Label Distribution Learning

Xin Geng*, and Quan Zhao

Abstract—Although multi-label learning can deal with many problems with label ambiguity, it does not fit some real applications well where the overall distribution of the importance of the labels matters. This paper proposes a novel learning paradigm named label distribution learning (LDL) for such kind of applications. The label distribution covers a certain number of labels, representing the degree to which each label describes the instance. LDL is a more general learning framework which includes both single-label and multi-label learning as its special cases. This paper proposes six working LDL algorithms in three ways: problem transformation, algorithm adaptation, and specialized algorithm design. In order to compare their performance, six evaluation measures are suggested for LDL algorithms, and the first batch of label distribution datasets are collected and made publicly available. Experimental results on one artificial and two real-world datasets show clear advantage of the specialized algorithms, which indicates the importance of special design for the characteristics of the LDL problem.

Index Terms—Multi-label learning, label distribution learning, learning with ambiguity

1 INTRODUCTION

Learning with ambiguity is a hot topic in recent machine learning and data mining research. A learning process is essentially building a mapping from the instances to the labels. Ambiguity at the label side of the mapping is embodied by the way the labels are assigned to the instances. There are mainly two ways to label an instance in existing learning paradigms: (1) a single label is assigned to this instance, and (2) multiple labels are assigned to this instance. Single-label learning assumes that all the instances in the training set are labeled in the first way. Multi-label learning (MLL) [31] allows the training instances to be labeled in the second way, and thus can deal with the ambiguity at the label side. Generally speaking, current MLL algorithms have been developed with two strategies [30]. The first strategy is problem transformation, where the basic idea is to transform the MLL task into one or more single-label learning tasks. For example, the MLL problem could be transformed into binary classification problems [14, 25], a label ranking problem [15, 9], or an ensemble learning problem [20, 17]. The second strategy is algorithm adaptation, where the basic idea is to extend specific learning algorithms to handle multi-label data. For example, it can be extended from k-NN [37, 5], decision tree [6, 24], or neural networks [36, 27].

MLL has been successfully applied to many real-world problems because it can well model the ambiguity of “what describes the instance”, i.e., multiple labels can be used to describe the same instance. However, MLL can hardly deal with the more general ambiguity of “how to describe the instance”, i.e., the exact role of each label is also involved in the description of the instance. Surprisingly, the real-world data with the information about such exact role of each label might be more common than many people think. For example, in many scientific experiments [8], the result is not a single output, but a series of numerical outputs (e.g., gene expression levels over a period of time). Each of these outputs alone might be of little importance. What really matters is the overall distribution of the whole series of outputs. If the learning task is to predict such distributions, then it can hardly fit into the MLL framework because the exact role of each output in the distribution is crucial, and there is no partition of relevant and irrelevant labels at all. Another example is from the research on the relationship between protein and cancer. It was discovered [34] that one particular kind of protein could be related to several cancers, and the expression levels of the protein in these related cancer cells are generally different. Higher expression level indicates closer relationship, i.e., the corresponding cancer plays a more important role when describing the protein. By regarding the cancers with the protein expression level higher than a threshold as the positive labels of the protein, the problem can fit into the MLL framework. Unfortunately, this will lose the important information of the different protein expression levels in the related cancers. Yet another example is from the overlapping community analysis of complex networks [20]. One node in a complex network might belong to multiple communities. But the importance of each community to this node is usually different. Such importance can be represented by certain criterion, such as the z-score [13], which indicates the role of the corresponding community when describing the node. Similarly, a threshold of the z-score can be used to determine the positive labels of a node, and then MLL can be applied to this problem. But this will lose the valuable information about the different importance of the overlapping communities to the node.

For the applications mentioned above, a more natural way to label an instance \( x \) is to assign a real number \( d_y^x \) to each possible label \( y \), representing the degree to which \( y \) describes \( x \). For example, if \( x \) represents a protein, \( y \) represents a cancer, then \( d_y^x \) should be the expression level of the protein \( x \) in the
cancer \(y\). Without loss of generality, assume that \(d_y^x \in [0, 1]\). Further suppose that the label set is complete, i.e., using all the labels in the set can always fully describe the instance. Then, \(\sum_y d_y^x = 1\). Such \(d_y^x\) is called the description degree of \(y\) to \(x\). For a particular instance, the description degrees of all the labels constitute a data form similar to probability distribution. So, it is called label distribution. The learning process on the instances labeled by label distributions is therefore called label distribution learning (LDL). The aforementioned applications reveal that the nature of label ambiguity might exceed the current framework of MLL. In such case, it is necessary to extend MLL to LDL. In fact, the real applications suitable for LDL might be more common than those suitable for MLL because when there are multiple labels associated with one instance, their importance or relevance to the instance can hardly be exactly the same.

It is not a rare case in existing single-label or multi-label machine learning literatures that a numerical indicator (e.g., probability, confidence, grade, score, votes, etc.) is associated with each label \([33], [37], [15], [4]\). As shown in Fig. 1 label distribution learning is different from these learning methods in mainly three aspects:

1) Each training instance of LDL is explicitly associated with a label distribution, rather than a single label or a relevant (positive) label set. The label distribution comes from the application itself as a natural part of the original data, while most numerical label indicators used in previous learning algorithms are artificially generated from the original data for later decision making.

2) The purpose of most kinds of numerical label indicators used in previous learning algorithms is to rank the labels, and then decide the positive label(s) through, say, thresholding over the ranking. The actual value of the indicator does not matter much so long as the ranking does not change. In most cases, it only cares about the first or first several labels in the ranking. On the other hand, what LDL cares about is the overall label distribution. The exact value of each label’s description degree is thus important.

3) The performance evaluation measures of previous learning algorithms with numerical label indicators are still those commonly used for single-label learning (e.g., classification accuracy, error rate, etc.) or multi-label learning (e.g., Hamming loss, one-error, coverage, ranking loss, etc.). On the other hand, the performance of LDL should be evaluated by the similarity or distance between the predicted label distribution and the real label distribution, which will be further discussed in Section 3.4.

This paper extends our preliminary work [10]. The main contribution of this paper includes: 1) A novel machine learning paradigm named label distribution learning is formulated; 2) Six working label distribution learning algorithms are proposed and compared in the experiments; 3) Six measures
Fig. 2. Three cases of label distribution.

for evaluation of label distribution learning algorithms are suggested; 4) The first batch of real-world label distribution datasets are prepared and made publicly available to the research community. The rest of the paper is organized as follows. Firstly, the problem of LDL is formulated in Section 1. Then, six LDL algorithms are presented in Section 3. After that, the experiments on artificial as well as real-world datasets are reported in Section 4. Finally, the paper is summarized and some discussions on future work are given in Section 5.

2 FORMULATION OF LDL

By the definition of label distribution given in Section 1, both single-label and multi-label annotations can be viewed as special cases of label distribution. Fig. 2 gives one label distribution example for the single-label, multi-label, and general cases, respectively. For the single-label case (a) the label $y_2$ fully describes the instance, so $d_{x|x}^y = 1$. For the multi-label case (b) each of the two positive labels $y_2$ and $y_4$ by default describes 50% of the instance, so $d_{x|x}^y = d_{x|x}^{y_4} = 0.5$. Finally, (c) represents a general case of label distribution, which satisfies the constraints $d_{x|x}^y \in [0,1]$ and $\sum_y d_{x|x}^y = 1$. Such examples illustrate that label distribution is more general than both the single-label and multi-label cases, and thus can provide more flexibility in the learning process.

Nevertheless, more flexibility usually means larger output space. From single label to multi-label, and then to label distribution, the size of the output space of the learning process becomes increasingly larger. In detail, for a problem with $c$ different labels, there are $c$ possible outputs for a single-label learner, and $2^c - 1$ possible outputs for a multi-label learner. As for a label distribution learner, there are infinite possible outputs as long as they satisfy that $d_{x|x}^y \in [0,1]$ and $\sum_y d_{x|x}^y = 1$.

Fig. 3 gives three cases of the decision regions for a typical two-label problem. Note also that $d_{x|x}^y$ is not the probability that $y$ correctly labels $x$, but the proportion that $y$ accounts for in a full description of $x$. Thus, all the labels with a non-zero description degree are actually the ‘correct’ labels to describe the instance, but just with different importance measured by $d_{x|x}^y$. Recognizing this, one can distinguish label distribution from the previous studies on probabilistic labels [28], [7], [22], where the basic assumption is that there is only one ‘correct’ label for each instance. Fortunately, although not a probability by definition, $d_{x|x}^y$ still shares the same constraints with probability, i.e., $d_{x|x}^y \in [0,1]$ and $\sum_y d_{x|x}^y = 1$. Thus, many theories and methods in statistics can be applied to label distributions.

First of all, $d_{x|x}^y$ can be represented by the form of conditional probability, i.e., $d_{x|x}^y = P(y|x)$. Then, the problem of LDL can be formulated as follows.

Let $X = \mathbb{R}^q$ denote the input space and $Y = \{0,1\}^c$. Membership is a truth value that may range between completely true and completely false. It is designed to handle the status of partial truth which often appears in the non-numeric linguistic variables. On the other hand, description degree reflects the ambiguity of the label description of the instance, i.e., one label may only partially describe the instance, but it is completely true that the label describes the instance. Based on fuzzy set theory, a recent extension of multi-label classification, namely graded multilabel classification [4], allows for graded membership of an instance belonging to a class. Apart from the different meanings of continuous description degree and graded membership, LDL is also different from this work in methodology. The strategy in [4] is to reduce graded multi-label problems to conventional multi-label problems, while LDL aims to directly model the mapping from instances to label distributions.
\{y_1, y_2, \cdots, y_c\} denote the complete set of labels. Given a training set \(S = \{(x_1, D_1), (x_2, D_2), \cdots, (x_n, D_n)\}\), where \(x_i \in \mathcal{X}\) is an instance and \(D_i = \{d_{x_i}^1, d_{x_i}^2, \cdots, d_{x_i}^{y_i}\}\) is the label distribution associated with \(x_i\), the goal of label distribution learning is to learn a conditional probability mass function \(p(y|x)\) from \(S\), where \(x \in \mathcal{X}\) and \(y \in \mathcal{Y}\).

Suppose \(p(y|x)\) is a parametric model \(p(y|x; \theta)\), where \(\theta\) is the parameter vector. Given the training set \(S\), the goal of LDL is to find the \(\theta\) that can generate a distribution similar to \(D_i\) given the instance \(x_i\). As will be discussed in Section 3.3, there are different criteria that can be used to measure the distance or similarity between two distributions. For example, if the Kullback-Leibler divergence is used as the distance measure, then the best parameter vector \(\theta^*\) is determined by

\[
\theta^* = \arg\min_\theta \sum_i \sum_j \left( d_{x_i}^{y_j} \ln \frac{d_{x_i}^{y_j}}{p(y_j|x_i; \theta)} \right) = \arg\max_\theta \sum_i \sum_j d_{x_i}^{y_j} \ln p(y_j|x_i; \theta). \tag{1}
\]

It is interesting to examine the traditional learning paradigms under the optimization criterion shown in Eq. (1). For single-label learning (see Fig. 2(a) for example), \(d_{x_i}^{y_j} = K\delta(y_j, y_i)\), where \(K\delta(\cdot, \cdot)\) is the Kronecker delta function and \(y_i(x_i)\) is the single label of \(x_i\). Consequently, Eq. (1) can be simplified to

\[
\theta^* = \arg\max_\theta \sum_i \ln p(y_i(x_i)|x_i; \theta). \tag{2}
\]

This is actually the maximum likelihood (ML) estimation of \(\theta\). The later use of \(p(y|x; \theta)\) for classification is equivalent to the maximum a posteriori (MAP) decision.

For multi-label learning, each instance is associated with a label set (see Fig. 2(b) for example). Consequently, Eq. (1) can be changed into

\[
\theta^* = \arg\max_\theta \sum_i \frac{1}{|Y_i|} \sum_{y \in Y_i} \ln p(y|x_i; \theta), \tag{3}
\]

where \(Y_i\) is the label set associated with \(x_i\). Eq. (3) can be viewed as a ML criterion weighted by the reciprocal cardinality of the label set associated with each instance. In fact, this is equivalent to first applying Entropy-based Label Assignment (ELA) [31] to transform the multi-label instances into the weighted single-label instances, and then optimizing the ML criterion based on the weighted single-label instances.

It can be seen from the above analysis that with proper constraints, a LDL model can be transformed into the commonly used methods for single-label or multi-label learning. Thus, LDL may be viewed as a more general learning framework which includes both single-label and multi-label learning as its special cases.

### 3 LDL Algorithms

We follow three strategies to design algorithms for LDL. The first strategy is problem transformation, i.e., transform the LDL problem into existing learning paradigms. The second strategy is algorithm adaptation, i.e., extend existing learning algorithms to deal with label distributions. Finally, the third strategy is to design specialized algorithms according to the characteristics of LDL. Following each of the three strategies, two typical algorithms are proposed in this section.

#### 3.1 Problem Transformation

One straightforward way to transform a LDL problem into a single-label learning problem is to change the training examples to weighted single-label examples. In detail, each training example \((x_i, D_i)\) is transformed into \(c\) single-label examples \((x_i, y_j)\) with the weight \(d_{x_i}^{y_j}\), where \(i = 1, \ldots, n\) and \(j = 1, \ldots, c\). The training set is then resampled according to the weight of each example. The resampled training set becomes a standard single-label training set including \(c \times n\) examples, and then any single-label learning algorithms can be applied to the training set.

In order to predict the label distribution of a previously unseen instance \(x\), the learner must be able to output the confidence/probability for each label \(y_j\), which can be regarded as the description degree of the corresponding label, i.e., \(d_{x_i}^{y_j} = P(y_j|x)\). Two representative algorithms are adopted here for this purpose. One is the Bayes classifier, the other is SVM. In detail, the Bayes classifier assumes Gaussian distribution for each class, and the posterior probability computed by the Bayes rule is regarded as the description degree of the corresponding label. As to SVM, the probability estimates are obtained by a pairwise coupling method proposed in [33] for multi-class classification, which has also been implemented in the software LIBSVM [3]. Again, the probability for each label is regarded as the description degree of the corresponding label. When Bayes and SVM are applied to the resampled training set, the resulted methods are denoted by PT-Bayes and PT-SVM, respectively. Where ‘PT’ is the abbreviation of ‘Problem Transformation’.

#### 3.2 Algorithm Adaptation

Certain existing learning algorithms can be naturally extended to deal with label distributions, among which two adapted algorithms are proposed here. The first one is \(k\)-NN. Given a new instance \(x\), its \(k\) nearest neighbors are first found in the training set. Then, the mean of the label distributions of all the \(k\) nearest neighbors is calculated as the label distribution of \(x\), i.e.,

\[
p(y|x) = \frac{1}{k} \sum_{i \in N_k(x)} d_{x_i}^{y_j}, (j = 1, 2, \ldots, c), \tag{4}
\]

where \(N_k(x)\) is the index set of the \(k\) nearest neighbors of \(x\) in the training set. This adapted algorithm is denoted by AA-\(k\)-NN, where ‘AA’ is the abbreviation of ‘Algorithm Adaptation’.

The second algorithm is the backpropagation (BP) neural network. The three-layer neural network has \(q\) (the dimensionality of \(x\)) input units which receive \(x\), and \(c\) (the number of different labels) output units each of which outputs the
description degree of a label $y_j$. For single-label or multi-label learning, the desired output $t$ is usually a vector with 1s at the positions corresponding to the positive labels of the input instance, and 0s otherwise. For LDL, $t$ becomes the real label distribution of the input training instance. Thus, the target of the BP algorithm is to minimize the sum-squared error of the output of the neural network compared with the real label distributions. To make sure the output of the neural network $z = \{z_1, z_2, \ldots, z_c\}$ satisfies that $z_j \in [0, 1]$ for $j = 1, 2, \ldots, c$ and $\sum_j z_j = 1$, the softmax activation function is used in each output unit. Let the net input to the $j$-th output unit be $\eta_j$, then the softmax output $z_j$ is

$$z_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{c} \exp(\eta_k)} (j = 1, 2, \ldots, c), \quad (5)$$

This adapted algorithm is denoted by AA-BP.

### 3.3 Specialized Algorithms

Different from the indirect strategy of problem transformation and algorithm adaptation, the specialized algorithms directly match the LDL problem, e.g., by directly solving the optimization problem in Eq. (1). One good start toward this end might be our previous work on facial age estimation [11], [12], where in order to solve the insufficient training data problem, each face image is labeled by not only its chronological age, but also the neighboring ages. This actually forms a special label distribution with the highest description degree of the chronological age, and gradually decreasing description degrees on both neighboring sides of the chronological age. We proposed two algorithms IIS-LLD and CPNN for such special data, both of which were designed to solve an optimization problem similar to Eq. (1). Unfortunately, CPNN is only suitable for totally ordered labels (such as the age), and thus cannot be applied to the general LDL problem. As for IIS-LLD, although it was designed for a particular form of label distribution, it can be generalized to deal with any LDL problems. The optimization process of IIS-LLD, however, has been evidenced to not be very efficient [13]. So, an improved version is further proposed in this section.

IIS-LLD assumes the parametric model $p(y|x; \theta)$ to be the maximum entropy model [1], i.e.,

$$p(y|x; \theta) = \frac{1}{Z} \exp \left( \sum_k \theta_{y,k} g_k(x) \right), \quad (6)$$

where $Z = \sum_y \exp \left( \sum_k \theta_{y,k} g_k(x) \right)$ is the normalization factor, $\theta_{y,k}$ is an element in $\theta$, and $g_k(x)$ is the $k$-th feature of $x$. Substituting Eq. (6) into Eq. (1) yields the target function of $\theta$

$$T(\theta) = \sum_{i,j} \Delta_{i,j} \theta_{y_i,k} g_k(x_i)$$

$$- \sum_i \ln \sum_j \exp \left( \sum_k \theta_{y_j,k} g_k(x_j) \right). \quad (7)$$

In each step of the optimization, IIS-LLD updates the current estimate of the parameters $\theta$ to $\theta + \Delta$, where $\Delta$ maximizes a lower bound to the change of the target function output. The element of $\Delta$, $\Delta_{y,j,k}$, can be obtained by solving the equation

$$\sum_i p(y_i|x_i; \theta) g_k(x_i) \exp(\delta_{y_i,k} s(g_k(x_i)) g^{\#}(x_i))$$

$$- \sum_i \delta_{x_i,k} g_k(x_i) = 0,$$

where $g^{\#}(x_i) = \sum_k g_k(x_i)$ and $s(g_k(x_i))$ is the sign of $g_k(x_i)$. Eq. (8) can be solved by nonlinear equation solvers, such as the Gauss-Newton method. The pseudocode of IIS-LLD is given in Algorithm 1.

The optimization in IIS-LLD uses a strategy similar to Improved Iterative Scaling (IIS) [21], which is a well-known algorithm for maximizing the likelihood of the maximum entropy model. However, it has been reported in the literature [13] that IIS often performs worse than several other optimization algorithms such as conjugate gradient and quasi-Newton methods. Here we follow the idea of an effective quasi-Newton method BFGS [19] to further improve IIS-LLD.

Consider the second-order Taylor series of $T'(\theta) = -T(\theta)$ at the current estimate of the parameter vector $\theta^{(l)}$:

$$T'(\theta^{(l+1)}) \approx T'(\theta^{(l)}) + \nabla T'(\theta^{(l)})^T \Delta + \frac{1}{2} \Delta^T H(\theta^{(l)}) \Delta, \quad (9)$$

where $\Delta = \theta^{(l+1)} - \theta^{(l)}$ is the update step, $\nabla T'(\theta^{(l)})$ and $H(\theta^{(l)})$ are the gradient and Hessian matrix of $T'(\theta)$ at $\theta^{(l)}$, respectively. The minimizer of Eq. (9) is

$$\Delta^{(l)} = -H^{-1}(\theta^{(l)}) \nabla T'(\theta^{(l)}). \quad (10)$$

The line search Newton method uses $\Delta^{(l)}$ as the search direction $p^{(l)} = \Delta^{(l)}$ and updates the parameter vector by

$$\theta^{(l+1)} = \theta^{(l)} + \alpha^{(l)} p^{(l)}, \quad (11)$$

where the step length $\alpha^{(l)}$ is obtained from a line search procedure to satisfy the strong Wolfe conditions [19]:

$$T'(\theta^{(l)}) + \alpha^{(l)} p^{(l)} \leq T'(\theta^{(l)}) + c_1 \alpha^{(l)} \nabla T'(\theta^{(l)})^T p^{(l)}, \quad (12)$$

$$\| \nabla T'(\theta^{(l)}) + \alpha^{(l)} p^{(l)} \| \leq c_2 \| \nabla T'(\theta^{(l)}) \| p^{(l)}, \quad (13)$$

where $0 < c_1 < c_2 < 1$. 

---

**Algorithm 1: IIS-LLD**

**Input**: The training set $S = \{ (x_i, D_i) \}_{i=1}^n$ and the convergence criterion $\varepsilon$

**Output**: $p(y|x; \theta)$

1. Initialize the model parameter vector $\theta^{(0)}$;
2. $l \leftarrow 0$;
3. **repeat**
4. $l \leftarrow l + 1$;
5. Solve Eq. (8) for $\delta_{y,k}$;
6. $\theta^{(l)} \leftarrow \theta^{(l-1)} + \Delta$;
7. until $T(\theta^{(l)}) - T(\theta^{(l-1)}) < \varepsilon$;
8. $p(y|x; \theta) \leftarrow \frac{1}{Z} \exp \left( \sum_k \theta_{y,k} g_k(x) \right)$.

---


Algorithm 2: BFGS-LLD

\textbf{Input:} The training set \( S = \{ (x_i, D_i) \}_{i=1}^n \) and the convergence criterion \( \varepsilon \)

\textbf{Output:} \( p(y | x; \theta) \)

1. Initialize the model parameter vector \( \theta^{(0)} \);
2. Initialize the inverse Hessian approximation \( B^{(0)} \);
3. Compute \( \nabla T'(\theta^{(0)}) \) by Eq. (13);
4. \( l \leftarrow 0 \);
5. repeat
   6. Compute search direction \( p^{(l)} \leftarrow -B^{(l)} \nabla T'(\theta^{(l)}) \);
   7. Compute the step length \( \alpha^{(l)} \) by a line search procedure to satisfy Eq. (12) and (13);
   8. Compute \( \theta^{(l+1)} \leftarrow \theta^{(l)} + \alpha^{(l)} p^{(l)} \);
   9. Compute \( \nabla T'(\theta^{(l+1)}) \) by Eq. (18);
   10. \( s^{(l)} \leftarrow \theta^{(l+1)} - \theta^{(l)} \);
   11. \( u^{(l)} \leftarrow \nabla T'(\theta^{(l+1)}) - \nabla T'(\theta^{(l)}) \);
   12. \( \rho^{(l)} \leftarrow \frac{1}{s^{(l)} T u^{(l)}} \);
   13. \( B^{(l+1)} = (I - \rho^{(l)} s^{(l)} (u^{(l)})^T) B^{(l)} (I - \rho^{(l)} u^{(l)} (s^{(l)})^T) + \rho^{(l)} s^{(l)} (s^{(l)})^T \); \( l \leftarrow l + 1 \);
6. until \( \| \nabla T'(\theta^{(l)}) \| < \varepsilon \);
7. \( p(y | x; \theta) \leftarrow \frac{1}{Z} \exp \left( \sum_k \theta_{y,k} g_k(x) \right) \);

One problem of the above method is the calculation of the inverse Hessian matrix in each iteration, which is computationally expensive. The idea of BFGS is to avoid explicit calculation of \( H^{-1}(\theta^{(l)}) \) by approximating it with an iteratively updated matrix \( B \):

\[
B^{(l+1)} = (I - \rho^{(l)} s^{(l)} (u^{(l)})^T) B^{(l)} (I - \rho^{(l)} u^{(l)} (s^{(l)})^T) + \rho^{(l)} s^{(l)} (s^{(l)})^T ,
\]

where

\[
s^{(l)} = \theta^{(l+1)} - \theta^{(l)},
\]

\[
u^{(l)} = \nabla T'(\theta^{(l+1)}) - \nabla T'(\theta^{(l)}),
\]

\[
\rho^{(l)} = \frac{1}{s^{(l)} T u^{(l)}}.
\]

As to the optimization of the target function \( T'(\theta) \), the computation of BFGS is mainly related to the first-order gradient, which can be obtained through

\[
\frac{\partial T'(\theta)}{\partial \theta_{y,k}} = \sum_i \exp \left( \sum_k \theta_{y,j,k} g_k(x_i) \right) g_k(x_i) - \sum_i d^{y_i}_i g_k(x_i).
\]

Thus it performs much more efficiently than the standard line search Newton method, and based on previous studies \cite{18}, it stands a good chance of outperforming the IIS-based algorithm IIS-LLD. This improved algorithm is denoted by BFGS-LLD, and its pseudocode is shown in Algorithm 2.

### 3.4 Evaluation Measures

The output of a LDL algorithm is a label distribution, which is different from both the single label output of single-label learning and the label set output of multi-label learning. Accordingly, the evaluation measures for LDL algorithms should be different from those used for single-label and multi-label learning algorithms. A natural choice of such measure is the average distance or similarity between the predicted label distribution and the real label distribution. There are many measures for the distance/similarity between probability distributions \cite{2}, which can be well borrowed to measure the distance/similarity between label distributions. In this paper, six representative measures are selected as the evaluation measures for the LDL algorithms. Four of them are distance measures (the smaller the better), i.e., Euclidean, Sørensen, Squared \( \chi^2 \), and Kullback-Leibler (K-L), and two of them are similarity measures (the larger the better), i.e., Intersection and Fidelity. Suppose the real label distribution is \( D = \{ d_1, d_2, \ldots, d_q \} \), the predicted label distribution is \( \hat{D} = \{ \hat{d}_1, \hat{d}_2, \ldots, \hat{d}_q \} \), then the formulae of the six measures are summarized in Table 1.

On a particular dataset, each of the measures may reflect a certain aspect of an algorithm. It is hard to say which measure is the best. Thus we suggest to use all of the six measures when comparing different LDL algorithms, which is analogous to the common practice of using multiple evaluation measures for MLL algorithms.

### 4 Experiments

#### 4.1 Methodology

There are three datasets used in the experiments: an artificial toy dataset, a moderate-sized real-world dataset (Yeast Gene) with ten subsets, and a large-scale real-world dataset (Human Gene). Some basic statistics about these three datasets are given in Table 2.

The artificial toy dataset is generated to show in a direct and visual way whether the LDL algorithms can learn the mapping from the instance to the label distribution. In this dataset, the instance \( x \) is of three-dimensional and there are three labels, i.e., \( q = 3 \) and \( c = 3 \). The label distribution

\[
1. \text{The datasets and the Matlab code used in the experiments can be downloaded from http://cse.seu.edu.cn/PersonalPage/xgeng/LLD/index.htm}
\]

| Name                          | Formula                                                                 |
|-------------------------------|------------------------------------------------------------------------|
| Distance                      |                                                                         |
| Euclidean                     | \( D_{\text{Dis}}(D, \hat{D}) = \sqrt{\sum_{j=1}^{q} (d_j - \hat{d}_j)^2} \) |
| Sørensen                      | \( D_{\text{Dis}}(D, \hat{D}) = \frac{2 \sum_{j=1}^{q} |d_j - \hat{d}_j|}{\sum_{j=1}^{q} (d_j + \hat{d}_j)} \) |
| Squared \( \chi^2 \)         | \( D_{\text{Dis}}(D, \hat{D}) = \sum_{j=1}^{q} \frac{(d_j - \hat{d}_j)^2}{d_j + \hat{d}_j} \) |
| Kullback-Leibler (K-L)        | \( D_{\text{Dis}}(D, \hat{D}) = \sum_{j=1}^{q} d_j \ln \frac{d_j}{\hat{d}_j} \) |
| Similarity                    |                                                                         |
| Intersection                  | \( S_{\text{Int}}(D, \hat{D}) = \min_{k=1}^{c} \sum_{j=1}^{q} d_j \)    |
| Fidelity                      | \( S_{\text{Fid}}(D, \hat{D}) = \sum_{j=1}^{q} \sqrt{d_j \hat{d}_j} \)  |
$D = \{d_{x1}^w, d_{x2}^w, d_{x3}^w\}$ of an instance $x = [x_1, x_2, x_3]^T$ is created in the following way.

$$
t_1 = ax_1 + bx_2^2 + cx_3^3 + d,
$$
(19)

$$
t_2 = ax_2 + bx_2^2 + cx_3^3 + d,
$$
(20)

$$
t_3 = ax_3 + bx_2^2 + cx_3^3 + d,
$$
(21)

$$
\psi_1 = (w_1^T t)^2,
$$
(22)

$$
\psi_2 = (w_2^T t + \lambda_1 \psi_1)^2,
$$
(23)

$$
\psi_3 = (w_3^T t + \lambda_2 \psi_2)^2,
$$
(24)

$$
d_{x1}^w = \frac{\psi_1}{\psi_1 + \psi_2 + \psi_3},
$$
(25)

$$
d_{x2}^w = \frac{\psi_2}{\psi_1 + \psi_2 + \psi_3},
$$
(26)

$$
d_{x3}^w = \frac{\psi_3}{\psi_1 + \psi_2 + \psi_3},
$$
(27)

where $t = [t_1, t_2, t_3]^T$. Note that Eq. (23) and (24) deliberately make the description degree of one label depend on those of other labels. The parameters in Eq. (19)–(24) are set as $a = 1$, $b = 0.5$, $c = 0.2$, $d = 1$, $w_1 = [4, 2, 1]^T$, $w_2 = [1, 2, 4]^T$, $w_3 = [1, 4, 2]^T$, and $\lambda_1 = \lambda_2 = 0.01$. To generate the training set, each component of $x$ is uniformly sampled within the range $[-1, 1]$. In total, there are 500 instances sampled in this way. Then, the label distribution corresponding to each instance is calculated via Eq. (19)–(27). Such 500 examples are used as the training set for the LDL algorithms proposed in Section 3.

In order to show the result of the LDL algorithms in a direct and visual way, the test examples of the toy dataset are selected from a certain manifold in the instance space. The first two components of the test instance $x$, $x_1$ and $x_2$, are located at a grid of the interval 0.01 within the range $[-1, 1]$ on both dimensions, i.e., there are in total $201 \times 201 = 40,401$ test instances. The third component $x_3$ is calculated by

$$
x_3 = \sin((x_1 + x_2) \times \pi).
$$
(28)

Then, the label distribution of each test instance, either the ground-truth calculated via Eq. (19)–(27) or the prediction given by the LDL algorithms, is transformed into a color. Thus the ground-truth and predicted label distributions of the test instances can be compared visually through the color pattern on the manifold.

The second dataset is a real-world dataset collected from ten biological experiments on the budding yeast *Saccharomyces cerevisiae* [8]. The result of each of the ten experiments constitutes a subset of this dataset. There are in total 2,465 yeast genes included, each of which is represented by an associated phylogenetic profile vector of length 24. The labels correspond to the discrete time points in different biological experiments. The gene expression level (after normalization) at each time point provides a natural measure of the description degree of the corresponding label. The number of labels in the ten subsets of the Yeast Gene dataset is summarized in Table 2. The description degrees (normalized gene expression level) of all the labels (time points) constitute a label distribution for a particular yeast gene.

The third dataset is a large-scale real-world dataset collected from the biological research on the relationship between human genes and diseases. There are in total 30,542 human genes included in this dataset, each of which is represented by the 36 numerical descriptors for a gene sequence proposed in [55]. The labels correspond to 68 different diseases. The gene expression level (after normalization) for each disease is regarded as the description degree of the corresponding label. The description degrees (normalized gene expression level) of all the 68 labels (diseases) constitute a label distribution for a particular human gene.

All the six algorithms described in Section 3, i.e., PT-Bayes, PT-SVM, AA-kNN, AA-BP, IIS-LLD, and BFGS-LLD, are applied to the three datasets shown in Table 2 and compared by the six measures listed in Table 1. On the Yeast Gene and Human Gene datasets, ten-fold cross validation is conducted for each algorithm and the mean value and standard deviation of each evaluation measure are recorded.

For each algorithm, several parameter configurations are tried. On the artificial dataset, one fifth of the training set is randomly selected as the validation set. The model is trained on those examples left in the training set and tested on the validation set to select the best parameters. Then, the model is trained on the whole training set with the best parameters and tested on the test set. On the Yeast Gene and Human Gene datasets, the parameter selection process is nested into the ten-fold cross validation. In detail, the whole data is first randomly split into 10 chunks. Each time, one chunk is used as test set, another is used as validation set, and the rest 8 chunks are used as training set. Then, the model is trained with different parameter settings on the training set and tested on the validation set. This procedure is repeated 10 times (each time with different training and validation sets), and the parameter setting with the best average performance is selected. After that, the original validation set is merged into the training set and the test set remains unchanged. The model is trained with the selected parameter setting on the

| Dataset | Artificial | Yeast Gene (Ten Subsets) | Human Gene |
|---------|------------|--------------------------|------------|
|         | # Examples (n) | Artifical | alpha | cdc | elu | diau | heat | spo | cold | dtt | spo5 | spoem |
|         | 500 (training) | 40,401 (test) | 2,465 | 2,465 | 2,465 | 2,465 | 2,465 | 2,465 | 2,465 | 2,465 | 2,465 | 30,542 |
| # Features (q) | 3 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 36 |
| # Labels (c) | 3 | 18 | 15 | 14 | 7 | 6 | 6 | 4 | 4 | 3 | 2 | 68 |
Fig. 4. Comparison between the ground-truth and predicted label distributions (regarded as RGB colors) on the artificial test manifold.
updated training set and tested on the test set. This procedure is repeated 10 times and the average performance is recorded.

4.2 Results

4.2.1 Artificial Dataset

In order to visually show the label distributions of the test examples of the artificial dataset, the description degrees of the three labels are regarded as the three color channels of the RGB color space. Then, each label distribution is converted into a color, and the predictions made by the LDL algorithms can be compared with the ground-truth label distributions through observing the color patterns on the manifold where the test examples lie on. For easier comparison, the images are visually enhanced by applying a decorrelation stretch process. The results are shown in Fig. 4. It can be seen that the two specialized LDL algorithms, IIS-LLD and BFGS-LLD, predict almost identical color patterns with the ground-truth. PT-Bayes and AA-kNN can also discover similar color patterns with the ground-truth. However, PT-SVM and AA-BP fail to obtain a reasonable result.

For more detailed comparison, the six numerical evaluation measures listed in Table 1 are given in Table 3. For the four distance measures (Euclidean, Sørensen, Squared χ², and K-L), “↓” indicates “the smaller the better”. For the two similarity measures (Intersection and Fidelity), “↑” indicates “the larger the better”. The best performance on each measure is highlighted by boldface. As can be seen that the specialized algorithms (IIS-LLD and BFGS-LLD) can generally give better label distribution predictions compared with other algorithms, in the sense of either visual similarity or numerical evaluation measures.

The experimental results of the six LDL algorithms on the ten subsets of the Yeast Gene dataset are tabulated in Table 5. On each subset and each measure, the algorithms are ranked in decreasing order of their performance indicated by the mean value of the corresponding measure. The ranks are given in the brackets right after the performance values. The average rank of each algorithm over all subsets is calculated for each measure and given in the rows named “Avg. Rank”. There are in total 60 pairs of subset and measure in Table 5, i.e., 10 subsets × 6 measures. In each case specified by the (subset, measure) pair, the best (rank 1) performance among the six compared algorithms is highlighted by boldface. Based on the average rank, the specialized LDL algorithms (IIS-LLD and BFGS-LLD) perform remarkably better than those algorithms obtained from problem transformation (PT-Bayes and PT-SVM) and algorithm adaptation (AA-kNN and AA-BP). In all the 60 cases, BFGS-LLD and IIS-LLD consistently achieve the best and the second best performance, respectively. The performance comparison result between the ‘PT’ algorithms and the ‘AA’ algorithms, however, is not so constant: while the average rank of PT-SVM is before that of AA-kNN, the performance of PT-Bayes ranks consistently after that of AA-BP.

4.2.2 Yeast Gene Dataset

To give a direct idea of the predictions made by the LDL algorithms, some typical examples of the predicted label distributions on the Yeast Gene dataset given by the six LDL algorithms are first shown in Table 4. The first row shows the real label distributions of four typical test instances with 18, 7, 4 and 2 labels, respectively. Each of the following rows shows the corresponding predictions of one LDL algorithm. The evaluation measures are given under each predicted label distribution in two sets. The first set includes the four distance measures {Euclidean, Sørensen, Squared χ², K-L}, and the second set includes the two similarity measures {Intersection, Fidelity}. The best performance in each case is highlighted by boldface. As can be seen that the specialized algorithms (IIS-LLD and BFGS-LLD) can generally give better label distribution predictions compared with other algorithms, in the sense of the number of examples, features, and labels. However, the relative performances, i.e., the rankings, of the algorithms are similar to those on the Yeast Gene dataset. This is because that the Yeast Gene dataset is much more complex than the Yeast Gene dataset in the sense of the number of examples, features, and labels.
| Algorithm      | Real 18 Labels | Real 7 Labels | Real 4 Labels | Real 2 Labels |
|---------------|----------------|--------------|--------------|--------------|
| PT-Bayes      | 0.052, 0.091, 0.363, 0.0408 | 0.3561, 0.4095, 0.4014, 0.3532 | 0.4249, 0.4024, 0.4569, 0.7929 | 0.3915, 0.2768, 0.1703, 0.1907 |
| PT-SVM        | 0.0161, 0.0252, 0.0266, 0.0025 | 0.0321, 0.0305, 0.0039, 0.0038 | 0.0165, 0.0123, 0.0005, 0.0005 | 0.0813, 0.0575, 0.0066, 0.0066 |
| AA-ANN        | 0.0162, 0.0240, 0.0225, 0.0025 | 0.0377, 0.0486, 0.0052, 0.0053 | 0.0139, 0.0107, 0.0004, 0.0004 | 0.0809, 0.0572, 0.0066, 0.0066 |
| AA-BP         | 0.0974, 0.1733, 0.0856, 0.0274 | 0.0618, 0.0770, 0.0150, 0.0949 | 0.2933, 0.2415, 0.2773, 0.8120 | 0.3679, 0.2601, 0.1487, 0.1638 |
| IIS-LLD       | 0.0116, 0.0156, 0.0013, 0.0013 | 0.0310, 0.0365, 0.0034, 0.0034 | 0.0131, 0.0114, 0.0003, 0.0003 | 0.0039, 0.0027, 0.0, 0 |
| BFGS-LLD      | 0.0104, 0.0145, 0.0011, 0.0010 | 0.0268, 0.0325, 0.0026, 0.0026 | 0.0101, 0.0088, 0.0002, 0.0002 | 0.0076, 0.0054, 0.0001, 0.0001 |

**TABLE 4**
Comparison of the Typical Real and Predicted Label Distributions on the Yeast Gene Dataset
### TABLE 5
Experimental Results (mean±std(rank)) of the Six LDL Algorithms on the Ten Subsets of the Yeast Gene Dataset

| Criterion | Subset | PT-Bayes | PT-SVM | AA-KNN | AA-BP | IIS-LLD | BFGS-LD |
|-----------|--------|----------|--------|--------|--------|---------|---------|
|            |        |          |        |        |        |         |         |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |
|            |        | 3.0      | 3.4    | 5.6    | 5.0    | 5.0     | 1.0     |

- **Subset**
  - **alpha**: $0.1502\pm0.0129(6)$, $0.2433\pm0.0070(3)$, $0.2429\pm0.0060(4)$, $0.4804\pm0.0300(5)$, $0.2400\pm0.0070(2)$, $0.2031\pm0.0007(1)$
  - **cdc**: $0.1584\pm0.0135(6)$, $0.2391\pm0.0080(3)$, $0.3013\pm0.0080(4)$, $0.5566\pm0.0270(5)$, $0.2806\pm0.0080(2)$, $0.2797\pm0.0008(1)$
  - **elu**: $0.1627\pm0.0145(6)$, $0.2942\pm0.0080(3)$, $0.2979\pm0.0090(4)$, $0.5538\pm0.0310(5)$, $0.2286\pm0.0080(2)$, $0.2777\pm0.0008(1)$
  - **diau**: $0.2147\pm0.0176(6)$, $0.6522\pm0.0080(3)$, $0.5932\pm0.0090(4)$, $0.9345\pm0.0310(5)$, $0.6510\pm0.0080(2)$, $0.5531\pm0.0020(1)$
  - **heat**: $0.2279\pm0.0226(6)$, $0.6201\pm0.0170(3)$, $0.6241\pm0.0190(4)$, $0.9671\pm0.0170(5)$, $0.6004\pm0.0180(2)$, $0.5924\pm0.0170(1)$
  - **elu**: $0.2292\pm0.0162(6)$, $0.9001\pm0.0300(3)$, $0.8797\pm0.0290(3)$, $1.0922\pm0.0850(5)$, $0.8381\pm0.0210(2)$, $0.8210\pm0.0210(1)$
  - **cold**: $0.2376\pm0.0262(6)$, $0.7503\pm0.0424(3)$, $0.7242\pm0.0263(3)$, $1.1386\pm0.0312(5)$, $0.6986\pm0.0312(2)$, $0.6862\pm0.0291(1)$
  - **dtt**: $0.2242\pm0.0300(6)$, $0.5080\pm0.0303(3)$, $0.5127\pm0.0184(3)$, $0.9272\pm0.0969(5)$, $0.5000\pm0.0220(2)$, $0.4800\pm0.0201(1)$
  - **spo5**: $0.2707\pm0.0137(6)$, $1.1967\pm0.0454(3)$, $1.2115\pm0.0484(4)$, $1.4927\pm0.0115(5)$, $1.1833\pm0.0342(2)$, $1.1711\pm0.0391(1)$
  - **spoem**: $0.2608\pm0.0232(6)$, $1.2897\pm0.0686(4)$, $1.2776\pm0.0654(3)$, $1.7073\pm0.0203(5)$, $1.2523\pm0.0512(2)$, $1.2281\pm0.0664(1)$

**Avg. Rank**: 21.9

**Squares y^2**: 1.8

**K-L ↓**: 11.0

**Intersection ↑**: 10.0

**Fidelity ↑**: 11.0

**Avg. Rank**: 3.0

**Squares y^2**: 2.2

**K-L ↓**: 10.0

**Intersection ↑**: 9.0

**Fidelity ↑**: 9.0

**Avg. Rank**: 3.0

**Squares y^2**: 2.2

**K-L ↓**: 10.0

**Intersection ↑**: 9.0

**Fidelity ↑**: 9.0

**Avg. Rank**: 3.0
In summary, the experimental results on all of the artificial, Yeast Gene and Human Gene datasets show that the specialized algorithms directly designed for the LDL problem (IIS-LLD and BFGS-LLD) generally perform better than those transformed from traditional learning algorithms (PT-Bayes, PT-SVM, AA-\(k\)NN, and AA-BP). This is due to the advantages of the specialized algorithms in discovering the overall label distributions. By using more effective optimization process, BFGS-LLD remarkably improves the performance of IIS-LLD. This also indicates the necessity of further research on the algorithms specially designed for the LDL problem.

5 SUMMARY AND DISCUSSIONS

This paper proposes label distribution learning, which is a more general learning framework than single-label and multi-label learning. It can deal with not only multiple labels of one instance, but also the different importance of these labels. This paper proposes six working LDL algorithms in three ways: problem transformation, algorithm adaptation, and specialized algorithm design. In order to compare these algorithms, six evaluation measures are used and the first batch of label distribution datasets are prepared and made publicly available. Experimental results on one artificial and two real-world datasets show clear advantage of the specialized algorithms. This illustrates that the characteristics of LDL require special design to achieve good performance.

This paper is motivated by the real-world data with natural measures of description degrees (e.g., gene expression level). However, as a general learning framework, label distribution learning might also be used to solve other kinds of problems. Generally speaking, there are at least three scenarios where label distribution learning could be helpful:

1) There is a natural measure of description degree that associates the labels with the instances. This is the most direct application of label distribution learning, as described in this paper.

2) When there are multiple labeling sources (e.g., multiple experts) for one instance, the annotations from different sources might be significantly inconsistent. In such case, it is usually better for the learning algorithm to integrate the labels from all the sources rather than to decide one or more ‘winning label(s)’ via majority voting \([23]\). One good way to incorporate all the labeling sources is to generate a label distribution for the instance: the label favored by more sources is given a higher description degree, while that chosen by fewer sources is assigned with a lower description degree. In this way, the multi-labeling-source problem is transformed into a label distribution learning problem.

3) Some labels are highly correlated with other labels. Utilizing such correlation is one of the most important approaches to improve the learning process \([16], [32], [29]\). Label distribution learning provides a new way toward this purpose. The key step is to transform a single-label or multi-label learning problem into a label distribution learning problem. This can be achieved by generating a label distribution for each instance according to the correlation among the labels.

Each of the three scenarios actually covers a vast area of applications. A lot of interesting work, both at the theoretical level and at the application level, may be conducted in the future.

ACKNOWLEDGMENT

This research was supported by the National Science Foundation of China (61273300, 61232007), and Jiangsu Natural Science Funds for Distinguished Young Scholar (BK20140022).

REFERENCES

[1] A. L. Berger, S. D. Pietra, and V. J. D. Pietra, “A maximum entropy approach to natural language processing,” Computational Linguistics, vol. 22, no. 1, pp. 39–71, 1996.

[2] S.-H. Cha, “Comprehensive survey on distance/similarity measures between probability density functions,” International Journal of Mathematical Models and Methods in Applied Sciences, vol. 1, pp. 300–307, 2007.

[3] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” ACM Transactions on Intelligent Systems and Technology, vol. 2, pp. 27:1–27:27, 2011.

[4] W. Cheng, K. Dembczynski, and E. Hüllermeier, “Graded multilabel classification: The ordinal case,” in Proc. 27th Int’l Conf. Machine Learning, Haifa, Israel, 2010, pp. 223–230.

[5] W. Cheng and E. Hüllermeier, “Combining instance-based learning and logistic regression for multilabel classification,” Machine Learning, vol. 76, no. 2-3, pp. 211–225, 2009.

[6] A. Clare and R. D. King, “Knowledge discovery in multi-label phenotype data,” in Proc. European Conf. Principles of Data Mining and Knowledge Discovery, Freiburg, Germany, 2001, pp. 42–53.

[7] T. Denoeux and L. M. Zouhal, “Handling possibilistic labels in pattern classification using evidential reasoning,” Fuzzy Sets and Systems, vol. 122, no. 3, pp. 409–424, 2001.

[8] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, “Cluster analysis and display of genome-wide expression patterns,” Proceedings of the National Academy of Science, vol. 95, no. 25, pp. 14 863–14 868, 1998.

[9] J. Fürrnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, “Multilabel classification via calibrated label ranking,” Machine Learning, vol. 73, no. 2, pp. 133–153, 2008.
[10] X. Geng and R. Ji, “Label distribution learning,” in Proc. 2013 Int’l Conf. Data Mining Workshops, Dallas, TX, 2013, pp. 377–383.

[11] X. Geng, K. Smith-Miles, and Z.-H. Zhou, “Facial age estimation by learning from label distributions,” in Proc. 24th AAAI Conf. Artificial Intelligence, Atlanta, GA, 2010, pp. 451–456.

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

[13] R. Guimerà and L. A. N. Amaral, “Functional cartography of complex metabolic networks,” Nature, vol. 433, pp. 895–900, 2005.

[14] Y. Guo and S. Gu, “Multi-label classification using conditional dependency networks,” in Proc. 22nd Int’l Joint Conf. Artificial Intelligence, Barcelona, Spain, 2011, pp. 1300–1305.

[15] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker, “Label ranking by pairwise preferences,” Artif. Intell., vol. 172, no. 16-17, pp. 1897–1916, 2008.

[16] F. Kang, R. Jin, and R. Suthikankan, “Correlated label propagation with application to multi-label learning,” in Proc. IEEE Conf. Computer Vision and Pattern Recognition, New York, NY, 2006, pp. 1719–1726.

[17] P. Li, H. Li, and M. Wu, “Multi-label ensemble based on variable pairwise constraint projection,” Information Sciences, vol. 222, pp. 269–281, 2013.

[18] R. Malouf, “A comparison of algorithms for maximum entropy parameter estimation,” in Proc. 6th Conf. Computational Natural Language Learning, Taipei, Taiwan, 2002, pp. 49–55.

[19] J. Nocedal and S. Wright, Numerical Optimization, 2nd ed. New York, NY: Springer, 2006.

[20] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek, “Uncovering the overlap- ping community structure of complex networks in nature and society,” Nature, vol. 433, pp. 814–818, 2005.

[21] S. D. Pietra, V. J. D. Pietra, and J. D. Lafferty, “Inducing features of random fields,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 4, pp. 380–393, 1997.

[22] B. Quost and T. Denoeux, “Learning from data with uncertain labels by boosting credal classifiers,” in Proc. 1st ACM SIGKDD Workshop on Knowledge Discovery from Uncertain Data, Paris, France, 2009, pp. 38–47.

[23] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, “Learning from crowds,” Journal of Machine Learning Research, vol. 11, no. Apr, pp. 1297–1322, 2010.

[24] J. Read, A. Bifet, G. Holmes, and B. Pfahringer, “Scalable and efficient multi-label classification for evolving data streams,” Machine Learning, vol. 88, no. 1-2, pp. 243–272, 2012.

[25] J. Read, B. Pfahringer, G. Holmes, and E. Frank, “Classifier chains for multi-label classification,” Machine Learning, vol. 85, no. 3, pp. 333–359, 2011.

[26] J. Read, B. Pfahringer, and G. Holmes, “Multi-label classification using ensembles of pruned sets,” in Proc. IEEE Int’l Conf. Data Mining, Pisa, Italy, 2008, pp. 995–1000.

[27] E. P. Sapozhnikova, “Art-based neural networks for multi-label classification,” in Int’l Symposium Intelligent Data Analysis, Lyon, France, 2009, pp. 167–177.

[28] P. Smyth, “Learning with probabilistic supervision,” in Computational Learning Theory and Natural Learning System, T. Petsche, Ed. MA: MIT Press, 1995, vol. III, pp. 163–182.

[29] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, “Multi-label learning with weak label,” in Proc. 24th AAAI Conf. Artificial Intelligence, Atlanta, GA, 2010, pp. 593–598.

[30] G. Tsoumakas, M.-L. Zhang, and Z.-H. Zhou, “Tutorial on learning from multi-label data,” in European Conf. Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Bled, Slovenia, 2009.

[31] G. Tsoumakas and I. Katakis, “Multi-label classification: An overview,” Int’l Journal of Data Warehousing and Mining, vol. 3, no. 3, pp. 1–13, 2007.

[32] H. Wang, M. Huang, and X. Zhu, “A generative probabilistic model for multi-label classification,” in Proc. 8th IEEE Int’l Conf. Data Mining, Pisa, Italy, 2008, pp. 628–637.

[33] T.-F. Wu, C.-J. Lin, and R. C. Weng, “Probability estimates for multiclass classification by pairwise coupling,” Journal of Machine Learning Research, vol. 5, pp. 975–1005, 2004.

[34] J. D. Wulfkuhle, L. A. Liotta, and E. F. Petricoin, “Proteomic applications for the early detection of cancer,” Nature Reviews Cancer, vol. 3, no. 4, pp. 267–275, 2003.

[35] J.-F. Yu, D.-K. Jiang, K. Xiao, Y. Jin, J.-H. Wang, and X. Sun, “Discriminate the falsely predicted protein coding genes in aeropyrum pernix k1 genome based on graphical representation,” MATCH Commun. Math. Comput. Chem., vol. 67, no. 3, pp. 845–866, 2012.

[36] M.-L. Zhang and Z.-H. Zhou, “Multilabel neural networks with applications to functional genomics and text categorization,” IEEE Trans. Knowl. Data Eng., vol. 18, no. 10, pp. 1338–1351, 2006.

[37] J.-H. Zimmermann, Ed., Practical Applications of Fuzzy Technologies. Netherlands: Kluwer Academic Publishers, 1999.