Multi-Label Learning by Exploiting Label Dependency

Min-Ling Zhang [1,2]
[1] School of Computer Science and Technology, Southeast University
Nanjing 210096, China
[2] National Key Laboratory for Novel Software Technology, Nanjing University
Nanjing 210093, China
zhangml@seu.edu.cn

Kun Zhang
Max Planck Institute for Biological Cybernetics
72076 Tübingen
Germany
kzhang@tuebingen.mpg.de

ABSTRACT

In multi-label learning, each training example is associated with *a set of* labels and the task is to predict the proper label set for the unseen example. Due to the tremendous (exponential) number of possible label sets, the task of learning from multi-label examples is rather challenging. Therefore, the key to successful multi-label learning is how to effectively exploit *correlations* between different labels to facilitate the learning process. In this paper, we propose to use a Bayesian network structure to efficiently encode the *conditional dependencies* of the labels as well as the feature set, with the feature set as the common parent of all labels. To make it practical, we give an approximate yet efficient procedure to find such a network structure. With the help of this network, multi-label learning is decomposed into a series of single-label classification problems, where a classifier is constructed for each label by incorporating its parental labels as additional features. Label sets of unseen examples are predicted recursively according to the label ordering given by the network. Extensive experiments on a broad range of data sets validate the effectiveness of our approach against other well-established methods.

Categories and Subject Descriptors

I.2.6 [**Computing Methodologies**]: Learning—*concept learning, induction*

General Terms

Algorithms

1. INTRODUCTION

Traditional supervised learning works under the *single-label* scenario, i.e. each example is associated with *one* single label characterizing its property. However, in many real-world applications, objects are usually associated with *multiple* labels simultaneously. To name a few, in text categorization, each document may belong to several topics, such as *Shanghai World Expo*, *economics* and even *volunteers* [14, 19]; In bioinformatics, each gene may be associated with a number of functional classes, such as *metabolism*, *transcription* and *protein synthesis* [7]; In automatic video annotation, each video clip may be related to several semantic classes, such as *urban* and *building* [16]. In multi-label learning, each example in the training set is represented by a feature vector and associated with *a set of* labels. The task is then to predict the label sets of unseen examples through analyzing training examples with known label sets.

Formally, learning from multi-label examples corresponds to find a mapping from the space of features to the space of label sets, i.e. the *power set* of all labels. Therefore, when there is large or even moderate number of labels, the task of multi-label learning would become rather challenging due to the tremendous (*exponential*) number of possible label sets. To cope with this issue, it is deemed that the *correlations* between different labels should be exploited to facilitate multi-label learning [21, 23]. For example, the probability of an image be annotated with label *Africa* would be high if we know it has labels *lion* and *grassland*; a document is unlikely to be labeled as *politics* if we know it is related to *entertainment*. Thus, effective exploitation of correlation information among different labels is crucial for the success of any multi-label learning system.

Roughly speaking, existing strategies to multi-label learning problems can be characterized into the following categories based on the *order of correlations* considered by the system:

- *First-order* approaches: The task of multi-label learning is tackled by considering decomposing it into a number of *independent* binary classification problems, one for each possible label [1, 4, 5, 29].
- *Second-order* approaches: The task of multi-label learning is tackled by considering the *pairwise* relations between labels, such as the ranking between the proper label and the improper label of an example [7, 8, 19, 28], or the interaction between any pair of labels [9, 16, 24, 30].
- *High-order* approaches: The task of multi-label learning is tackled by considering the *high-order* relations between labels, such as the full-order style of imposing all other labels' influences on each label in an indirect manner [3, 10, 11, 25], or the random style of combining an ensemble of classifiers each addressing correlations among a random subset of labels [17, 18, 22].

KDD'10, July 25–28, 2010, Washington, DC, USA.

First-order approaches simply ignore the correlations between different labels and this may weaken the generalization abilities of these approaches. For the latter two strategies however, their model complexities are usually high due to the exploitation of label combinations. Furthermore, the generality of these two strategies is also limited: a) Second-order approaches may suffer from the fact that the correlations between different labels would possibly go beyond second-order. b) The full-order approaches may not work well when certain structures exist among labels (e.g. label subgroups), while the random approaches may not work well due to their randomness in addressing label correlations.

In this paper, we aim to address the label correlations in an effective yet computational efficient way. Specifically, a novel approach named LEAD (*multi-label Learning by Exploiting lAbel Dependency*) is proposed to learn from multi-label examples.

At first, a Bayesian network (or *directed acyclic graph*, DAG) is built to characterize the joint probability of all labels conditioned on the feature set, such that correlations among labels are *explicitly* expressed through their dependency relations represented by the DAG structure. After that, a binary classifier is learned for each label by treating its parental labels in the DAG as additional input features. Finally, the label sets of unseen examples are predicted by reasoning with the identified Bayesian network together with the learned binary classifiers.

In contrast to other multi-label learning approaches, LEAD bears the following advantages through employing Bayesian network: 1) The underlying structure inherent in the label space is *explicitly* expressed in a compact way, which offers a promising opportunity to gain further insights on the concerned learning problem; 2) It is capable of addressing *arbitrary* order of label correlations, where the order of dependency is "controlled" by the number of parents of each label; 3) The model complexity is *linear* to the number of possible labels (one binary classifier per label), and making predictions for unseen example is *straightforward* with respect to the Bayesian network and the learned classifiers.

Extensive experiments across a broad range of multi-label data sets show that LEAD achieves highly competitive performance to the well-established first-order, second-order as well as high-order approaches.

The rest of this paper is organized as follows. Section 2 presents the LEAD approach. Section 3 reports our experimental results. Finally, Section 4 concludes.

2. THE LEAD APPROACH

Let $\mathcal{X} = \mathcal{R}^d$ be the d-dimensional input space and $\mathcal{Y} = \{1, 2, \ldots, q\}$ be finite set of q possible labels. Given a multi-label training set $\mathcal{D} = \{(\boldsymbol{x}_i, Y_i) \mid 1 \le i \le m\}$, where $\boldsymbol{x}_i \in \mathcal{X}$ is a feature vector and $Y_i \subseteq \mathcal{Y}$ is the set of labels associated with $\boldsymbol{x}_i$, the goal of multi-label learning is to learn a function $h : \mathcal{X} \to 2^{\mathcal{Y}}$ from $\mathcal{D}$ which maps each unseen example to a set of proper labels. From the Bayesian point of view, this problem can be reduced to model the conditional joint distribution of $P(\boldsymbol{y}|\boldsymbol{x})$, where $\boldsymbol{x} \in \mathcal{X}$ is the feature vector while $\boldsymbol{y} = (y_1, y_2, \ldots, y_q) \in \{0, 1\}^q$ is a binary label vector indicating whether $\boldsymbol{x}$ is associated with the k-th label ($y_k = 1$) or not ($y_k = 0$).

As reviewed in Section 1, previous approaches tackle the problem of modeling $P(\boldsymbol{y}|\boldsymbol{x})$ in various ways. First order approaches solve the problem by decomposing it into a number of independent tasks through modeling $P(y_k|\boldsymbol{x})\,(1 \le k \le q)$; Second-order approaches solve the problem by considering interactions between a pair of labels through modeling $P((y_k, y_{k'})|\boldsymbol{x})$ $(k \ne k')$; High-order approaches solve the problem by addressing correlations between a subset of labels through modeling $P((y_{k_1}, y_{k_2}, \ldots, y_{k_{q'}})|\boldsymbol{x})\,(q' \le q)$.

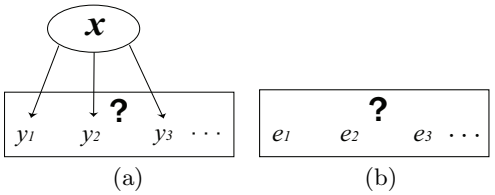

Figure 1: The structures used to encode the conditional dependencies/independencies of the labels. (a) A loyal Bayesian network representation, where all labels have common cause $\boldsymbol{x}$, and we need to identify the links between y_k given $\boldsymbol{x}$. (b) A simplified version, where we first eliminate the effects of $\boldsymbol{x}$ on all labels and find the errors e_k, and then exploit the Bayesian network of the errors e_k.

Our goal is to find a simple and efficient way to improve the performance of multi-label learning by exploiting the label dependencies. In this section we present the basic idea and procedure of such an approach.

2.1 Basic Idea

Mathematically, multi-label learning aims to model and predict $p(\boldsymbol{y}|\boldsymbol{x})$. Our objective is to make use of the conditional dependencies among the labels y_k $(1 \le k \le q)$ such that for each example we can better predict their combination. The problem is how to find and make use of such conditional dependencies in an efficient way. To this end, we adopt the Bayesian network [13] as a compact manner to encode the label dependencies; for simplicity of the representation, we assume that the joint distribution of the labels y_k and the feature set $\mathcal{X}$ factorizes according to some Bayesian network structure, or directed acyclic graph. Note that in multi-label learning, all labels inherently depend on the feature set, therefore, $\boldsymbol{x}$ is the common parent of all labels. Consequently, we have

$$p(\boldsymbol{y}|\boldsymbol{x}) = \prod_{k=1}^{q} p(y_k|\mathbf{pa}_k, \boldsymbol{x}), \tag{1}$$

where $\mathbf{pa}_k$ denotes the set of parents of the label y_k, excluding the inherent parent $\boldsymbol{x}$. In this way, the multi-label classification problem is decomposed into a series of small-scale single-label classification problems.

Fig. 1 (a) describes the relations among all labels y_k, and the feature set $\boldsymbol{x}$ (note that the links among y_k are not given since they are to be found). From this figure one can see that there are two types of dependencies among the labels. One is due to the common parent, i.e., the feature set $\boldsymbol{x}$; because of its effect, labels become dependent even if they are conditionally independent given $\boldsymbol{x}$. The other is the direct dependencies of the labels. One should be aware that the links among y_k given in Fig. 1 (a) may be very different from those implied by the conditional dependencies of y_k without considering the effect of $\boldsymbol{x}$; in fact, the effect of the

common parent $\boldsymbol{x}$ makes learning the relations between y_k complicate.

Generally speaking, there exist two kinds of approaches to Bayesian network structure learning [13]. One is constraint-based, and the other is score-based. Constraint-based approaches exploit (conditional) independence relations between the variables to construct the causal structure. When performing conditional independence tests in such approaches, one usually assumes that the variables are either discrete or jointly Gaussian with linear relations.[1] Score-based approaches view a Bayesian network as specifying a statistical model and then address learning as a model selection problem; they find the Bayesian network structure which maximizes a score function reflecting the goodness of fit and complexity of the model. In our problem, the labels are binary while the features are usually continuous. Moreover, there are usually a large number of features, and the effect of the features on the labels are significantly nonlinear. Consequently, both kinds of approaches mentioned above would encounter difficulties in learning the structure shown in Fig. 1 (a).

2.2 A Practical Approach

2.2.1 DAG's on Errors: To Eliminate the Effect of Features

We then aim to develop a simplified procedure to identify the links between the labels in Fig. 1 (a), with the help of certain reasonable assumptions. To facilitate the following analysis, we consider the binary classification problem as a special case of the nonlinear regression problem:

$$y = f(\boldsymbol{x}) + e, \tag{2}$$

where y denotes the target variable, $\boldsymbol{x}$ the set of predictors, and e the noise. The following proposition shows the relationship between maximizing the data likelihood of this model and minimizing the mutual information between $\boldsymbol{x}$ and the estimate of e.[2]

PROPOSITION 1. *Consider the nonlinear regression model Eq. 2, where f is smooth function. Given the examples $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$, fitting the above model with maximum likelihood is equivalent to minimizing the mutual information between $\boldsymbol{x}$ and the estimate of e.*

Proof of this proposition is given in the Appendix. We view classification as an extreme case of nonlinear regression: in classification, y denotes the target class label (0 or 1), f involves threshold functions, and the error e, which is discrete, may be 0, 1, or -1. $e = 1$ (-1) means that the example, which actually came from class 1 (0), is classified to class 0 (1).

[1]We note that recently, in the causal discovery scenario, a constraint-based method was proposed to find the network structure between a moderate number of continuous variables with nonlinear relations [27]. In principle it can be easily extended to solve our problem; however, due to the computational loads, it is not feasible if the number of labels is large (say, larger than 20).

[2]Mutual information is a canonical measure of dependence [6]. The mutual information amongst a set of variables $v_1, v_2, ..., v_n$ is defined as $I(v_1, ..., v_n) = \sum_{i=1}^n H(v_i) - H(v_1, ..., v_n)$, where $H(\cdot)$ denotes the entropy. Mutual information is always non-negative, and is zero if and only if the involved variables are mutually independent.

For two different classification problems exploiting the same feature set, the following proposition holds straightforwardly.

PROPOSITION 2. *Suppose that we have two classification problems with the same attributes:*

$$y_1 = f_1(\boldsymbol{x}) + e_1 \text{ and } y_2 = f_2(\boldsymbol{x}) + e_2. \tag{3}$$

If (1) both e_1 and e_2 are independent from $\boldsymbol{x}$, and (2) e_1 and e_2 are also independent from each other, then y_1 and y_2 are conditionally independent given $\boldsymbol{x}$.

As an extension of Proposition 1, Condition (1) in Proposition 2, which states that both e_1 and e_2 are independent from $\boldsymbol{x}$, approximately holds. Consequently, roughly speaking, y_1 and y_2 are conditionally independent given the feature set $\boldsymbol{x}$ if and only if e_1 is independent from e_2.

In other words, here we reasonably assume that the effect of $\boldsymbol{x}$ is "separable": we can first eliminate the influences of $\boldsymbol{x}$ in all labels, and then discover the conditional independencies among y_k (conditioned on $\boldsymbol{x}$) by analyzing the errors. The assumption may not always hold rigorously. However, it provides a greatly simplified manner to identify the links between y_k in presence of the common parent $\boldsymbol{x}$ in the network Fig. 1 (a).

2.2.2 Procedure of LEAD

We can then find the links between y_k in the network Fig. 1 (a) in the following way. We first eliminate the effects of the feature set $\boldsymbol{x}$ on all labels by constructing classifiers for all labels and finding the corresponding errors. Then, we find the Bayesian network structure of the errors e_k and treat it as an approximate of that of the labels with $\boldsymbol{x}$ as the common parent. Fig. 1 (b) illustrates this idea. With this Bayesian network, we then find $\mathbf{pa}_k$ for each label y_k in Eq. 1. In our approach, we make use of the links in the Bayesian network structure by directly incorporating $\mathbf{pa}_k$ into the "feature set" when constructing the classifier for y_k.

Our proposed approach consists of the following four steps.

1. Construct the classifiers for all labels independently. This produces the error e_k for each label y_k (Eq. 2).
2. Learn the Bayesian network structure $\mathcal{G}$ of e_k, $1 \leq k \leq q$.
3. For each label y_k, construct the new classifier $\mathcal{C}_k$ by incorporating $\mathbf{pa}_k$ implied in the network $\mathcal{G}$ into the feature set.
4. For testing data, recursively predict y_k with the classifier $\mathcal{C}_k$ and the feature set $\boldsymbol{x} \bigcup \widehat{\mathbf{pa}}_k$ according to the ordering of the labels implied in $\mathcal{G}$.

2.2.3 On Bayesian Network Learning

In Step 2 we need to choose suitable techniques for Bayesian network structure learning. Over 50 software packages are listed in [15] for different applications of Bayesian networks. We used the BDAGL (Bayesian DAG learning) package[3], which implemented the dynamic programming-based algorithm for computing the marginal posterior probability of every edge in a Bayesian network [12]. This algorithm takes $\mathcal{O}(q \cdot 2^q)$ both in time and space, where q is the number of

[3]`http://www.cs.ubc.ca/~murphyk/Software/BDAGL/index.html`

variables. It is very efficient when q is small, and is limited to about 20 variables. (In practice, it takes about 5 seconds for 10 variables to about 5 minutes for 20 variables.) When the number of variables is larger than 20, we resorted to the Banjo (Bayesian ANalysis with Java Objects) package [20]. This package performs approximate maximum a posterior (MAP) structure learning using simulated annealing and hill climbing for searching, and is suitable to analyze large data sets. When using it, one needs to specify the maximum running time and some other necessary parameters, and it will finally report the best network found.

3. EXPERIMENTS

3.1 Evaluation Metrics

Performance evaluation in multi-label learning is much more complicated than traditional single-label learning, as each example is associated with multiple labels simultaneously. One straightforward solution is to calculate the classical single-label metric (such as precision, recall and F-measure) on each possible label independently, and then combine the metric value from each label through *micro-* or *macro*-averaging [23]. However, this intuitive way of evaluation fails to directly address the correlations between different labels of each example.

In this paper, five popular metrics specially designed for multi-label learning [19, 23] are used, i.e. *hamming loss*, *one-error*, *coverage*, *ranking loss* and *average precision*. Given a multi-label data set $\mathcal{S} = \{(\boldsymbol{x}_i, Y_i)|1 \leq i \leq p\}$, the five metrics are defined as below. Here, $h(\boldsymbol{x}_i)$ returns a set of proper labels of $\boldsymbol{x}_i$; $h(\boldsymbol{x}_i, y)$ returns a real-value indicating the confidence for y to be a proper label of $\boldsymbol{x}_i$; $rank^h(\boldsymbol{x}_i, y)$ returns the rank of y derived from $h(\boldsymbol{x}_i, y)$.

- *Hamming loss*:

$$\text{hloss}_{\mathcal{S}}(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|\mathcal{Y}|}|h(\boldsymbol{x}_i)\Delta Y_i| \tag{4}$$

Here Δ denotes the symmetric difference between two sets. The hamming loss evaluates how many times an example-label pair is misclassified.

- *One-error*:

$$\text{one-error}_{\mathcal{S}}(h) = \frac{1}{p}\sum_{i=1}^{p}[\![\,[\arg\max_{y\in\mathcal{Y}} h(\boldsymbol{x}_i, y)] \notin Y_i]\!] \tag{5}$$

Here for predicate π, $[\![\pi]\!]$ equals 1 if π holds and 0 otherwise. The one-error evaluates how many times the top-ranked label is not in the set of proper labels of the example.

- *Coverage*:

$$\text{coverage}_{\mathcal{S}}(h) = \frac{1}{p}\sum_{i=1}^{p}\max_{y\in Y_i} rank^h(\boldsymbol{x}_i, y) - 1 \tag{6}$$

The coverage evaluates how many steps are need, on average, to move down the label list in order to cover all the proper labels of the example.

- *Ranking loss*:

$$\text{rloss}_{\mathcal{S}}(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i||\bar{Y}_i|}\cdot|\mathcal{R}_i|, \quad \text{where}$$

$$\mathcal{R}_i = \{(y_1, y_2)|h(\boldsymbol{x}_i, y_1) \leq h(\boldsymbol{x}, y_2),\ (y_1, y_2) \in Y_i \times \bar{Y}_i\} \tag{7}$$

Here $\bar{Y}_i$ denotes the complementary set of Y_i in $\mathcal{Y}$. The ranking loss evaluates the average fraction of label pairs that are misordered for the example.

- *Average precision*:

$$\text{avgprec}_{\mathcal{S}}(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i|}\cdot\frac{|\mathcal{P}_i|}{rank^h(\boldsymbol{x}_i, y))}, \quad \text{where}$$

$$\mathcal{P}_i = \{y'|rank^h(\boldsymbol{x}_i, y') \leq rank^h(\boldsymbol{x}_i, y),\ y' \in Y_i\} \tag{8}$$

The average precision evaluates the average fraction of proper labels ranked above a particular label $y \in Y_i$.

For the first four metrics, the *smaller* the value the better the performance. For average precision, on the other hand, the *larger* the value the better the performance. Furthermore, we choose to normalize the *coverage* metric (Eq. 6) by $|\mathcal{Y}|$ so that all the five metrics vary between [0 1].

3.2 Data Sets

A total of fourteen multi-label data sets are collected for experiments in this paper, whose characteristics are summarized in Table 1. Given a multi-label data set $\mathcal{S} = \{(\boldsymbol{x}_i, Y_i)|\ 1 \leq i \leq p\}$, we use $|\mathcal{S}|$, $dim(\mathcal{S})$, $L(\mathcal{S})$, $F(\mathcal{S})$ to represent the number of examples, number of features, number of possible labels, and feature type respectively. In addition, several multi-label statistics [18, 23] are also shown in the Table:

a) Label cardinality $LCard(\mathcal{S}) = \frac{1}{p}\sum_{i=1}^{p}|Y_i|$, which measures the average number of labels per example;

b) Label density $LDen(|\mathcal{S}|) = \frac{LCard(\mathcal{S})}{L(\mathcal{S})}$, which normalizes $LCard(\mathcal{S})$ by the number of possible labels;

c) Distinct label sets $DL(\mathcal{S}) = |\{Y|\exists\,\boldsymbol{x} : (\boldsymbol{x}, Y) \in \mathcal{S}\}|$, which counts the number of distinct label combinations appeared in the data set;

d) Proportion of distinct label sets $PDL(\mathcal{S}) = \frac{DL(\mathcal{S})}{|\mathcal{S}|}$, which normalizes $DL(\mathcal{S})$ by the number of examples.

As shown in Table 1, seven regular-scale data sets (first part) as well as seven large-scale data sets (second part) are included whose sizes are roughly ordered by $|\mathcal{S}|$. In addition, dimensionality reduction is performed on `rcv1 (subset 1)` to `rcv1 (subset 5)` as well as `tmc2007`, where the top 2% features with highest *document frequency* [26] are retained. To the best of our knowledge, few works on multi-label learning have conducted experimental evaluation across such broad range of data sets. One notable exception is [18] where a total of 12 data sets (6 regular-scale, 6 large-scale) are considered. Further details on these data sets are available at different sites.[4]

Intuitively, for the data whose underlying joint label dependence could be well represented by a DAG with the feature vector as a common parent, learning with LEAD would give excellent performance.

3.3 Experimental Results

In this paper, we compare LEAD with several state-of-the-art multi-label learning methods, including two *first-order* approaches BSVM [1] and ML-KNN [29], one *second-order* approach BP-MLL [28] and one *high-order* approach ECC [18]. For fair comparison, Libsvm (with linear kernel) [2] is employed as the base classifier for LEAD, BSVM and ECC.

[4] `More multi-label data sets could be found at http://mulan.sourceforge.net/datasets.html, http://www.cs.waikato.ac.nz/~jmr30/`

Table 1: Characteristics of the experimental data sets.

Data set	$\|\mathcal{S}\|$	$dim(\mathcal{S})$	$L(\mathcal{S})$	$F(\mathcal{S})$	$LCard(\mathcal{S})$	$LDen(\mathcal{S})$	$DL(\mathcal{S})$	$PDL(\mathcal{S})$	Domain
emotions	593	72	6	numeric	1.869	0.311	27	0.046	music
genbase	662	1185	27	nominal	1.252	0.046	32	0.048	biology
medical	978	1449	45	nominal	1.245	0.028	94	0.096	text
enron	1702	1001	53	nominal	3.378	0.064	753	0.442	text
image	2000	294	5	numeric	1.236	0.247	20	0.010	media
scene	2407	294	6	numeric	1.074	0.179	15	0.006	media
yeast	2417	103	14	numeric	4.237	0.303	198	0.082	biology
rcv1 (subset 1)	6000	944	101	numeric	2.880	0.029	1028	0.171	text
rcv1 (subset 2)	6000	944	101	numeric	2.634	0.026	954	0.159	text
rcv1 (subset 3)	6000	944	101	numeric	2.614	0.026	939	0.157	text
rcv1 (subset 4)	6000	944	101	numeric	2.484	0.025	816	0.136	text
rcv1 (subset 5)	6000	944	101	numeric	2.642	0.026	946	0.158	text
bibtex	7395	1836	159	nominal	2.402	0.015	2856	0.386	text
tmc2007	28596	981	22	nominal	2.158	0.098	1341	0.047	text

Table 2: Performance (mean±std.) of each algorithm in terms of *hamming loss*. •/◦ indicates whether LEAD is statistically superior/inferior to the compared algorithm (pairwise t-test at 5% significance level).

	Algorithm				
Data Set	LEAD	BSVM	ML-KNN	BP-MLL	ECC
emotions	0.197±0.024	0.199±0.022	0.194±0.013	0.219±0.021•	0.192±0.021
genbase	0.001±0.001	0.001±0.001	0.005±0.002•	0.004±0.002•	0.001±0.001
medical	0.010±0.001	0.010±0.001	0.016±0.002•	0.019±0.002•	0.010±0.001
enron	0.050±0.003	0.060±0.003•	0.052±0.002•	0.052±0.003•	0.055±0.004•
image	0.173±0.011	0.176±0.007	0.170±0.008	0.253±0.024•	0.180±0.015•
scene	0.098±0.005	0.104±0.006	0.084±0.008◦	0.282±0.014•	0.096±0.010◦
yeast	0.202±0.011	0.199±0.010	0.195±0.011◦	0.205±0.010•	0.208±0.010•
rcv1 (subset 1)	0.027±0.001	0.026±0.001◦	0.027±0.001•	0.033±0.001•	0.033±0.003•
rcv1 (subset 2)	0.023±0.001	0.023±0.001	0.024±0.001•	0.028±0.001•	0.029±0.002•
rcv1 (subset 3)	0.023±0.001	0.023±0.001	0.023±0.001	0.028±0.001•	0.029±0.003•
rcv1 (subset 4)	0.020±0.001	0.020±0.001	0.021±0.001•	0.025±0.001•	0.025±0.002•
rcv1 (subset 5)	0.023±0.001	0.023±0.001	0.024±0.001•	0.029±0.001•	0.028±0.002•
bibtex	0.013±0.001	0.016±0.001•	0.014±0.001•	0.016±0.001•	0.016±0.001•
tmc2007	0.063±0.001	0.063±0.001	0.073±0.001•	0.098±0.006•	0.064±0.001•

Furthermore, parameters suggested in respective literatures are used for the compared algorithms: For BSVM, models are learned via the cross-training strategy [1]; For ML-KNN, the number of nearest neighbors considered is set to 10 and Euclidean distance is used as the distance measure [29]; For BP-MLL, the number of hidden neurons is set to 20% of the dimensionality and the number of training epochs is set to 100 [28]; For ECC, the ensemble size is set to 10 and sampling ratio is set to 67% [18].

Ten-fold cross-validation is performed on each experimental data set, where Tables 2 to 6 report the detailed results in terms of different evaluation metrics. On each data set, the mean metric value as well as the standard deviation of each algorithm is recorded. Furthermore, to statistically measure the significance of performance difference, pairwise t-tests at 5% significance level are conducted between the algorithms. Specifically, whenever LEAD achieves significantly better/worse performance than the compared algorithm on any data set, a win/loss is counted and a maker •/◦ is shown in the Table. Otherwise, a tie is counted and no marker is given. The resulting win/tie/loss counts for LEAD against the compared algorithms are summarized in Tables 7 and 8, grouped by $|\mathcal{S}|$ and $L(\mathcal{S})$ respectively.

As shown in Table 7, for data sets with regular number of examples ($|\mathcal{S}| < 5000$), LEAD is significantly superior to the compared algorithms in 31.4% (BSVM), 31.4% (ML-KNN), 68.6% (BP-MLL) and 54.3% (ECC) cases, and is inferior to them in much less 0.0% (BSVM), 17.1% (ML-KNN), 8.6% (BP-MLL) and 17.1% (ECC) cases; Furthermore, for data sets with large number of examples ($|\mathcal{S}| > 5000$), LEAD is significantly superior to the compared algorithms in 57.1% (BSVM), 97.1% (ML-KNN), 91.4% (BP-MLL) and 82.9% (ECC) cases, and is inferior to them in much less 5.7% (BSVM), 0.0% (ML-KNN), 8.6% (BP-MLL) and 5.7% (ECC) cases. These results indicate that LEAD is highly competitive to the state-of-the-art approaches, especially on data sets with large number of examples.

As shown in Table 8, for data sets with regular num-

Table 3: Performance (mean±std.) of each algorithm in terms of *one-error*. •/○ indicates whether LEAD is statistically superior/inferior to the compared algorithm (pairwise t-test at 5% significance level).

	Algorithm				
Data Set	LEAD	BSVM	ML-KNN	BP-MLL	ECC
emotions	0.248±0.071	0.253±0.070	0.263±0.067	0.318±0.057•	0.216±0.085
genbase	0.002±0.005	0.002±0.005	0.009±0.011	0.000±0.000	0.000±0.000
medical	0.139±0.044	0.151±0.054	0.252±0.045•	0.327±0.057•	0.099±0.034○
enron	0.283±0.041	0.308±0.050•	0.313±0.035	0.237±0.038○	0.212±0.026○
image	0.313±0.026	0.314±0.021	0.320±0.026	0.600±0.079•	0.289±0.026○
scene	0.264±0.024	0.250±0.027	0.219±0.029○	0.821±0.031•	0.226±0.034○
yeast	0.235±0.025	0.230±0.023	0.228±0.029	0.235±0.030	0.176±0.022○
rcv1 (subset 1)	0.435±0.016	0.396±0.013○	0.548±0.018•	0.714±0.017•	0.441±0.028
rcv1 (subset 2)	0.411±0.016	0.407±0.018	0.521±0.018•	0.619±0.020•	0.413±0.030
rcv1 (subset 3)	0.421±0.014	0.477±.0127	0.519±0.024•	0.639±0.017•	0.428±0.039
rcv1 (subset 4)	0.358±0.019	0.391±0.082	0.457±0.022•	0.625±0.020•	0.377±0.027•
rcv1 (subset 5)	0.404±0.022	0.432±0.090	0.499±0.029•	0.718±0.019•	0.408±0.044
bibtex	0.404±0.013	0.444±0.011•	0.589±0.019•	0.431±0.024•	0.341±0.022○
tmc2007	0.226±0.011	0.225±0.010	0.308±0.012•	0.444±0.050•	0.176±0.009○

Table 4: Performance (mean±std.) of each algorithm in terms of *coverage*. •/○ indicates whether LEAD is statistically superior/inferior to the compared algorithm (pairwise t-test at 5% significance level).

	Algorithm				
Data Set	LEAD	BSVM	ML-KNN	BP-MLL	ECC
emotions	0.292±0.022	0.295±0.027	0.300±0.019	0.300±0.022	0.322±0.022•
genbase	0.019±0.015	0.011±0.005	0.021±0.013	0.025±0.012	0.013±0.007
medical	0.039±0.017	0.047±0.011•	0.060±0.025•	0.047±0.024•	0.071±0.023•
enron	0.232±0.016	0.425±0.037•	0.247±0.014•	0.204±0.012○	0.387±0.032•
image	0.184±0.007	0.189±0.021	0.194±0.020	0.343±0.029•	0.199±0.020•
scene	0.087±0.007	0.089±0.009	0.078±0.010○	0.374±0.024•	0.091±0.008•
yeast	0.455±0.019	0.514±0.018•	0.447±0.014	0.456±0.019	0.516±0.015•
rcv1 (subset 1)	0.124±0.006	0.219±0.008•	0.219±0.010•	0.222±0.010•	0.353±0.018•
rcv1 (subset 2)	0.108±0.007	0.206±0.010•	0.203±0.012•	0.250±0.010•	0.350±0.018•
rcv1 (subset 3)	0.112±0.006	0.207±0.010•	0.202±0.010•	0.262±0.005•	0.340±0.015•
rcv1 (subset 4)	0.095±0.008	0.187±0.010•	0.176±0.007•	0.245±0.010•	0.302±0.016•
rcv1 (subset 5)	0.106±0.007	0.200±0.011•	0.198±0.010•	0.229±0.008•	0.342±0.013•
bibtex	0.159±0.007	0.226±0.010•	0.340±0.008•	0.096±0.005○	0.347±0.011•
tmc2007	0.135±0.002	0.135±0.003	0.183±0.004•	0.268±0.021•	0.239±0.008•

ber of labels ($L(\mathcal{S}) < 50$), LEAD is significantly superior to the compared algorithms in 17.1% (BSVM), 34.3% (ML-KNN), 80.0% (BP-MLL) and 54.3% (ECC) cases, and is inferior to them in much less 0.0% (BSVM), 17.1% (ML-KNN), 0.0% (BP-MLL) and 17.1% (ECC) cases; Furthermore, for data sets with large number of labels ($L(\mathcal{S}) > 50$), LEAD is significantly superior to the compared algorithms in 71.4% (BSVM), 97.1% (ML-KNN), 80.0% (BP-MLL) and 82.9% (ECC) cases, and is inferior to them in much less 5.7% (BSVM), 0.0% (ML-KNN), 11.4% (BP-MLL) and 5.7% (ECC) cases. In general, correlations among labels would be complex when the label space becomes larger. Therefore, it is very attracting that LEAD gains greater advantages over the comparing algorithms when there is large number class labels, which validates LEAD's effectiveness in exploiting label dependency to facilitate multi-label learning.

4. CONCLUSION

In this paper, a novel approach to multi-label learning is proposed by exploiting the dependencies among labels. Specifically, Bayesian networks are employed to represent the joint distribution of the label space conditioned on the feature space, which is capable of modeling arbitrary order of label correlations. We present an efficient way to approximately find such networks, by working on the classification errors of all labels, instead of of the original labels. The learning system involves a complexity linear in the number of possible labels. Experiments over a broad range of data sets show that our method is highly comparable to the state-of-the-art approaches, especially on learning tasks with large number of labels as well as examples. Due to its accuracy and efficiency, LEAD is expected to be a practically appealing multi-label learning method for large-scale problems.

In the future, we will explore if there exist better ways

Table 5: Performance (mean±std.) of each algorithm in terms of *ranking loss*. •/○ indicates whether LEAD is statistically superior/inferior to the compared algorithm (pairwise t-test at 5% significance level).

	Algorithm				
Data Set	LEAD	BSVM	ML-KNN	BP-MLL	ECC
emotions	0.154±0.029	0.156±0.034	0.163±0.022	0.173±0.020•	0.233±0.040•
genbase	0.005±0.008	0.001±0.002	0.006±0.006	0.008±0.006	0.008±0.008
medical	0.024±0.016	0.032±0.012•	0.042±0.021•	0.032±0.018•	0.098±0.032•
enron	0.084±0.008	0.180±0.022•	0.093±0.007•	0.068±0.006	0.241±0.025•
image	0.164±0.018	0.169±0.019	0.175±0.019	0.366±0.037•	0.245±0.024•
scene	0.087±0.009	0.089±0.011	0.076±0.012○	0.434±0.026•	0.135±0.013•
yeast	0.172±0.015	0.200±0.013•	0.166±0.015	0.171±0.015	0.285±0.022•
rcv1 (subset 1)	0.051±0.003	0.097±0.004•	0.105±0.005•	0.115±0.006•	0.382±0.025•
rcv1 (subset 2)	0.046±0.003	0.096±0.005•	0.100±0.007•	0.152±0.007•	0.377±0.031•
rcv1 (subset 3)	0.049±0.002	0.097±0.006•	0.100±0.006•	0.166±0.002•	0.368±0.020•
rcv1 (subset 4)	0.040±0.003	0.091±0.004•	0.083±0.005•	0.155±0.006•	0.317±0.026•
rcv1 (subset 5)	0.043±0.003	0.091±0.008•	0.095±0.005•	0.118±0.004•	0.369±0.025•
bibtex	0.086±0.005	0.127±0.006•	0.209±0.006•	0.051±0.003○	0.411±0.013•
tmc2007	0.055±0.002	0.054±0.002	0.089±0.003•	0.147±0.015•	0.179±0.006•

Table 6: Performance (mean±std.) of each algorithm in terms of *average precision*. •/○ indicates whether LEAD is statistically superior/inferior to the compared algorithm (pairwise t-test at 5% significance level).

	Algorithm				
Data Set	LEAD	BSVM	ML-KNN	BP-MLL	ECC
emotions	0.811±0.035	0.807±0.037	0.799±0.031	0.779±0.027•	0.796±0.042•
genbase	0.994±0.008	0.998±0.004	0.989±0.010•	0.988±0.010•	0.994±0.006
medical	0.890±0.037	0.871±0.047•	0.806±0.036•	0.782±0.042•	0.872±0.033•
enron	0.663±0.022	0.591±0.035•	0.626±0.022•	0.705±0.025○	0.640±0.025•
image	0.799±0.017	0.796±0.015	0.792±0.017	0.601±0.040•	0.794±0.016
scene	0.848±0.014	0.849±0.016	0.869±0.017○	0.445±0.018•	0.852±0.016
yeast	0.761±0.020	0.749±0.019•	0.765±0.021	0.754±0.020•	0.728±0.019•
rcv1 (subset 1)	0.600±0.009	0.588±0.008•	0.478±0.011•	0.388±0.011•	0.475±0.020•
rcv1 (subset 2)	0.641±0.010	0.612±0.011•	0.513±0.012•	0.389±0.011•	0.498±0.014•
rcv1 (subset 3)	0.629±0.011	0.576±0.054•	0.523±0.013•	0.388±0.009•	0.499±0.018•
rcv1 (subset 4)	0.683±0.012	0.635±0.036•	0.575±0.016•	0.407±0.014•	0.558±0.017•
rcv1 (subset 5)	0.642±0.016	0.600±0.047•	0.530±0.019•	0.391±0.005•	0.507±0.028•
bibtex	0.537±0.009	0.516±0.010•	0.350±0.011•	0.557±0.013○	0.512±0.013•
tmc2007	0.802±0.005	0.804±0.005	0.726±0.007•	0.603±0.031•	0.768±0.005•

to identify, encode, and make use of the conditional dependencies of the labels with the feature set as the common parent.

5. ACKNOWLEDGMENTS

The authors wish to thank the anonymous reviewers for their invaluable comments. This work is supported by the National Science Foundation of China (60805022), Ph.D. Programs Foundation of Ministry of Education of China for Young Faculties (200802941009), Open Foundation of National Key Laboratory for Novel Software Technology of China (KFKT2008B12).

6. REFERENCES

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

[2] C.-C. Chang and C.-J. Lin. *LIBSVM: A library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm.

[3] W. Cheng and E. Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.

[4] A. Clare and R. D. King. Knowledge discovery in multi-label phenotype data. In L. D. Raedt and A. Siebes, editors, *Lecture Notes in Computer Science 2168*, pages 42–53. Springer, Berlin, 2001.

[5] F. D. Comité, R. Gilleron, and M. Tommasi. Learning multi-label altenating decision tree from texts and data. In P. Perner and A. Rosenfeld, editors, *Lecture Notes in Computer Science 2734*, pages 35–49. Springer, Berlin, 2003.

[6] T. M. Cover and J. A. Thomas. *Elements of*

Table 7: The win/tie/loss results (grouped by $|\mathcal{S}|$) for LEAD against the compared algorithms in terms of different evaluation metrics.

Evaluation	LEAD against							
	BSVM		ML-KNN		BP-MLL		ECC	
Metric	$\|\mathcal{S}\| < 5000$	$\|\mathcal{S}\| > 5000$	$\|\mathcal{S}\| < 5000$	$\|\mathcal{S}\| > 5000$	$\|\mathcal{S}\| < 5000$	$\|\mathcal{S}\| > 5000$	$\|\mathcal{S}\| < 5000$	$\|\mathcal{S}\| > 5000$
hamming loss	**1/6/0**	**1/5/1**	**3/2/2**	**6/1/0**	**7/0/0**	**7/0/0**	**3/3/1**	**7/0/0**
one-error	**1/6/0**	**1/5/1**	**1/5/1**	**7/0/0**	**4/2/1**	**7/0/0**	**0/2/5**	**1/4/2**
coverage	**3/4/0**	**6/1/0**	**2/4/1**	**7/0/0**	**3/3/1**	**6/0/1**	**6/1/0**	**7/0/0**
ranking loss	**3/4/0**	**6/1/0**	**2/4/1**	**7/0/0**	**4/3/0**	**6/0/1**	**6/1/0**	**7/0/0**
average precision	**3/4/0**	**6/1/0**	**3/3/1**	**7/0/0**	**6/0/1**	**6/0/1**	**4/3/0**	**7/0/0**
In Total	**11/24/0**	**20/13/2**	**11/18/6**	**34/1/0**	**24/8/3**	**32/0/3**	**19/10/6**	**29/4/2**

Table 8: The win/tie/loss results (grouped by $L(\mathcal{S})$) for LEAD against the compared algorithms in terms of different evaluation metrics.

Evaluation	LEAD against							
	BSVM		ML-KNN		BP-MLL		ECC	
Metric	$L(\mathcal{S}) < 50$	$L(\mathcal{S}) > 50$	$L(\mathcal{S}) < 50$	$L(\mathcal{S}) > 50$	$L(\mathcal{S}) < 50$	$L(\mathcal{S}) > 50$	$L(\mathcal{S}) < 50$	$L(\mathcal{S}) > 50$
hamming loss	**0/7/0**	**2/4/1**	**3/2/2**	**6/1/0**	**7/0/0**	**7/0/0**	**3/3/1**	**7/0/0**
one-error	**0/7/0**	**2/4/1**	**2/4/1**	**7/0/0**	**5/2/0**	**6/0/1**	**0/2/5**	**1/4/2**
coverage	**2/5/0**	**7/0/0**	**2/4/1**	**7/0/0**	**4/3/0**	**5/2/0**	**6/1/0**	**7/0/0**
ranking loss	**2/5/0**	**7/0/0**	**2/4/1**	**7/0/0**	**5/2/0**	**5/1/1**	**6/1/0**	**7/0/0**
average precision	**2/5/0**	**7/0/0**	**3/3/1**	**7/0/0**	**7/0/0**	**5/0/2**	**4/3/0**	**7/0/0**
In Total	**6/29/0**	**25/8/2**	**12/17/6**	**34/1/0**	**28/7/0**	**28/3/4**	**19/10/6**	**29/4/2**

Information Theory. Wiley-Interscience, New York, NY, 1991.

[7] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 681–687. MIT Press, Cambridge, MA, 2002.

[8] J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73(2):133–153, 2008.

[9] N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 195–200, Bremen, Germany, 2005.

[10] S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In H. Dai, R. Srikant, and C. Zhang, editors, *Lecture Notes in Artificial Intelligence 3056*, pages 22–30. Springer, Berlin, 2004.

[11] S. Ji, L. Tang, S. Yu, and J. Ye. Extracting shared subspace for multi-label classification. In *Proceedings of the 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 381–389, Las Vegas, NV, 2008.

[12] M. Koivisto. Advances in exact bayesian structure discovery in bayesian networks. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 241–248, Menlo Park, CA, 2006. AUAI Press.

[13] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

[14] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *Working Notes of the AAAI'99 Workshop on Text Learning*, Orlando, FL, 1999.

[15] K. Murphy. Software packages for graphical models / bayesian networks. *International Society for Bayesian Analysis*, 2007.

[16] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang. Correlative multi-label video annotation. In *Proceedings of the 15th ACM International Conference on Multimedia*, pages 17–26, Augsburg, Germany, 2007.

[17] J. Read, B. Pfahringer, and G. Holmes. Multi-label classification using ensembles of pruned sets. In *Proceedings of the 9th IEEE International Conference on Data Mining*, pages 995–1000, Pisa, Italy, 2008.

[18] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. In W. Buntine, M. Grobelnik, and J. Shawe-Taylor, editors, *Lecture Notes in Artificial Intelligence 5782*, pages 254–269. Springer, Berlin, 2009.

[19] R. E. Schapire and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[20] V. Smith, J. Yu, T. Smulders, A. Hartemink, and E. Jarvis. Computational inference of neural information flow networks. *PLoS Computational Biology*, 2(11):1436–1449, 2006.

[21] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*. Springer, Berlin, 2010.

[22] G. Tsoumakas and I. Vlahavas. Random k-labelsets: an ensemble method for multilabel classification. In J. N. Kok, J. Koronacki, R. L. de Mantaras, S. Matwin, D. Mladenič, and A. Skowron, editors, *Lecture Notes in Artificial Intelligence 4701*, pages 406–417. Springer, Berlin, 2007.

[23] G. Tsoumakas, M.-L. Zhang, and Z.-H. Zhou. Tutorial on learning from multi-label data [http://www.ecml pkdd2009.net/wp-content/uploads/2009/08/learning-from-multi-label-data.pdf]. In *ECML/PKDD 2009*, Bled, Slovenia, 2009.

[24] N. Ueda and K. Saito. Parametric mixture models for multi-label text. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 721–728. MIT Press, Cambridge, MA, 2003.

[25] R. Yan, J. Tešić, and J. R. Smith. Model-shared subspace boosting for multi-label classification. In *Proceedings of the 13th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 834–843, San Jose, CA, 2007.

[26] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, Nashville, TN, 1997.

[27] K. Zhang and A. Hyvärinen. Causality discovery with additive disturbances: An information-theoretical perspective. In W. Buntine, M. Grobelnik, and J. Shawe-Taylor, editors, *Lecture Notes in Artificial Intelligence 5782*, pages 570–585. Springer, Berlin, 2009.

[28] M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

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

[30] S. Zhu, X. Ji, W. Xu, and Y. Gong. Multi-labelled classification using maximum entropy method. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 274–281, Salvador, Brazil, 2005.

APPENDIX

A. PROOF OF PROPOSITION 1

PROOF. Denote by $\hat{e}$ and $\hat{f}$ the estimate of e and f, respectively. Suppose that the density of the noise, p_e, is given (which may be adaptively estimated from data or fixed to a reasonable prior distribution). The maximum likelihood estimate of f and e is obtained by maximizing the data log-likelihood

$$l = \sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i) = \sum_{i=1}^{N} \log p_e(y_i - \hat{f}(\boldsymbol{x}_i)). \qquad (9)$$

Now let us see how the above quantity is related to $I(\boldsymbol{x}, \hat{e})$, the mutual information between $\boldsymbol{x}$ and the estimate of e. Consider the transformation from $(\boldsymbol{x}, y)^T$ to $(\boldsymbol{x}, \hat{e})^T$. As $\hat{e} = y - \hat{f}(\boldsymbol{x})$, the Jacobian matrix in that transformation is

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}} & \frac{\partial \boldsymbol{x}}{\partial y} \\ (\frac{\partial \hat{e}}{\partial \boldsymbol{x}})^T & \frac{\partial \hat{e}}{\partial y} \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -(\frac{\partial \hat{f}}{\partial \boldsymbol{x}})T & 1 \end{pmatrix},$$

where $\mathbf{I}$ denotes the identity matrix and $\mathbf{0}$ denotes the vector of zeros. Clearly, one can see that the determinant of $\mathbf{J}$ is $|\mathbf{J}| = 1$. Consequently, we have $p(\boldsymbol{x}, \hat{e}) = p(\boldsymbol{x}, y)/|\mathbf{J}| = p(\boldsymbol{x}, y)$. That is, the joint entropy of $(\boldsymbol{x}, \hat{e})$ is

$$H(\boldsymbol{x}, \hat{e}) = -E\{\log p(\boldsymbol{x}, \hat{e})\} = -E\{\log p(\boldsymbol{x}, y)\} = H(\boldsymbol{x}, y).$$

Mutual information between $\boldsymbol{x}$ and $\hat{e}$ is then

$$\begin{aligned} I(\boldsymbol{x}, \hat{e}) &= H(\boldsymbol{x}) + H(\hat{e}) - H(\boldsymbol{x}, \hat{e}) \\ &= H(\boldsymbol{x}) + H(\hat{e}) - H(\boldsymbol{x}, y). \end{aligned}$$

As the first and third terms in the above quantity do not depend on $\hat{f}$, minimizing $I(\boldsymbol{x}, \hat{e})$ is then equivalent to minimizing $H(\hat{e})$, or maximizing $\sum_{i=1}^{N} \log p_e(\hat{e}_i) = \sum_{i=1}^{N} \log p_e(y_i - \hat{f}(\boldsymbol{x}_i))$, which is exactly the log-likelihood given in Eq. 9. One can then see that maximum likelihood is equivalent to minimizing the mutual information between $\boldsymbol{x}$ and $\hat{e}$. □