (10)	0.4427 (9)	0.2031 (7)	0.2001 (6)	0.2660 (8)	0.1397 (2)	0.1899 (5)	0.1404 (3)	0.1410 (4)	0.1174 (1)
Yeast-cold	0.9252 (10)	0.3335 (9)	0.1248 (3)	0.1383 (6)	0.1422 (8)	0.1390 (7)	0.1253 (4)	0.1324 (5)	0.1210 (2)	0.0453 (1)
Yeast-diau	0.5910 (10)	0.3529 (9)	0.1680 (6)	0.1481 (5)	0.1974 (8)	0.1323 (3)	0.1278 (2)	0.1461 (4)	0.1872 (7)	0.1114 (1)
Yeast-dtt	0.9298 (10)	0.3334 (9)	0.0755 (8)	0.0507 (5)	0.0627 (7)	0.0491 (4)	0.0398 (3)	0.0542 (6)	0.0384 (2)	0.0244 (1)
Yeast-elu	0.4626 (10)	0.4318 (9)	0.1541 (6)	0.1251 (1.5)	0.1313 (4)	0.1592 (7)	0.1251 (1.5)	0.1931 (8)	0.1423 (5)	0.1245 (3)
Yeast-heat	0.7207 (10)	0.3430 (9)	0.2009 (8)	0.1438 (1)	0.1730 (4)	0.1761 (5)	0.1514 (3)	0.1851 (6)	0.1997 (7)	0.1501 (2)
Yeast-spo	0.9345 (10)	0.3488 (9)	0.1738 (7)	0.1619 (3)	0.1712 (4)	0.1736 (6)	0.1793 (8)	0.1561 (2)	0.1730 (5)	0.1503 (1)
Yeast-spo5	0.9345 (10)	0.3541 (9)	0.1323 (4)	0.1943 (7)	0.1908 (6)	0.1504 (5)	0.1177 (3)	0.2057 (8)	0.1128 (2)	0.1105 (1)
Yeast-spoem	0.8528 (10)	0.4249 (9)	0.0140 (3)	0.0846 (8)	0.0140 (3)	0.0386 (6)	0.0632 (7)	0.0176 (5)	0.0132 (1)	0.0140 (3)
Human Gene	5.9841 (9)	8.4928 (10)	3.0756 (7)	3.4892 (8)	0.9650 (2)	1.3913 (6)	1.0162 (3)	1.0485 (4)	1.3765 (5)	0.9147 (1)
Natural Scene	2.1597 (8)	3.9153 (10)	2.1240 (5)	2.1327 (6)	2.1043 (4)	1.8009 (2)	2.2530 (9)	1.7982 (1)	2.1448 (7)	2.0568 (3)
Movie	0.8265 (10)	0.3618 (3)	0.4607 (8)	0.3387 (1)	0.3931 (6)	0.4724 (9)	0.3861 (5)	0.4079 (7)	0.3674 (4)	0.3499 (2)
SJAFFE	0.7202 (10)	0.3449 (9)	0.3215 (8)	0.2729 (7)	0.2511 (6)	0.2174 (3)	0.2474 (5)	0.2375 (4)	0.2138 (2)	0.1620 (1)
SBU_3DFE	0.7075 (10)	0.3455 (8)	0.3368 (7)	0.3112 (5)	0.2943 (4)	0.3363 (6)	0.3509 (9)	0.2807 (3)	0.2481 (2)	0.2401 (1)
Average rank	9.8 (10)	8.6 (9)	6.27 (8)	4.9 (5)	5.27 (6)	5.33 (7)	4.7 (3)	4.8 (4)	3.8 (2)	**1.53 (1)**

TABLE 4
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by Canberra distance ↓

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	4.3526 (10)	2.0118 (9)	1.0239 (5)	1.0452 (6)	1.0234 (4)	1.4548 (8)	1.0085 (3)	1.0573 (7)	1.0008 (2)	0.8726 (1)
Yeast-cdc	2.3191 (10)	1.7145 (9)	0.6542 (6)	0.6556 (7)	0.8938 (8)	0.3801 (2)	0.5645 (5)	0.4443 (3)	0.4692 (4)	0.3610 (1)
Yeast-cold	1.8284 (10)	0.6708 (9)	0.2273 (7)	0.2108 (3)	0.2198 (4)	0.2228 (5)	0.2256 (6)	0.2387 (8)	0.2081 (2)	0.0723 (1)
Yeast-diau	1.5087 (10)	0.9371 (9)	0.3899 (7)	0.3005 (6)	0.4733 (8)	0.2991 (5)	0.2980 (4)	0.2742 (3)	0.2567 (2)	0.2429 (1)
Yeast-dtt	1.8497 (10)	0.6684 (9)	0.1267 (8)	0.0797 (4)	0.1203 (7)	0.0829 (5)	0.0677 (3)	0.0889 (6)	0.0658 (2)	0.0453 (1)
Yeast-elu	1.6892 (10)	1.6156 (9)	0.4645 (5)	0.3226 (2.5)	0.4069 (4)	0.4904 (6)	0.3226 (2.5)	0.5928 (7)	0.7096 (8)	0.3189 (1)
Yeast-heat	1.6901 (10)	0.8422 (9)	0.4816 (8)	0.2935 (1)	0.3357 (3)	0.3732 (5)	0.3376 (4)	0.3965 (7)	0.3848 (6)	0.3349 (2)
Yeast-spo	1.3873 (10)	0.8612 (9)	0.3938 (8)	0.3318 (3)	0.3650 (6)	0.3127 (2)	0.3838 (7)	0.2942 (1)	0.3391 (5)	0.3379 (4)
Yeast-spo5	1.4992 (10)	0.6519 (9)	0.1941 (3)	0.3121 (7)	0.2825 (6)	0.2270 (5)	0.1823 (2)	0.3409 (8)	0.1961 (4)	0.1590 (1)
Yeast-spoem	1.0883 (10)	0.1265 (9)	0.0198 (3)	0.1195 (8)	0.0198 (3)	0.0545 (6)	0.0883 (7)	0.0249 (5)	0.0186 (1)	0.0198 (3)
Human Gene	49.1943 (9)	70.0317 (10)	20.7807 (7)	23.3088 (8)	6.4936 (5)	9.6774 (6)	6.3145 (3)	6.4525 (4)	6.1646 (2)	5.7178 (1)
Natural Scene	5.8285 (9)	5.1420 (4)	5.3662 (7)	5.2818 (5)	5.3364 (6)	4.6644 (2)	5.8775 (10)	4.5593 (1)	5.6182 (8)	4.9419 (3)
Movie	1.5102 (10)	0.7807 (7)	0.8623 (9)	0.7367 (6)	0.7194 (3)	0.8318 (8)	0.7290 (5)	0.6882 (1)	0.7215 (4)	0.6933 (2)
SJAFFE	1.5609 (10)	0.8499 (9)	0.6150 (8)	0.5797 (7)	0.4754 (3)	0.3949 (2)	0.5041 (5)	0.5339 (6)	0.4825 (4)	0.3720 (1)
SBU_3DFE	1.1140 (10)	0.8794 (9)	0.7412 (8)	0.6210 (6)	0.6169 (5)	0.5703 (3)	0.7260 (7)	0.5790 (4)	0.5509 (2)	0.4815 (1)
Average rank	9.87 (10)	8.60 (9)	6.60 (8)	5.30 (7)	5.00 (6)	4.67 (3)	4.90 (5)	4.73 (4)	3.73 (2)	**1.60 (1)**

TABLE 5
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by Kullback-Leibler divergence ↓

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	0.4913 (10)	0.2870 (9)	0.0114 (4.5)	0.0114 (4.5)	0.0116 (6)	0.0317 (8)	0.0101 (3)	0.0117 (7)	0.0091 (2)	0.0075 (1)
Yeast-cdc	0.3083 (10)	0.2706 (9)	0.0055 (6.5)	0.0055 (6.5)	0.0092 (8)	0.0027 (3)	0.0049 (5)	0.0027 (3)	0.0027 (3)	0.0018 (1)
Yeast-cold	1.0126 (10)	0.1385 (9)	0.0073 (3)	0.0095 (7.5)	0.0095 (7.5)	0.0092 (6)	0.0077 (4)	0.0082 (5)	0.0069 (2)	0.0010 (1)
Yeast-diau	0.4543 (10)	0.1944 (9)	0.0079 (6)	0.0063 (4)	0.0108 (8)	0.0053 (3)	0.0044 (2)	0.0065 (5)	0.0101 (7)	0.0036 (1)
Yeast-dtt	1.0118 (10)	0.1386 (9)	0.0027 (8)	0.0012 (4.5)	0.0020 (7)	0.0012 (4.5)	0.0008 (3)	0.0014 (6)	0.0007 (2)	0.0002 (1)
Yeast-elu	0.2396 (9)	0.2638 (10)	0.0033 (6)	0.0023 (2.5)	0.0024 (4)	0.0037 (7)	0.0023 (2.5)	0.0055 (8)	0.0028 (5)	0.0022 (1)
Yeast-heat	0.6071 (10)	0.1791 (9)	0.0138 (8)	0.0067 (1)	0.0099 (4)	0.0100 (5)	0.0075 (3)	0.0116 (6)	0.0128 (7)	0.0074 (2)
Yeast-spo	0.6110 (10)	0.1787 (9)	0.0103 (6)	0.0082 (2)	0.0098 (4)	0.0108 (8)	0.0107 (7)	0.0083 (3)	0.0099 (5)	0.0077 (1)
Yeast-spo5	1.3050 (10)	1.0929 (9)	0.0119 (4)	0.0246 (6)	0.0252 (7)	0.0147 (5)	0.0089 (3)	0.0301 (8)	0.0087 (2)	0.0078 (1)
Yeast-spoem	1.7051 (10)	0.6928 (9)	0.0001 (2)	0.0072 (8)	0.0001 (2)	0.0015 (6)	0.0038 (7)	0.0003 (5)	0.0002 (4)	0.0001 (2)
Human Gene	1.7932 (10)	0.4191 (8)	0.3242 (7)	0.4361 (9)	0.0283 (2)	0.0594 (6)	0.0314 (3)	0.0330 (4)	0.0383 (5)	0.0252 (1)
Natural Scene	0.7917 (10)	0.6062 (6)	0.4782 (5)	0.4162 (4)	0.1398 (2)	0.6874 (7)	0.7376 (9)	0.0109 (1)	0.6891 (8)	0.3268 (3)
Movie	0.8739 (10)	0.1524 (9)	0.0578 (7)	0.0409 (2)	0.0375 (1)	0.0617 (8)	0.0450 (6)	0.0420 (4)	0.0449 (5)	0.0419 (3)
SJAFFE	0.6268 (10)	0.1771 (9)	0.0412 (8)	0.0270 (7)	0.0249 (6)	0.0191 (3)	0.0233 (5)	0.0204 (4)	0.0159 (2)	0.0095 (1)
SBU_3DFE	0.6314 (10)	0.1767 (9)	0.0433 (6)	0.0361 (4)	0.0395 (5)	0.0455 (7)	0.0565 (8)	0.0309 (3)	0.0228 (2)	0.0217 (1)
Average rank	9.93 (10)	8.80 (9)	5.80 (8)	4.83 (5)	4.90 (6)	5.77 (7)	4.70 (3)	4.80 (4)	4.07 (2)	**1.40 (1)**

TABLE 6
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by cosine coefficient ↑

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	0.9888 (5)	0.9788 (9)	0.9895 (4)	0.9882 (6)	0.9880 (7)	0.9686 (10)	0.9903 (3)	0.9879 (8)	0.9908 (2)	0.9928 (1)
Yeast-cdc	0.9971 (5)	0.9943 (9)	0.9945 (7.5)	0.9945 (7.5)	0.9912 (10)	0.9973 (3.5)	0.9952 (6)	0.9973 (3.5)	0.9974 (2)	0.9982 (1)
Yeast-cold	0.9883 (9)	0.9835 (10)	0.9931 (3)	0.9909 (7)	0.9908 (8)	0.9911 (6)	0.9924 (4)	0.9922 (5)	0.9934 (2)	0.9990 (1)
Yeast-diau	0.9884 (9)	0.9611 (10)	0.9925 (6)	0.9939 (4)	0.9897 (8)	0.9945 (3)	0.9958 (2)	0.9936 (5)	0.9900 (7)	0.9963 (1)
Yeast-dtt	0.9927 (10)	0.9934 (9)	0.9974 (8)	0.9989 (4.5)	0.9981 (7)	0.9989 (4.5)	0.9992 (3)	0.9987 (6)	0.9993 (2)	0.9997 (1)
Yeast-elu	0.9930 (10)	0.9931 (9)	0.9967 (6)	0.9977 (3)	0.9977 (3)	0.9963 (7)	0.9977 (3)	0.9946 (8)	0.9968 (5)	0.9978 (1)
Yeast-heat	0.9881 (7)	0.9813 (10)	0.9863 (9)	0.9937 (1)	0.9903 (4.5)	0.9903 (4.5)	0.9928 (3)	0.9884 (6)	0.9874 (8)	0.9930 (2)
Yeast-spo	0.9880 (10)	0.9886 (8.5)	0.9897 (6)	0.9923 (1)	0.9903 (4)	0.9886 (8.5)	0.9894 (7)	0.9917 (3)	0.9899 (5)	0.9920 (2)
Yeast-spo5	0.9076 (10)	0.9524 (9)	0.9880 (4)	0.9770 (6)	0.9746 (7)	0.9857 (5)	0.9915 (3)	0.9704 (8)	0.9920 (2)	0.9927 (1)
Yeast-spoem	0.9788 (10)	0.9803 (9)	0.9998 (2.5)	0.9929 (8)	0.9998 (2.5)	0.9985 (6)	0.9965 (7)	0.9997 (5)	0.9998 (2.5)	0.9998 (2.5)
Human Gene	0.7339 (10)	0.9015 (7)	0.7972 (8)	0.7647 (9)	0.9718 (3)	0.9420 (6)	0.9678 (4)	0.9673 (5)	0.9731 (2)	0.9751 (1)
Natural Scene	0.7389 (7)	0.7385 (8)	0.8128 (6)	0.8792 (4)	0.9953 (2)	0.8278 (5)	0.7244 (9)	1.0000 (1)	0.6799 (10)	0.8920 (3)
Movie	0.9210 (9)	0.8907 (10)	0.9620 (7)	0.9669 (6)	0.9771 (1)	0.9589 (8)	0.9731 (4)	0.9746 (3)	0.9678 (5)	0.9751 (2)
SJAFFE	0.9446 (9)	0.9327 (10)	0.9547 (8)	0.9723 (7)	0.9733 (6)	0.9784 (4)	0.9761 (5)	0.9792 (3)	0.9842 (2)	0.9901 (1)
SBU_3DFE	0.9295 (10)	0.9491 (8)	0.9566 (5)	0.9623 (4)	0.9555 (6)	0.9521 (7)	0.9382 (9)	0.9677 (3)	0.9762 (2)	0.9773 (1)
Average rank	8.67 (9)	9.03 (10)	6.00 (8)	5.20 (5)	5.27 (6)	5.87 (7)	4.80 (3)	4.83 (4)	3.90 (2)	**1.43 (1)**

TABLE 7
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by intersectional similarity ↑

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	0.9976 (1)	0.8000 (10)	0.9453 (3)	0.9408 (7)	0.9421 (6)	0.9173 (9)	0.9443 (5)	0.9402 (8)	0.9446 (4)	0.9519 (2)
Yeast-cdc	0.9999 (1)	0.5000 (10)	0.9565 (7)	0.9559 (8)	0.9416 (9)	0.9746 (3)	0.9622 (6)	0.9702 (4)	0.9687 (5)	0.9758 (2)
Yeast-cold	0.3636 (10)	0.4000 (9)	0.9448 (6)	0.9486 (3)	0.9458 (4)	0.9452 (5)	0.9429 (7)	0.9415 (8)	0.9488 (2)	0.9820 (1)
Yeast-diau	0.6364 (10)	0.7000 (9)	0.9452 (6)	0.9570 (4)	0.9338 (8)	0.9559 (5)	0.9585 (3)	0.9596 (2)	0.9345 (7)	0.9648 (1)
Yeast-dtt	0.3636 (10)	0.4000 (9)	0.9690 (8)	0.9814 (4)	0.9700 (7)	0.9796 (5)	0.9830 (3)	0.9784 (6)	0.9835 (2)	0.9887 (1)
Yeast-elu	0.9965 (1)	0.4000 (10)	0.9672 (7)	0.9768 (3.5)	0.9713 (5)	0.9649 (8)	0.9768 (3.5)	0.9578 (9)	0.9694 (6)	0.9771 (2)
Yeast-heat	0.5455 (10)	0.6000 (9)	0.9191 (8)	0.9521 (1)	0.9441 (4)	0.9387 (5)	0.9447 (3)	0.9334 (7)	0.9339 (6)	0.9451 (2)
Yeast-spo	0.5455 (10)	0.6000 (9)	0.9338 (8)	0.9468 (3)	0.9394 (6)	0.9489 (2)	0.9359 (7)	0.9509 (1)	0.9428 (4.5)	0.9428 (4.5)
Yeast-spo5	0.2727 (10)	0.3000 (9)	0.9336 (4)	0.8988 (7)	0.9031 (6)	0.9256 (5)	0.9409 (3)	0.8844 (8)	0.9433 (2)	0.9494 (1)
Yeast-spoem	0.1818 (10)	0.2000 (9)	0.9901 (3)	0.9403 (8)	0.9901 (3)	0.9728 (6)	0.9569 (7)	0.9875 (5)	0.9907 (1)	0.9901 (3)
Human Gene	0.8380 (7)	0.6800 (9)	0.7043 (8)	0.6785 (10)	0.9040 (3.5)	0.8567 (6)	0.9068 (2)	0.9040 (3.5)	0.8574 (5)	0.9156 (1)
Natural Scene	0.4553 (10)	0.9000 (2)	0.6573 (6)	0.7171 (5)	0.8645 (3)	0.6142 (7)	0.5522 (8)	0.9868 (1)	0.5227 (9)	0.7278 (4)
Movie	0.4545 (10)	0.5000 (9)	0.8566 (7)	0.8615 (6)	0.8891 (2)	0.8550 (8)	0.8764 (4)	0.8930 (1)	0.8758 (5)	0.8881 (3)
SJAFFE	0.5455 (10)	0.6000 (9)	0.8850 (8)	0.8989 (7)	0.9144 (4)	0.9251 (2)	0.9124 (5)	0.9081 (6)	0.9180 (3)	0.9355 (1)
SBU_3DFE	0.5455 (10)	0.6000 (9)	0.8718 (7)	0.8901 (5)	0.8830 (6)	0.8942 (4)	0.8631 (8)	0.8972 (3)	0.9040 (2)	0.9159 (1)
Average rank	8.00 (9)	8.73 (10)	6.40 (8)	5.43 (7)	5.10 (5)	5.33 (6)	4.97 (4)	4.83 (3)	4.23 (2)	**1.97 (1)**

two are similarity metrics. The notation '↓' after a metric indicates 'the smaller the better', while '↑' after a metric means 'the larger the better'. Additionally, the run times of all the 10 algorithms are also compared. Obviously, the runtime is a metric that is the smaller the better, i.e., runtime ↓.

Quantitative experimental results of the 10 algorithms applied to these 15 datasets of [41] are compared in Tables 2 to 7 for the six evaluation metrics measuring the distance between the ground-true label distributions and the estimated label distributions, respectively. In each of these six tables, each row presents the metric values attained by the 10 algorithms together with the rankings achieved in brackets for the corresponding dataset. For example, in Table 2, the entry for the PLEA is 0.0150 (1) for the dataset Yeast-alpha. This indicates that the PLEA achieves the Cheb metric value of 0.0150, and it ranks No.1 among the 10 algorithms for Yeast-alpha. 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 8 summarizes the ranking performance of the 10 algorithms average over the 15 datasets of [41] and the 6 estimation accuracy measures.

The results show that our proposed PLEA consistently performs the best among the 10 algorithms for all the six metrics that measure the estimation accuracy. In particular, observe that the estimated label distributions obtained by the PLEA are more accurate than those by the PLEA$^-$. The

TABLE 8
Estimation accuracy ranking performance of 10 algorithms averaged over 15 datasets of [41] and 6 estimation accuracy measures

Algorithm	Average rank
PLEA	**1.56** (**1**)
PLEA$^-$	3.86 (2)
IIS-LLD	4.79 (3)
LDSVR	4.93 (4)
CPNN	5.09 (5)
BFGS-LLD	5.11 (6)
AA-kNN	5.54 (7)
AA-BP	6.08 (8)
BP-MLL	8.79 (9)
ML-KNN	9.26 (10)

reason is as explained in Subsection 3.5. Using the estimated label distributions $\widetilde{d}_i^j$, rather than the binary labels y_i^j, for the extracted features, the PLEA is provided with more and better information. From Tables 2 to 7, it can be seen that on average the PLEA$^-$ achieves the second best performance. But for each estimation accuracy measure, there always have one to two datasets on which the performance of PLEA$^-$ are poor. For the dataset Natural Scene, for instance, the PLEA$^-$ attains the worst estimation accuracy as measure by the cosine coefficient, while it achieves the second worst estimation accuracy as measured by the intersectional similarity. Also as expected, our proposed PLEA consistently outperforms the IIS-LLD, in terms of estimation accuracy, the latter ranked as the third best on average.

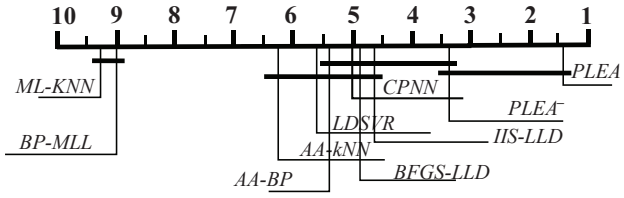
The runtime performance of the 10 algorithms on the 15 datasets of [41] are compared in Table 9. For these 15 datasets, the BP-MLL is the clear winner, in terms of runtime performance. But it has the second worst estimation accuracy. Of particular interest is to compare the computational complexity of the PLEA, PLEA$^-$ and IIS-LLD, as they all are based on similar maximum entropy principle. Observe that the proposed PLEA consistently imposes lower overall complexity than the IIS-LLD with the except of Yeast-spo5 and Yeast-spoem datasets. As discussed in the previous section, compared with the IIS-LLD, our PLEA introduces additional complexity in feature extraction and robust regression, while reducing the computational complexity in the iterative maximum entropy based optimization. For these 15 datasets at least, it seems that the complexity reduction in iterative maximum entropy optimization outweighs the complexity increase in feature extraction and robust regression. Consequently, the PLEA imposes lower overall computational complexity than the IIS-LLD. This is significant, as we already know that the PLEA consistently outperforms the IIS-LLD, in terms of estimation accuracy. Also as expected, the PLEA$^-$ imposes lower overall computational complexity than the PLEA, as the former does not perform robust regression. The aforementioned observation also suggests that the robust regression in the PLEA introduces sizable computational complexity.

TABLE 10
Friedman statistics F_F, in terms of each evaluation metric and the critical value at a significance level of 0.05 (comparing algorithms: 10, datasets: 15)

Evaluation metric	F_F	Critical value
Chebyshev distance	20.8749	1.955
Clark distance	20.2312	
Canberra distance	21.7505	
Kullback-Leibler divergence	23.2253	
cosine coefficient	15.3180	
intersectional similarity	9.0452	
Runtime [s]	86.7684	

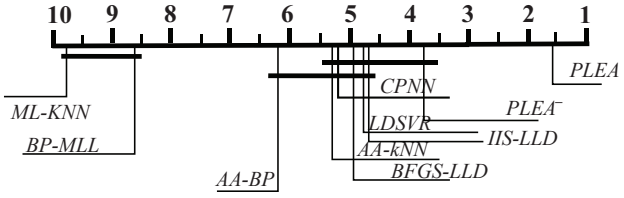
4.2.1 Friedman test and critical difference diagram

Friedman test statistically compares relative performance among multiple algorithms over multiple datasets [45]. We use this test to validate the statistical significance of the performance of various algorithms given in Tables 2 to 7 and 9. Table 10 shows the Friedman statistic F_F and the critical



(a) Chebyshev distance

Fig. 1. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for Chebyshev distance evaluation metric

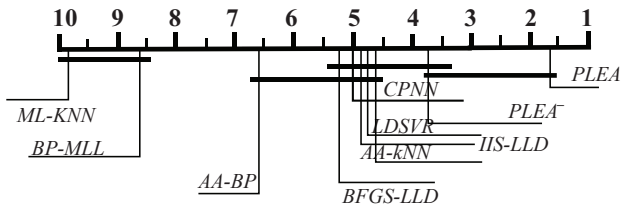


(b) Clark distance

Fig. 2. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for Clark distance evaluation metric

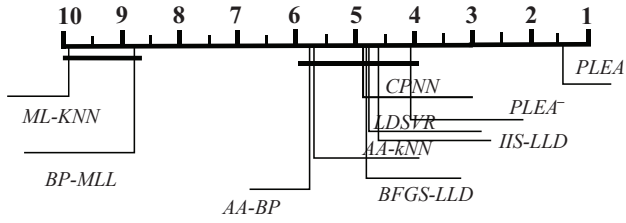
TABLE 9
Experimental results of 10 algorithms on 15 datasets of [41] with ground-true label distributions measured by runtime [s] ↓

Algorithms	ML-KNN	BP-MLL	AA-BP	BFGS-LLD	CPNN	AA-kNN	IIS-LLD	LDSVR	PLEA$^-$	PLEA
Yeast-alpha	69.0379947 (10)	0.3393088 (1)	25.3371071 (9)	20.6197742 (8)	18.2402994 (4)	1.0530555 (2)	19.9805153 (7)	1.1044699 (3)	18.5070840 (5)	18.5150298 (6)
Yeast-cdc	771.2362400 (10)	0.3479317 (1)	25.7166417 (9)	21.9220179 (8)	16.2724184 (4)	0.8835698 (2)	18.9835713 (7)	1.0490428 (3)	17.4905000 (5)	17.5303417 (6)
Yeast-cold	741.5257997 (10)	0.3306586 (2)	20.7717504 (8)	21.5406990 (9)	5.2897021 (4)	0.9374218 (3)	17.5567584 (7)	0.3171071 (1)	16.7180290 (5)	17.2799748 (6)
Yeast-diau	876.8656592 (10)	0.3303416 (1)	23.6153073 (8)	35.7156677 (9)	9.2031102 (4)	0.9130630 (2)	18.7652929 (7)	1.0042549 (3)	17.3959081 (5)	17.9098228 (6)
Yeast-dtt	9720.5132829 (10)	0.3333213 (1)	21.8426624 (9)	21.4975081 (8)	6.0808753 (4)	0.9139258 (3)	19.0879882 (7)	0.8811964 (2)	17.4582517 (5)	17.8935213 (6)
Yeast-elu	45.7656728 (10)	0.3319725 (1)	25.5688326 (8)	26.4238290 (9)	14.8539697 (4)	0.9085137 (2)	19.8464912 (7)	1.0312526 (3)	17.2856214 (5)	17.3790918 (6)
Yeast-heat	1075.5594064 (10)	0.3243420 (1)	24.1267218 (8)	31.2775096 (9)	8.7873933 (4)	0.9475289 (2)	18.5278278 (7)	0.9593309 (3)	17.7321616 (5)	17.9099572 (6)
Yeast-spo	264.0368251 (10)	0.3215843 (1)	23.3895604 (8)	31.7739671 (9)	7.9834021 (4)	0.9443548 (3)	18.9763837 (7)	0.3705112 (2)	17.6141895 (5)	17.8956388 (6)
Yeast-spo5	55.6260337 (10)	0.3116829 (1)	21.9314612 (9)	20.1523984 (8)	4.3023357 (4)	0.8911607 (2)	18.0635962 (5)	1.1629526 (3)	18.0930088 (6)	18.1495418 (7)
Yeast-spoem	600.2141110 (10)	0.3117286 (2)	21.8737154 (9)	17.9855542 (8)	3.5101353 (4)	1.0004449 (3)	17.1522648 (5)	0.1555636 (1)	17.5363122 (6)	17.6775431 (7)
Human Gene	542.2362178 (8)	3.0386092 (1)	228.4788136 (7)	621.8826294 (9)	627.7233122 (10)	45.1854937 (2)	187.1719491 (6)	120.2554692 (3)	154.3660852 (4)	176.1519635 (5)
Natural Scene	1397.6507917 (10)	0.2698749 (1)	33.4653595 (4)	521.4445668 (8)	582.3852263 (9)	1.5060697 (3)	187.0668309 (7)	0.3724432 (2)	141.5362390 (5)	157.9246864 (6)
Movie	654.7098608 (10)	1.1130160 (1)	200.3474911 (3)	302.7946178 (9)	247.1238897 (7)	256.6007756 (8)	224.6541811 (6)	24.3013388 (2)	215.8494958 (4)	216.8764491 (5)
SJAFFE	382.9251875 (10)	0.0442678 (3)	15.6127801 (7)	79.1725374 (9)	2.4304717 (4)	0.0230612 (2)	21.9615007 (8)	0.0192349(1)	2.9851822 (5)	3.2241836 (6)
SBU_3DFE	256.1257912 (10)	0.3322891 (1)	32.9426969 (8)	118.8989318 (9)	19.5153681 (4)	1.7064824 (3)	25.9361933 (7)	0.4548313 (2)	22.6366748 (5)	23.9204424 (6)
Average rank	9.87 (10)	**1.27** (**1**)	7.60 (8)	8.60 (9)	4.93 (4)	2.80 (3)	6.67 (7)	2.27 (2)	5.00 (5)	6.00 (6)



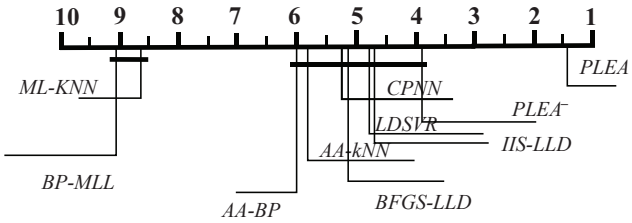
(c) Canberra distance

Fig. 3. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for Canberra distance evaluation metric



(d) Kullback-Leibler divergence

Fig. 4. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for Kullback-Leibler divergence metric



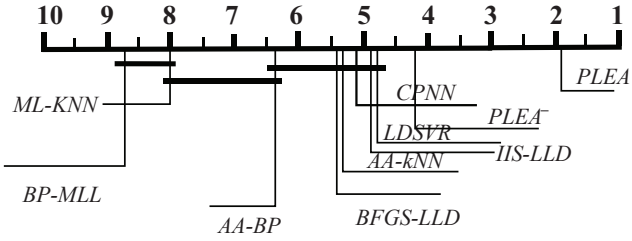
(e) cosine coefficient

Fig. 5. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for cosine coefficient evaluation metric

value on each evaluation metric at a significance level of 0.05, among the 10 comparing algorithms and 15 datasets.

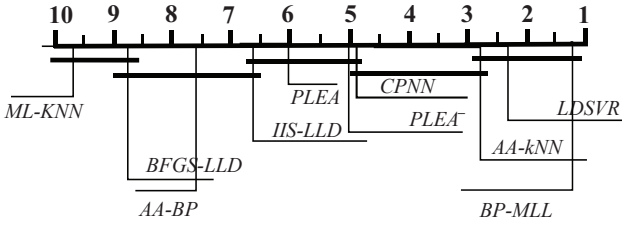
As confirmed in Table 10, the F_F values on all the evaluation metrics are greater than the critical value. Therefore, Bonferroni-Dunn test [45] can be adopted as a post hoc test to show the algorithms' relative performances. Specifically, based on Table 10, we use Nemenyi test [45] to check the average ordering comparison between two algorithms. Figs. 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 comparison algorithms is 10, and the number of datasets is 15, the CD value is CD $= 2.1613$ 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 and they are connected by a line segment in the CD graph. Algorithms not connected with the PLEA in the CD diagrams are considered to have significant performance difference from the control algorithm, given the CD value of 2.1613 at a significance level of 0.05.

From the CD diagrams of Nemenyi tests for the six estimation accuracy metrics depicted in Figs. 1 to 6, it can be seen that only the PLEA$^-$ has line segments connected with the PLEA in the tests for Chebyshev distance metric and Canberra distance metric. Thus our conclusion that the PLEA consistently achieves the best estimation accuracy is



(f) intersectional similarity

Fig. 6. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for intersectional similarity evaluation metric



(g) runtime

Fig. 7. CD diagrams given CD $= 2.1613$ of Nemenyi tests on the 10 algorithms and 15 datasets for run time (s) evaluation metric

statistically very significant.

4.3 Runtime Evaluation Using BRVD Datasets

We further choose five real-world vehicle video datasets [42] to evaluate the runtime performance or computational complexity of an LDL algorithm. As these are real-world datasets, the underlying ground-true label distributions are unknown. Hence we cannot use them to compare the estimation accuracy performance of various algorithms. But these datasets have much large sample size n and feature dimension q. Thus, they are ideal for comparing the runtime performance of various algorithms. Appendix provides the details of how we construct the five multi-label training datasets from the raw real-world vehicle video datasets of [42]. The basic attributes of the constructed five multi-label training datasets are summarized in Table 10.

Table 11 compares the runtime performance of the 10 algorithms for these 5 datasets. Again we are particularly interested in the computational complexity of the PLEA, PLEA$^-$ and IIS-LLD. Observe from Table 11 that the proposed PLEA consistently imposes lower overall computational complexity than the IIS-LLD. Specifically, the PLEA ranks the fourth, while the IIS-LLD ranks the sixth, in terms of average runtime performance. This provides clear empirical evidence that for our PLEA, the complexity reduction in iterative maximum entropy optimization outweighs the complexity increase in feature extraction and robust regression, particularly for the cases of large sample size n and large feature dimension q. Observe also that the PLEA$^-$

TABLE 10
Five real-world vehicle video datasets with unknown ground-true label distributions [42] used in experimental evaluation

Dataset	Examples (n)	Features (q)	Labels (c)
BRVD1	27600	2054	9
BRVD2	28000	6254	9
BRVD3	27600	6059	9
BRVD4	28000	4072	9
BRVD5	47600	2021	9

TABLE 11
Experimental results of 10 algorithms on 5 real-world BRVD datasets of [42] measured by runtime performance [s] ↓

Algorithms	BRVD1	BRVD2	BRVD3	BRVD4	BRVD5	Average rank
ML-KNN	5633220582313 (10)	5649118669804 (10)	5656834598136 (10)	5685747899016 (10)	5731564113797 (10)	10 (10)
BP-MLL	472.2666600 (2)	476.8152100 (2)	466.6050700 (2)	541.4809200 (2)	1001.0416600 (5)	2.6 (2)
AA-BP	664.5604633 (4)	703.7420168 (3)	651.6207993 (3)	678.1440767 (3)	1185.3920020 (6)	3.8 (3)
BFGS-LLD	837.7749656 (7)	12834.6984572 (9)	11609.6843328 (9)	7856.9294133 (9)	1202.6173270 (7)	8.2 (9)
CPNN	854.6336230 (8)	884.0863271 (8)	696.1494858 (5)	746.3470792 (4)	1440.7072256 (8)	6.6 (7)
AA-kNN	905.3774852 (9)	874.5853572 (6)	874.9044343 (8)	983.7558637 (8)	1962.0394104 (9)	8.0 (8)
IIS-LLD	786.6401250 (6)	837.7878476 (5)	790.0793641 (7)	821.1904930 (6)	844.0866898 (3)	5.4 (6)
LDSVR	663.6055487 (3)	874.8968117 (7)	678.7562589 (4)	903.2649110 (7)	983.6192456 (4)	5.0 (5)
PLEA^{-}	403.6007055 (1)	472.4924280 (1)	460.2345572 (1)	493.5997872 (1)	512.4732885 (1)	**1.0** (**1**)
PLEA	756.6830498 (5)	805.4063519 (4)	770.5967830 (6)	801.1521475 (5)	839.1051827 (2)	4.4 (4)

TABLE 12
Characteristics of 10 real-world datasets from [41] with unknown ground-true 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

consistently imposes the lowest overall computational complexity, and it ranks the first, in terms of runtime performance. Noting that the PLEA^{-} imposes significantly lower overall computational complexity than the PLEA, we can see that the robust regression in the PLEA indeed introduces considerable computational complexity, particularly when the sample size n and feature dimension q are very large. This suggests that it is worth investigating alternative low-complexity regression technique for the PLEA to estimate the extracted features' label distributions.

4.4 Evaluation Using Mulan Datasets without Ground-True Label Distributions

Table 12 summarizes the features of the 10 real-world datasets from [41], with unknown ground-true label distributions. These datasets cover a wide range of multi-label attributes. In Table 12, 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. We choose five widely used MLL metrics, and they are: Hamming loss ↓, ranking loss ↓, one error ↓, coverage ↓, and average precision ↑.

In this set of multi-label classification experiments, 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 used 10-fold cross-validation on each dataset, and we record each algorithm's average performance on the five MLL evaluation metrics in Tables 13 to 17, respectively. The overall ranking performance on multi-label classification, averaged over the ten datasets and the five MLL metrics, are listed in Table 18. It can be seen that our proposed PLEA still holds the top rank position on average with the two state-of-the-art MLL algorithms, ML-kNN and MLNB, at the second and third ranking positions. Observe that PLEA^{-} ranks the fourth on average and the existing stat-of-the-art LDL algorithm, IIS-LLD, only ranks the eighth on average, on this set of multi-label classification experiments.

4.4.1 *Friedman test and critical difference diagram*

Table 19 lists the Friedman statistics F_F and the critical value on the five multi-label classification metrics at a significance level of 0.05, among 10 algorithms and 10 datasets. Based on Table 19, we use Nemenyi test [45] to check the average ordering comparison between two comparing algorithms. Figs. 8 to 12 represent the results with a CD graph for each of the five MLL metrics, respectively. The results

TABLE 13
Performance comparison of 10 algorithms on 10 real-world datasets of [41] without ground-true label distributions using Hamming loss ↓

Algorithms	BP-MLL	MLNB	ML-kNN	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	PLEA	PLEA^{-}
Yeast	0.4500 (6)	0.2061 (3)	0.1980 (2)	1.0000 (9.5)	0.3037 (5)	0.6964 (8)	0.2297 (4)	1.0000 (9.5)	0.1945 (1)	0.6963 (7)
Emotions	0.2987 (4)	0.2414 (2)	0.2584 (3)	1.0000 (9.5)	0.2996 (5)	0.7097 (8)	0.3006 (6)	1.0000 (9.5)	0.2406 (1)	0.6507 (7)
Medical	0.0290 (4)	0.0362 (5)	0.0178 (2)	1.0000 (10)	0.9721 (7)	0.9732 (8)	0.0184 (3)	0.9959 (9)	0.0115 (1)	0.9070 (6)
Cal500	0.1472 (2)	0.2062 (6)	0.1416 (1)	1.0000 (9.5)	0.1488 (3)	0.8522 (8)	0.1814 (5)	1.0000 9.5)	0.1596 (4)	0.7025 (7)
Birds	0.0683 (4)	0.0704 (5)	0.0546 (2)	1.0000 (9.5)	0.0517 (1)	0.9491 (8)	0.0748 (6)	1.0000 (9.5)	0.0645 (3)	0.4921 (7)
Image	0.3056 (5)	0.2108 (3)	0.1888 (2)	1.0000 (9.5)	0.7516 (6.5)	0.7522 (8)	0.2158 (4)	1.0000 (9.5)	0.1654 (1)	0.7516 (6.5)
Scene	0.2904 (6)	0.1225 (4)	0.0962 (2)	1.0000 (9.5)	0.1810 (5)	0.8194 (8)	0.1134 (3)	1.0000 (9.5)	0.0847 (1)	0.7446 (7)
Enron	0.0682 (4)	0.1162 (6)	0.0623 (2)	1.0000 (10)	0.0677 (3)	0.9339 (8)	0.0705 (5)	0.9919 (9)	0.0546 (1)	0.8892 (7)
Corel5k	0.0094 (2)	0.0138 (5)	0.0093 (1)	1.0000 (9.5)	0.9907 (7.5)	0.9907 (7.5)	0.0114 (4)	1.0000 (9.5)	0.0098 (3)	0.9877 (6)
Bibtex	0.0160 (4)	0.0846 (6)	0.0135 (2)	1.0000 (9.5)	0.0149 (3)	0.9853 (8)	0.0165 (5)	1.0000 (9.5)	0.0126 (1)	0.8492 (7)
Average rank	4.1 (3)	4.5 (4.5)	1.9 (2)	9.6 (10)	4.6 (6)	7.95 (8)	4.5 (4.5)	9.4 (9)	**1**.7 (**1**)	6.75 (7)

TABLE 14
Performance comparison of 10 algorithms on 10 real-world datasets of [41] without ground-true label distributions using ranking loss ↓

Algorithms	BP-MLL	MLNB	ML-kNN	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	PLEA	PLEA$^-$
Yeast	0.4450 (5)	0.2323 (2)	0.1716 (1)	0.5915 (9)	0.4974 (7)	0.9708 (10)	0.5054 (8)	0.4809 (6)	0.2904 (3)	0.4011 (4)
Emotions	0.4803 (7)	0.2285 (2)	0.2827 (3)	0.9453 (10)	0.5899 (8)	0.8511 (9)	0.4283 (6)	0.3877 (5)	0.2166 (1)	0.3670 (4)
Medical	0.2445 (5)	0.0623 (2)	0.0555 (1)	0.8245 (9)	0.5000 (7)	0.8982 (10)	0.5039 (8)	0.3082 (6)	0.1093 (3)	0.1157 (4)
Cal500	0.1996 (3)	0.1882 (2)	0.1880 (1)	0.5126 (8)	0.5005 (7)	0.8621 (10)	0.7750 (9)	0.4937 (6)	0.4749 (4)	0.4903 (5)
Birds	0.3964 (6)	0.2115 (1)	0.3035 (3)	0.6485 (9)	0.4374 (8)	0.3132 (2)	0.7335 (10)	0.4157 (7)	0.3865 (5)	0.3519 (4)
Image	0.7956 (9)	0.2231 (3)	0.2008 (2)	0.7320 (8)	0.5000 (7)	0.8892 (10)	0.3139 (4)	0.3819 (6)	0.1456 (1)	0.3206 (5)
Scene	0.5992 (6)	0.1070 (4)	0.1059 (3)	0.7328 (9)	0.6556 (7)	0.8609 (10)	0.1838 (5)	0.6753 (8)	0.0575 (1)	0.0592 (2)
Enron	0.3738 (4)	0.1776 (2)	0.1201 (1)	0.6236 (8)	0.4741 (6)	0.9621 (10)	0.8563 (9)	0.5165 (7)	0.3229 (3)	0.4146 (5)
Corel5k	0.2695 (2)	0.4145 (3)	0.2672 (1)	0.5134 (9)	0.5000 (8)	0.4990 (7)	0.9444 (10)	0.4954 (6)	0.4439 (4)	0.4696 (5)
Bibtex	0.4764 (6)	0.2037 (4)	0.2427 (5)	0.5243 (8)	0.5012 (7)	0.6954 (9)	0.7416 (10)	0.0000 (1)	0.0897 (2)	0.0960 (3)
Average rank	5.3 (5)	2.5 (2)	**2.1** (**1**)	8.7 (9.5)	7.2 (7)	8.7 (9.5)	7.9 (8)	5.8 (6)	2.7 (3)	4.1 (4)

TABLE 15
Performance comparison of 10 algorithms on 10 real-world datasets of [41] without ground-true label distributions using one error ↓

Algorithms	BP-MLL	MLNB	ML-kNN	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	PLEA	PLEA$^-$
Yeast	0.7034 (8)	0.2475 (4)	0.2454 (3)	0.7857 (9.5)	0.4286 (5)	0.0714 (1)	0.4999 (6)	0.7857 (9.5)	0.1429 (2)	0.6429 (7)
Emotions	0.7022 (10)	0.4100 (4)	0.4213 (5)	0.0000 (1)	0.6667 (8.5)	0.3333 (2.5)	0.4899 (6)	0.3333 (2.5)	0.5000 (7)	0.6667 (8.5)
Medical	0.4024 (4)	0.4324 (6)	0.2583 (2)	1.0000 (10)	0.5000 (8)	0.4290 (5)	0.1579 (1)	0.5789 (9)	0.3421 (3)	0.4737 (7)
Cal500	0.1071 (2.5)	0.1111 (4)	0.1071 (2.5)	0.8678 (9)	0.8563 (8)	0.3333 (5)	0.5862 (6)	0.8046 (7)	0.0776 (1)	0.8793 (10)
Birds	0.7989 (6)	0.5287 (4)	0.7126 (5)	0.8421 (8)	0.4990 (3)	0.8421 (8)	0.4737 (2)	0.9474 (10)	0.4474 (1)	0.8421 (8)
Image	0.6710 (9)	0.4030 (4)	0.3630 (3)	1.0000 (10)	0.5000 (6)	0.5470 (7)	0.4990 (5)	0.2000 (2)	0.0000 (1)	0.6000 (8)
Scene	0.8269 (8)	0.3002 (4)	0.2575 (3)	1.0000 (9.5)	0.4999 (6)	0.3333 (5)	0.5000 (7)	1.0000 (9.5)	0.0000 (1.5)	0.0000 (1.5)
Enron	0.2642 (1)	0.5009 (4)	0.4076 (2)	0.9804 (10)	0.9615 (9)	0.5050 (5)	0.4808 (3)	0.9423 (8)	0.6923 (6)	0.8846 (7)
Corel5k	0.9716 (7)	0.8868 (5)	0.7856 (4)	0.9865 (10)	0.4890 (3)	0.4400 (1)	0.4419 (2)	0.9797 (9)	0.9302 (6)	0.9767 (8)
Bibtex	0.4547 (3)	0.5681 (4)	0.6363 (5)	0.9937 (10)	0.9497 (8)	0.9874 (9)	0.7688 (6)	0.8428 (7)	0.3459 (1)	0.3899 (2)
Average rank	5.85 (6)	4.3 (3)	3.45 (2)	8.7 (10)	6.45 (7)	4.85 (5)	4.4 (4)	7.35 (9)	**2.95** (**1**)	6.7 (8)

TABLE 16
Performance comparison of 10 algorithms on 10 real-world datasets of [41] without ground-true label distributions using coverage ↓

Algorithms	BP-MLL	MLNB	ML-kNN	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	PLEA	PLEA$^-$
Yeast	0.8990 (9)	0.6629 (2)	0.6385 (1)	1.4925 (10)	0.8982 (8)	0.8845 (5)	0.8794 (3)	0.8976 (7)	0.8816 (4)	0.8868 (6)
Emotions	0.3089 (9)	0.1960 (7)	0.2320 (8)	0.4125 (10)	0.1568 (1)	0.1703 (6)	0.1661 (4)	0.1643 (3)	0.1570 (2)	0.1693 (5)
Medical	0.2955 (8)	0.1934 (5)	0.3564 (9)	0.5036 (10)	0.2087 (7)	0.2081 (6)	0.1374 (3)	0.1562 (4)	0.0520 (1)	0.0538 (2)
Cal500	1.3386 (10)	0.1346 (1)	1.3045 (9)	0.2332 (8)	0.2284 (4)	0.2316 (7)	0.2315 (6)	0.2286 (5)	0.2281 (2)	0.2282 (3)
Birds	0.4415 (10)	0.2695 (3)	0.3606 (9)	0.2702 (4)	0.3014 (8)	0.2309 (1)	0.2899 (7)	0.2771 (5)	0.2828 (6)	0.2646 (2)
Image	2.1460 (10)	1.1700 (9)	1.0760 (8)	0.9980 (7)	0.9608 (2)	0.9648 (4)	0.9644 (3)	0.9872 (6)	0.9512 (1)	0.9868 (5)
Scene	2.0761 (10)	0.6296 (1)	0.6405 (2)	1.2093 (9)	1.0843 (7)	1.0773 (6)	1.0505 (5)	1.1597 (8)	0.9330 (3)	0.9685 (4)
Enron	0.2369 (2)	0.2313 (1)	1.6046 (10)	1.0148 (9)	0.4936 (6)	0.5028 (8)	0.4956 (7)	0.4891 (5)	0.4579 (3)	0.4792 (4)
Corel5k	0.1980 (4)	0.2062 (6)	0.1983 (5)	1.6825 (10)	1.5023 (7.5)	1.5023 (7.5)	1.5126 (9)	0.1876 (3)	0.1806 (1)	0.1813 (2)
Bibtex	0.7356 (10)	0.3788 (8)	0.6146 (9)	0.3562 (5)	0.3382 (4)	0.3598 (7)	0.3585 (6)	0.2068 (1)	0.2457 (2)	0.2486 (3)
Average rank	8.2 (9.5)	4.3 (3)	7.0 (8)	8.2 (9.5)	5.45 (6)	5.75 (7)	5.3 (5)	4.7 (4)	**2.5** (**1**)	3.6 (2)

TABLE 17
Performance comparison of 10 algorithms on 10 real-world datasets of [41] without ground-true label distributions using average precision ↑

Algorithms	BP-MLL	MLNB	ML-kNN	AA-BP	LDSVR	CPNN	AA-kNN	IIS-LLD	PLEA	PLEA$^-$
Yeast	0.4297 (5)	0.7481 (2)	0.7566 (1)	0.2675 (10)	0.3965 (6)	0.3064 (9)	0.4779 (4)	0.3125 (8)	0.5085 (3)	0.3876 (7)
Emotions	0.5161 (4)	0.7324 (1)	0.6897 (2)	0.3422 (9)	0.4900 (6)	0.3123 (10)	0.4926 (5)	0.4220 (8)	0.6217 (3)	0.4502 (7)
Medical	0.2081 (6)	0.6080 (2)	0.7898 (1)	0.0186 (10)	0.0480 (8)	0.0467 (9)	0.3692 (5)	0.2035 (7)	0.5783 (3)	0.5315 (4)
Cal500	0.4783 (2)	0.4372 (3)	0.4882 (1)	0.1655 (9)	0.1676 (8)	0.1598 (10)	0.1705 (5)	0.1687 (6)	0.1815 (4)	0.1686 (7)
Birds	0.2460 (3)	0.5423 (1)	0.3875 (2)	0.0653 (10)	0.0759 (9)	0.1013 (8)	0.1131 (7)	0.1151 (6)	0.1255 (5)	0.1382 (4)
Image	0.5111 (5)	0.7386 (2)	0.7649 (1)	0.1650 (10)	0.2729 (8)	0.2645 (9)	0.5954 (4)	0.3663 (7)	0.7073 (3)	0.4176 (6)
Scene	0.4200 (7)	0.8191 (4)	0.8378 (2)	0.1247 (10)	0.7859 (5)	0.2954 (8)	0.7649 (6)	0.1615 (9)	0.8407 (1)	0.8353 (3)
Enron	0.2057 (3)	0.2135 (2)	0.5509 (1)	0.0522 (10)	0.0747 (9)	0.0828 (7)	0.1201 (6)	0.0812 (8)	0.1890 (4)	0.1207 (5)
Corel5k	0.2012 (2)	0.2200 (1)	0.1929 (3)	0.0140 (9)	0.0141 (7.5)	0.0141 (7.5)	0.0252 (5)	0.0137 (10)	0.0339 (4)	0.0193 (6)
Bibtex	0.0659 (6)	0.3874 (1)	0.3057 (4)	0.0155 (9)	0.0226 (7)	0.0182 (8)	0.1111 (5)	NaN (10)	0.3871 (2)	0.3768 (3)
Average rank	4.3 (4)	1.9 (2)	**1.8** (**1**)	9.6 (10)	7.35 (7)	8.55 (9)	5.2 (5.5)	7.9 (8)	3.2 (3)	5.2 (5.5)

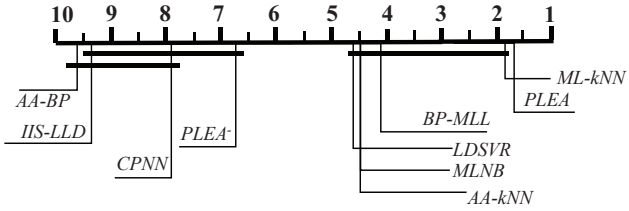
TABLE 18
Multi-label classification ranking performance of 10 algorithms averaged over 10 datasets of [41] and 5 MLL measures

Algorithm	Average rank
PLEA	**2.61** (**1**)
ML-kNN	3.25 (2)
MLNB	3.5 (3)
PLEA$^-$	5.27 (4)
AA-kNN	5.46 (5)
BP-MLL	5.55 (6)
LDSVR	6.21 (7)
IIS-LLD	7.03 (8)
CPNN	7.16 (9)
AA-BP	8.96 (10)

indicate that our PLEA on average achieves the best multi-label classification performance is statistically significant.

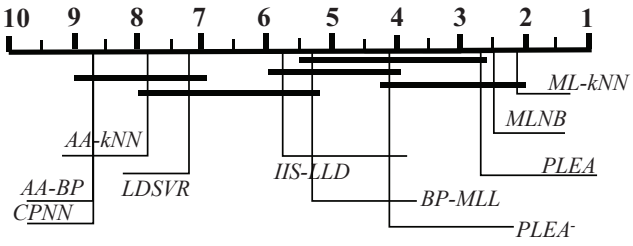
TABLE 19
Friedman statistics F_F, in terms of each evaluation metric and the critical value at a significance level of 0.05 (comparing algorithms 10, datasets 10)

Evaluation metric	F_F	Critical value
Hamming loss	60.0981	
ranking loss	23.4512	
one error	5.1172	1.998
coverage	5.5945	
average precision	45.4132	



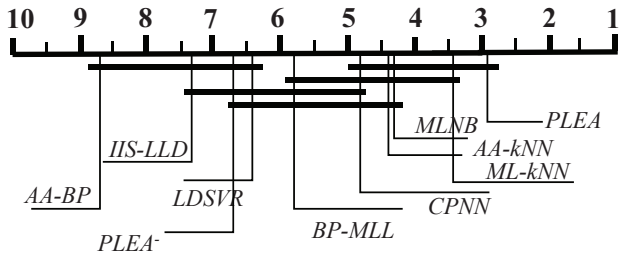
(a) Hamming-loss

Fig. 8. CD diagrams given CD $= 2.7053$ of Nemenyi tests on the 10 algorithms for Hamming loss evaluation metric



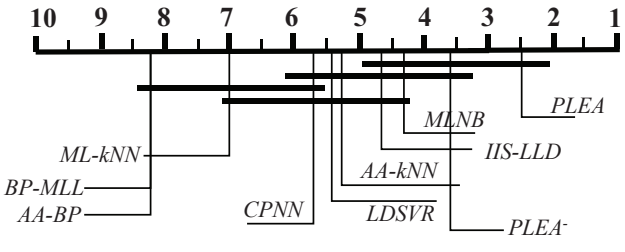
(b) ranking loss

Fig. 9. CD diagrams given CD $= 2.7053$ of Nemenyi tests on the 10 algorithms for ranking loss evaluation metric



(c) one error

Fig. 10. CD diagrams given CD $= 2.7053$ of Nemenyi tests on the 10 algorithms for one error evaluation metric

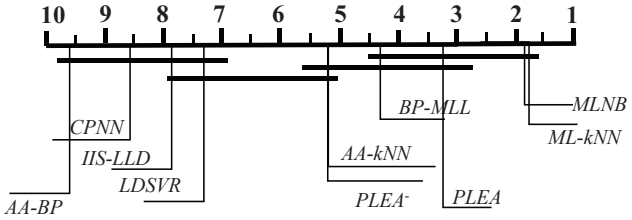


(d) coverage

Fig. 11. CD diagrams given CD $= 2.7053$ of Nemenyi tests on the 10 algorithms for coverage evaluation metric

4.5 Summary

Combined with the experimental results of the previous three subsections, we can confidently draw the conclusion that the proposed PLEA algorithm offers considerable advantages over the existing well-established LDL algorithms as well as the state-of-the-art MLL algorithms, in terms of both LDL accuracy and MLL performance. Specifically, our PLEA consistently outperforms the existing LDL and MLL algorithms in the multi-label distribution learning task, and it is also capable of offering excellent performance for the mulit-label classification learning task. Furthermore, our PLEA algorithm consistently achieves better runtime performance than the IIS-LLD, which is the existing state-of-the-art LDL algorithm.



(e) average precision

Fig. 12. CD diagrams given CD $= 2.7053$ of Nemenyi tests on the 10 algorithms for average precision evaluation metric

5 Conclusions

In this paper, we have proposed a new probabilistic label enhancement algorithm for the challenging multi-label label distribution learning problem. Our novel contribution has been twofold. More specifically, we have proposed a manifold space learning based feature extraction and a robust linear regression to derive the reduced-dimensional principal features and their corresponding estimated label distributions. This has enabled us to estimate the unknown true label distributions based on the enhanced maximum entropy model with improved estimation accuracy and reduced computational complexity. Extensive experimental results have confirmed that compared with the latest existing multi-label LDL algorithms, our proposed PLEA algorithm offers clear advantages, in terms of both label distribution estimation accuracy and computational complexity. Furthermore, our PLEA is also capable of offering excellent performance on the multi-label classification learning problem.

Appendix

We use five real-world multi-label vehicle video datasets with unknown ground-true label distributions of [42], denoted as BRVD1 to BRVD5, for evaluating various LDL algorithms. We now describe how the training dataset $\{\boldsymbol{x}_i \in \mathbb{R}^q, \boldsymbol{y}_i = [y_i^1 \cdots y_i^c]^{\mathrm{T}} \in \{-1, 1\}^c\}_{i=1}^n$ is constructed for each BRVD from the raw dataset of [42].

The BRVD datasets of [42] are collected for the purpose of training autonomous driving system. Each raw BRVD dataset contains a large number of short vehicular videos. We basically 'sample' videos to obtain training examples. The video collected by the driving recorder is a color video, which consists of three components, R (red), G (green) and B (blue). Directly 'sampling' colored videos will lead to examples with huge feature dimension, which requires huge memory space to store the data and imposes unacceptably high computation time for training the system. Since the color information is not needed in training the system, we first convert color video into a grayscale video using the weighted average method [26]. Specifically, the grayscale image $f(x_{\mathrm{h}}, x_{\mathrm{v}})$ is obtained from the R, G and B images, $R(x_{\mathrm{h}}, x_{\mathrm{v}})$, $G(x_{\mathrm{h}}, x_{\mathrm{v}})$ and $B(x_{\mathrm{h}}, x_{\mathrm{v}})$, according to

$$\begin{aligned} f(x_{\mathrm{h}}, x_{\mathrm{v}}) =& w_R \cdot R(x_{\mathrm{h}}, x_{\mathrm{v}}) + w_G \cdot G(x_{\mathrm{h}}, x_{\mathrm{v}}) \\ &+ w_B \cdot B(x_{\mathrm{h}}, x_{\mathrm{v}}). \end{aligned} \tag{20}$$

Since the sensitivity of human eye to blue color is relatively low and the sensitivity to green color is high, appropriate weighting values are chosen to be $w_R = 0.3$, $w_G = 0.59$, $w_R = 0.11$. After this preprocessing, for each raw BRVD

dataset, we select a number of short videos and we divide each short video into 400 segments. The number of short videos selected multiplying by 400 yields the number of examples n. Each video segment forms a raw example $\boldsymbol{x}_i^{\mathrm{raw}} \in \mathbb{R}^{q^{\mathrm{raw}}}$.

For each example, we classify it by $c = 9$ labels. These include:

Three driving-scene labels:	Highway, City, Country.
Two driving-time labels:	Day, Night.
Two labels for weather:	Sunny, Rain and snow.
One label for pedestrian:	Present or not.
One label for lane line:	Present or not.

Fig. 13. An example of driving video picture, i.e., a raw example $\boldsymbol{x}_i^{q^{\mathrm{raw}}}$.

The raw feature dimension q^{raw} is far too large and will impose unacceptably high computation time. We perform feature reduction manually by selecting q ($\ll q^{\mathrm{raw}}$) important features from $\boldsymbol{x}_i^{\mathrm{raw}}$ to form the reduced-dimension example $\boldsymbol{x}_i \in \mathbb{R}^q$. More specifically, since the original purpose of collecting these data is for training autonomous driving system, we only retain those features of $\boldsymbol{x}_i^{q^{\mathrm{raw}}}$ that are relevant for adding autonomous driving, and remove the features that do not add the driving system. A typical example of this manual feature reduction is illustrated in Fig. 13, where a typical raw example $\boldsymbol{x}_i^{q^{\mathrm{raw}}}$, i.e., a typical driving video picture is depicted. It is clear that the top sky part of the picture is irrelevant to the driving system, and hence can be removed. Similarly, the far left portion of the picture can also be removed. In other words, the manual feature reduction only keeps the important part of a picture, that is, only retains the relevant subset features of a raw example $\boldsymbol{x}_i^{q^{\mathrm{raw}}}$.

After performing the aforementioned preprocessing on the raw datasets of [42], we construct the five multi-label BRVD training datasets, whose numbers of labels c, numbers of examples n and feature dimensions q are listed in Table 10.

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] T. N. Rubin, A. Chambers, P. Smyth, and M. Steyvers, "Statistical topic models for multi-label document classification," *Machine Learning*, vol. 88, nos. 1-2, pp. 157-208, Jul. 2012.

[3] H.-Y. Lo, J.-C. Wang, H.-M. Wang, and S.-D. Lin, "Cost-sensitive multi-label learning for audio tag annotation and retrieval," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 518–529, Jun. 2011.

[4] R. S. Cabral, F. Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for multi label image classification," In *Proc. NIPS 2011* (Granada, Spain), Dec. 12-15, 2011, pp. 190–198.

[5] J.-D. Wang, Y.-H. Zhao, X.-Q. Wu, X.-S. Hua, "A transductive multi-label learning approach for video concept detection," *Pattern Recognition*, vol. 44, nos. 10-11, pp. 2274-2286, 2011.

[6] 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.

[7] Q. Zhao, J. Dong, H. Yu, and S. Chen, "Distilling ordinal relation and dark knowledge for facial age estimation," *IEEE Trans. Neural Networks and Learning Systems*, 10.1109/TNNLS.2020.3009523, 2020.

[8] B. Du, *et al.*, "Robust and discriminative labeling for multi-label active learning based on maximum correntropy criterion," *IEEE Trans. Image Processing*, vol. 26, no. 4, pp. 1694–1707, Apr. 2017.

[9] B. Du, *et al.*, "Robust graph-based semisupervised learning for noisy labeled data via maximum correntropy criterion," *IEEE Trans. Cybernetics*, vol. 49, no. 4, pp. 1440-1453, Apr. 2018.

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

[11] N. Xu, A. Tao, and X. Geng, "Label enhancement for label distribution learning," In *Proc. IJCAI 2018* (Stockholm, Sweden), Jul. 13-19, 2018, pp. 2926–2932.

[12] 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. 3742–3747.

[13] X. Geng, Q. Wang, and Y. Xia, "Facial age estimation by adaptive label distribution learning," In *Proc. ICPR 2014* (Stockholm, Sweden), Aug. 24-28, 2014, pp. 4465–4470.

[14] Y.-K. Li, M.-L. Zhang, and X. Geng, "Leveraging implicit relative labeling-importance information for effective multi-label learning," In *Proc. ICDM 2015* (Atlantic City, NJ, USA), Nov. 14-17, 2015, pp. 251–260.

[15] 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.

[16] 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.

[17] J. Nocedal and S. Wright, *Numerical Optimization* (2nd ed.). New York, USA: Springer, 2006.

[18] 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, Jul. 2006.

[19] 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. 1338–1351, Oct. 2006.

[20] J.-Y. Jiang, S.-C. Tsai, and S.-J. Lee, "FSKNN multi-label text categorization based on fuzzy similarity and k nearest neighbors," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2813–2821, Feb. 2012.

[21] Y. Yu, W. Pedrycz, and D. Miao, "Neighborhood rough sets based multi-label classification for automatic image annotation," *Int. J. Approximate Reasoning*, vol. 54, no. 9, pp. 1373–1387, Nov. 2013.

[22] S.-M. Liu and J.-H. Chen, "A multi-label classification based approach for sentiment classification," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1083–1093, Feb. 2015.
[23] M.-X. Ding, Y.-L. Yang, and Z.-Q. Lan, "Multi-label imbalanced classification based on assessments of cost and value," *Applied Intelligence*, vol. 48, no. 10, pp. 3577–3590, Oct. 2018.
[24] H.-R. Han, *et al.*, "Multi-label learning with label specific features using correlation information," *IEEE Access*, vol. 7, pp. 11474–11484, Jan. 2019.
[25] W.-S. Pan, L.-W. Jin, and Z.-Y. Feng, "Recognition of Chinese characters based on multi-scale gradient and deep neural network," *J. Beijing University of Aeronautics and Astronautics*, vol. 41, no. 4, pp. 751–756, Apr. 2015.
[26] J. Zhang, X.-B. Mao, and T.-J. Chen, "Survey of moving object tracking algorithm," *Application Research of Computers*, vol. 26, no. 12, pp. 4407–4410, Dec. 2009.
[27] W. Shen, K. Zhao, Y.-L. Guo, and A.-L. Yuille, "Label distribution learning forests," In *Proc. NIPS 2017* (Long Beach, CA, USA), Dec 4-9, 2017, pp. 834–843.
[28] B.-B. Gao, *et al.*, "Deep label distribution learning with label ambiguity," *IEEE Trans. Image Processing*, vol. 26, no. 6, pp. 2825–2838, Jun. 2017.
[29] Q. Wang, Z. Qin, F. Nie, and X. Li, "C2DNDA: A deep framework for nonlinear dimensionality reduction," *IEEE Trans. Industrial Electronics*, DOI: 10.1109/TIE.2020.2969072, 2020.
[30] Q. Wang, *et at.*, "Hierarchical feature selection for random projection," *IEEE Trans. Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1581–1586, May 2019.
[31] M.-H.-C. Law, and A.-K. Jain, "Incremental nonlinear dimensionality reduction by manifold learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 377–391, Mar. 2006.
[32] P. Jia, J.-S. Yin, X.-S. Huang, and D. Hu, "Incremental Laplacian eigenmaps by preserving adjacent information between data points," *Pattern Recognition Letters*, vol. 30, no. 16, pp. 1457–1463, Dec. 2009.
[33] O. Kouropteva, O. Okun, and M. Pietikäinen, "Incremental locally linear embedding," *Pattern Recognition*, vol. 38, no. 10, pp. 1764–1767, Oct. 2005.
[34] O. Abdel-Mannan, A.-B. Hamza, and A. Youssef, "Incremental line tangent space alignment algorithm," In *Proc. 2007 Canadian Conf. Electrical and Computer Engineering* (Vancouver, BC, Canada), Apr. 22-26, 2007, pp. 1329–1332.
[35] J.-Y. Weng, Y.-L. Zhang, and W.-S. Hwang, "Candid covariance-free incremental principal component analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 1034–1040, Aug. 2003.
[36] C. Tan, G.-L. Ji, and B. Zhao, "Self-adaptive streaming big data learning algorithm based on incremental tangent space alignment," *Journal of Computer Research and Development*, vol. 54, no. 11, pp. 2547–2557, Nov. 2017.
[37] S.-D. Pietra, V.-J.-D. Pietra, and J.-D. Lafferty, "Inducing features of random fields," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 380–393, Apr. 1997.
[38] P. Hou, X. Geng, and M. Zhang, "Multi-label manifold learning," In *Proc. AAAI-16* (Phoenix, AZ), Feb. 12-17, 2016, pp. 1680–1686.
[39] 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.
[40] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.
[41] Mulan: A java library for multi-label learning. http://mulan.sourceforge.net/datasets-mlc.html, 2018-03-01.
[42] BDD100K: A Large-scale Diverse Driving Video Database. https://bair.berkeley.edu/blog/2018/05/30/bdd/, 06/18/2018.
[43] M.-L. Zhang, J. M. Pena, and V. Robles, "Feature selection for multi-label naive Bayes classification," *Information Sciences*, vol. 179, no. 19, pp. 3218-3229, 2009.
[44] 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.
[45] J. Demšar, "Statistical comparisons of classifiers over multiple datasets," *J. Machine Learning Research*, vol. 7, no. 1, pp. 1-30, 2006.

Chao Tan received the B.E. and M.E. degree in Computer Science and Technology from Southeast University in 2005 and 2009, respectively, and received the PhD degree in Computer Science and Technology from Tongji University in 2015. She joined the Nanjing Normal University as a lecturer in 2015 and is an associate professor at present. She is now a postdoctoral researcher in Southeast University. Her research interests generally focus on machine learning, multi-label manifold learning and data mining.

Sheng Chen (Fellow, IEEE) received his BEng degree from the East China Petroleum Institute, Dongying, China, in 1982, and his PhD degree from the City University, London, in 1986, both in control engineering. In 2005, he was awarded the higher doctoral degree, Doctor of Sciences (DSc), from the University of Southampton, Southampton, UK. From 1986 to 1999, He held research and academic appointments at the Universities of Sheffield, Edinburgh and Portsmouth, all in UK. Since 1999, he has been with the School of Electronics and Computer Science, the University of Southampton, UK, where he holds the post of Professor in Intelligent Systems and Signal Processing. Dr Chen's research interests include adaptive signal processing, wireless communications, modelling and identification of nonlinear systems, neural network and machine learning, intelligent control system design, evolutionary computation methods and optimisation. He has published over 650 research papers. Professor Chen has 15,600+ Web of Science citations with h-index 55 and 31,500+ Google Scholar citations with h-index 77. Dr. Chen is a Fellow of the United Kingdom Royal Academy of Engineering, a Fellow of IET, a Distinguished Adjunct Professor at King Abdulaziz University, Jeddah, Saudi Arabia, and an original ISI highly cited researcher in engineering (March 2004).

Genlin Ji received the B.E. and M.E. degree in Computer Science and Technology from Nanjing University of Aeronautics and Astronautics in 1986 and 1989, respectively, and received the PhD degree in Computer Science and Technology from Southeast University in 2004. He is now a professor and dean of the school of Computer Science and Technology at Nanjing Normal University. His research interests generally focus on data mining and its application.

Xin Geng (Member, IEEE) received the BSc and MSc degrees in Computer Science from Nanjing University, China, in 2001 and 2004, respectively, and the PhD degree from Deakin University, Australia in 2008. He is currently a professor and dean of the school of Computer Science and Engineering at Southeast University. His research interests include pattern recognition, machine learning, and computer vision. He has published more than 40 refereed papers and holds four patents in these areas. He is member of the IEEE.