Neurocomputing 257 (2017) 164–174

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Multi-label classification by exploiting local positive and negative pairwise label correlation

Jun Huang[a,b], Guorong Li[a,*], Shuhui Wang[c], Zhe Xue[a], Qingming Huang[a,c,*]

[a] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 101408, China
[b] School of Computer Science and Technology, Anhui University of Technology, Maanshan 243032, China
[c] Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 4 July 2016
Revised 30 November 2016
Accepted 7 December 2016
Available online 6 February 2017

Keywords:
Multi-label classification
k nearest neighbors
Local label correlation
Positive and negative label correlation

ABSTRACT

In multi-label learning, each example is represented by a single instance and associated with multiple class labels. Existing multi-label learning algorithms mainly exploit label correlations globally, by assuming that the label correlations are shared by all the examples. Moreover, these multi-label learning algorithms exploit the positive label correlations among different class labels. In practical applications, however, different examples may share different label correlations, and the labels are not only positive correlated, but also mutually exclusive with each other. In this paper, we propose a simple and effective Bayesian model for multi-label classification by exploiting Local positive and negative Pairwise Label Correlations, named LPLC. In the training stage, the positive and negative label correlations of each ground truth label for all the training examples are discovered. In the test stage, the *k* nearest neighbors and their corresponding positive and negative pairwise label correlations for each test example are first identified, then we make prediction through maximizing the posterior probability, which is estimated on the label distribution, the local positive and negative pairwise label correlations embodied in the *k* nearest neighbors. A comparative study with the state-of-the-art approaches manifests a competitive performance of our proposed method.

1. Introduction

With the ever-growing amount of digital data in multimedia datab ases, there is a great need for algorithms that can provide effective semantic indexing. Classification on big multimedia data is a great challenging problem. Several issues lead to the difficulty of this problem, including big in volume, unstructured, noisy, redundant, and heterogeneous. What's more, a multimedia data object may describe many concepts simultaneously rather than a single one. In multi-label setting, the examples (objects) can belong to multiple labels (concepts) simultaneously and each example is represented by one single instance. The labels often have correlations with each other, and exploiting label correlation can significantly boost classification performance. The challenge is how to learn a well constructed classification model by exploiting label correlations to more accurately predict a set of possible labels for unseen examples.

To address this issue, a number of algorithms have been proposed for multi-label classification by exploiting *second-order* and *high-order* label correlations [1]. Most of these algorithms try to exploit label correlations globally, by assuming that the label correlations are shared by all the examples. In practical applications, however, different examples may share different label correlations. For example, as shown in Fig. 1, "sand" and "ship" both have strong correlations with label "sea". But these correlations may be different on different groups of images. The correlation between "sand" and "sea" is only shared by the examples like the images annotated as "sand" and "sea" in Fig. 1. While the correlation between "ship" and "sea" is only shared by the examples like the images annotated as "ship" and "sea" in Fig. 1.

On the other hand, exploiting label correlation globally may lead to unnecessary and error predictions [2,3]. For example, suppose there are three labels (e.g., y_1, y_2 and y_3), Fig. 2 shows two simple examples of global label correlation. In Fig. 2(a), labels y_2 and y_3 are dependent on y_1. It can be learned that labels y_2 and y_3 are conditional independent given y_1, thus the predictions of y_2 (e.g., $\Pr(y_2|y_1,x)$) and y_3 (e.g., $\Pr(y_3|y_1,x)$) will be the same both under global and local situation, respectively. If a new test example is only associated with labels y_1 and y_2, however, label y_3 (*an irrelevant label to the new test example*) will also be assigned to the

* Corresponding author at: School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 101408, China.
E-mail addresses: huangjun13b@mails.ucas.ac.cn (J. Huang), liguorong@ucas.ac.cn, grli@jdl.ac.cn (G. Li), wangshuhui@ict.ac.cn (S. Wang), xuezhe10@mails.ucas.ac.cn (Z. Xue), qmhuang@ucas.ac.cn (Q. Huang).

http://dx.doi.org/10.1016/j.neucom.2016.12.073
0925-2312/

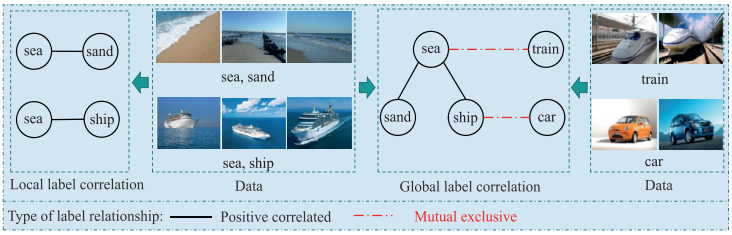


Fig. 1. Image classification: exploiting label correlations.

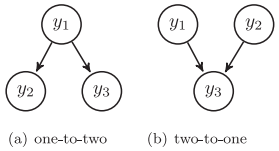


Fig. 2. Two simple examples of global label correlation.

new test example with a higher probability under the global label correlation situation. In Fig. 2(b), label y_3 is dependent on y_1 and y_2 globally. In this situation, the incorrect prediction of label y_1 or y_2 will propagate to the prediction of label y_3 (e.g., $\Pr(y_3|y_1, y_2, x)$). The above problems will be worse if the label dependency is "one-to-many" or "many-to-one". However, these negative impacts will be alleviated if the local label dependency structures for different examples are appropriately exploited. ML-LOC [2] and GCC [3] exploit label correlations locally, and yield good results. These algorithms only exploit the positive label correlations, i.e. an example that belongs to one label is also likely to belong to another. Such positive correlation among labels is often exploited by the co-occurrence between the label pairs [1]. However, the negative correlation, i.e. belonging to one label can indicate an example less likely to belong to another one, or not belonging to one label can indicate an example belong to another label with a higher probability [4], is often neglected. Thus, if two labels are negative correlated, they might be mutually exclusive. For example, as shown in Fig. 1, the correlations between "sea" and "train", and "ship" and "car". These pairs of labels might be not co-occurred with each other with a higher probability, and incorporating these negative correlations may boost the performance of multi-label classifiers.

In this paper, we propose a simple and effective Bayesian model for multi-label classification by exploiting local positive and negative pairwise label correlation simultaneously, named LPLC, i.e. multi-label classification by exploiting **L**ocal positive and negative **P**airwise **L**abel **C**orrelation. We exploit the positive and negative correlated class labels for each ground truth label of each training instance. In our method, for each instance, we assume that the positive label correlation only exists between any two ground truth labels, and the negative correlation only exists between a ground truth label and a label it does not have. The positive and negative correlation for each training example is exploited by computing a maximum conditional probability.

To be more specific, LPLC exploits the local positive and negative pairwise label correlations within two stages. First, in the training stage, we discover the positive and negative label correlations of each ground truth label for all the training examples. Second, in the test stage, we first identify the k nearest neighbors and their corresponding positive and negative pairwise label correlations for each test example, then we make prediction through maximizing the posterior probability, which is estimated on the label distribution, the local positive and negative pairwise label correlations of the k nearest neighbors. Our major contributions are summarized as follows:

- We propose to model local positive and negative pairwise label correlation for multi-label classification. To the best of our knowledge, this is the first work which explicitly exploits the positive and negative label correlations simultaneously.
- We propose a simple and effective Bayesian model for multi-label classification based on the kNN algorithm, and make prediction through maximizing the posterior probability, which is estimated on the label distribution, the local positive and negative pairwise label correlations of the k nearest neighbors.
- The experimental results on twelve multi-label benchmark data sets show the effectiveness of our method against the state-of-the-art multi-label classification algorithms.

The rest of this paper is organized as follows. Section 2 reviews previous works on multi-label learning. Section 3 presents details of the proposed method LPLC. Experimental results and analysis on twelve multi-label benchmark data sets are shown in Section 4. Finally, we conclude our paper in Section 5.

2. Related work

Multi-label learning has attracted significant attention from researchers. Unlike traditional single-instance single-label learning, multi-label learning deals with examples having multiple class labels simultaneously and each example is represented by one single instance. Multi-label learning has been applied to a variety of domains, such as text categorization [5–7], image annotation [8–11], video annotation [12,13], social networks [14], music emotion categorization [15,16]. The challenge is how to learn a well constructed classification model which can predict a set of possible labels for unseen examples.

In the past decades, many well-established methods have been proposed to solve multi-label learning problems in various domains. According to[1,17], the multi-label learning algorithms can be divided into two categories, including problem transformation methods (fitting data to algorithm) and algorithm adaption methods (fitting algorithm to data). Problem transformation methods transform the multi-label classification problem into either one or more single-label classification (binary classification, multi-class classification) problems, e.g., Label Powerset (LP) [17] and Binary Relevance (BR) [18]. Algorithm Adaption Methods modify traditional single-label classification algorithms for multi-label learning, which can handle multi-label data directly, such as the algorithms constructed on kNN [19–24], decision tree [25,26], neural network [5,27], support vector machines [2,28].

In multi-label learning, labels often have correlations with each other. A multitude of algorithms have been proposed for multi-label classification by exploiting the correlations among class labels to improve the performance of multi-label classification. Based on the way of label correlations being considered, existing algorithms can be grouped into three major categories, i.e., *first-order, second-order* and *high-order* algorithms.

First-order algorithms tackle multi-label classification problem by decomposing it into a number of independent binary classification problems without considering label correlations. BR [18] is

a first-order multi-label classification algorithm. It is a representative algorithm of problem transformation methods, and the basic idea of BR is to decompose a multi-label learning problem into q independent binary (one-vs-rest) classification problems, where q is the number of labels of a multi-label data set. ML-kNN [22] is a first-order and algorithm adaption multi-label learning algorithm which is derived from traditional kNN algorithm. The basic idea of ML-kNN is to adapt k-nearest neighbor techniques to deal with multi-label data directly. For each new test example, its k nearest neighbors in the training data are firstly identified. Then, the maximum a posteriori (MAP) rule is utilized to make prediction by reasoning with the labeling information embodied in the neighbors.

Second-order approaches tackle multi-label learning problem by mining *pairwise* relationships between class labels. One way to consider pairwise relationship is to incorporate the criterion of ranking loss into the objective function to be optimized by learning the classification models, e.g., RankSVM [28] and BP-MLL [5]. Another way to model pairwise correlation is to exploit the co-occurrence patterns between label pairs, such as CLR [29] and LLSF [30]. These algorithms only exploit the positive correlations between label pairs. However, in real applications, the class label might be dependent on more than one class labels. Moreover, the class labels might be negative correlated (or mutually exclusive) with each other, i.e. belonging to one label can make an example less likely to belong to the other label, or not belonging to one label can make an example more likely to belong to the other label.

High-order approaches tackle multi-label learning problem by ming relationships between all the class labels or subsets of labels. Classifier Chains (CC) [31] is a novel chain algorithm which models *high-order* label correlations by using the vector of class labels as additional example attributes. It transforms the multi-label classification problem into a chain of q binary classification problems, and the i-th classifier h_i is trained by using the results of labels $y_1, y_2, \ldots, y_{i-1}$ as additional input information. The training stage can be paralleled, while prediction for new instance is worked one by one. To predict subsequent labels in a given chain order, CC resorts to using outputs of the preceding classifiers, which makes them prone to errors. The performance of CC is seriously constrained by the training order of labels and error propagation. Besides, it may not be appropriate that each label is dependent on all the preceding labels in a given order of labels. Probabilistic Classifier Chains (PCC) [32] is an extended work on CC by formulating a probabilistic interpretation. PCC suffers from the computational issue that the inference (i.e. predicting the label of an example) requires time exponential in the number of class labels. Moreover, the performance of PCC is sensitive to the order of class labels while training. There are several extended works on CC and PCC by searching suitable order of labels or dependent structure between labels and reducing the computational complexity, such as HIROM [33], PruDent [34], LLSF-DL [35], BCC [36], PCC-beam [37], MCC [38], PACC [39], EMHG [40], and MIML-ECC [41].

Existing multi-label learning approaches mainly exploit label correlations globally, by assuming that the label correlations are shared by all the instances. In real-world tasks, however, different examples may share different label correlations, and few correlations are globally applicable. There are several works on exploiting label correlations locally, such as ML-LOC [2] and GCC [3]. ML-LOC [2] exploits label correlations locally for multi-label learning. It assumes that the instances can be separated into different groups and each group shares a subset of label correlations. To encode the local influence of label correlations, it constructs a LOC (LOcal Correlation) code for each instance and use this code as additional features for the instance. The classifier is trained with original features and LOC codes. For test examples, LOC codes are unknown and regression models are trained to predict their LOC codes. However, it is difficult to understand the semantic connection between LOC codes and the local label dependency structures. GCC [3] is a group sensitive classifier chains multi-label classifier. GCC first partition the data set into groups by clustering, and then learns a directed label correlation dependency graph on each cluster. The group specific classifier chains could be built based on the examples in each cluster and the corresponding label-dependency graph.

3. The proposed method

In this section, details of the proposed method LPLC will be presented. First, we will present how to model the positive and negative correlated local pairwise label correlations between labels, and then give the details of the proposed model LPLC and the probability estimation for it.

3.1. Preliminaries

In multi-label learning, suppose $\mathcal{X} = \mathbb{R}^d$ be an input space with d-dimensional and $\mathcal{Y} = \{y_1, y_2, \ldots, y_q\}$ be a finite set of q possible labels. $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) | 1 \le i \le n\}$ is a training data set with n examples. The ith example is denoted by a vector with d attribute values $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]$, $\mathbf{x}_i \in \mathcal{X}$, and $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]$ is a set of possible labels for $\mathbf{x}_i$. Each element $y_{ij} = 1$ if the label y_j is associated with x_i, otherwise $y_{ij} = 0$.

$\mathcal{N}(\mathbf{x})$ indicates the k nearest neighbors for example $\mathbf{x}$, and k is the number of nearest neighbors. The similarity between two examples is calculated according to Euclidean distance in this paper. Let $\mathbf{c}_{\mathbf{x}}^{j} = [c_{\mathbf{x}}^{1j}, c_{\mathbf{x}}^{2j}, ..., c_{\mathbf{x}}^{kj}]^T$ be a k-dimensional column vector indicating the membership to label y_j of the examples in $\mathcal{N}(\mathbf{x})$, the ith component of which indicates whether the ith example amongst the k-NNs of $\mathbf{x}$ belongs to class y_j, i.e. $c_{\mathbf{x}}^{ij} = y_{ij}, \forall\ \mathbf{x}_i \in \mathcal{N}(\mathbf{x})$. All the symbols used in this paper and their definitions are summarized in Table 1.

3.2. Local positive and negative pairwise label correlations

In multi-label learning, class labels often have correlations with each other, including positive and negative correlations. For example, an example that belongs to label y_1 is also likely to belong to label y_2, thus the correlation between y_1 and y_2 is positive. Conversely, if not belonging to label y_1 indicates that an example belongs to label y_3 with a higher probability, we say that y_1 and y_3 are negatively correlated. Exploiting the positive and negative correlations among class labels may improve the performance of multi-label classification.

Considering different examples may share different label correlations, in this paper, we try to exploit the positive and negative pairwise label correlations locally. We define two matrices $\mathbf{P} \in \mathbb{R}^{n \times q}$ and $\mathbf{N} \in \mathbb{R}^{n \times q}$. $\mathbf{P}$ stores the most positive correlated label for each ground truth label to which each training example $\mathbf{x}_i$ belongs, i.e. if $y_{ij} = 1$, $\mathbf{P}_{ij}$ stores the index of the most positive correlated class label (e.g., y_c) of y_j for $\mathbf{x}_i$, then $\mathbf{P}_{ij} = c$, otherwise, $\mathbf{P}_{ij} = 0$. If there are multiple most positive correlated class labels with y_j, the label with the largest value of $|\mathbf{c}_{\mathbf{x}_i}^{c}|_1$ will be selected. $\mathbf{N}$ stores the most negative correlated label for each ground truth label to which each training example $\mathbf{x}_i$ belongs, i.e. if $y_{ij} = 1$, $\mathbf{N}_{ij}$ stores the index of the most negative correlated class label (e.g., y_c) of y_j for x_i, then $\mathbf{N}_{ij} = c$, otherwise, $\mathbf{N}_{ij} = 0$. If there are multiple most negative correlated class labels with y_j, the label with the largest value of $|\bar{\mathbf{c}}_{\mathbf{x}_i}^{c}|_1$ will be selected.

3.2.1. Discovering local positive label correlations

For the positive label correlations, if class label y_l is associated with example $\mathbf{x}_i$, y_j might also be associated with $\mathbf{x}_i$ with a higher probability. Therefore, the annotation of y_l can help the annotation

Table 1
Important notations.

Symbol	Definition
k	the number of nearest neighbors
$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]$	the feature vector for the ith training example
$\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]$	the vector of variables for the label set of example $\mathbf{x}_i$, and $y_{ij} \in \{0, 1\}$, $1 \le j \le q$
$\bar{\mathbf{y}}_i = [\bar{y}_{i1}, \bar{y}_{i2}, \ldots, \bar{y}_{iq}]$	$\bar{y}_{ij} = 1 - y_{ij}, 1 \le j \le q$
$\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}$	a training data set with n examples
$\mathcal{N}(\mathbf{x})$	a subset of the training data set composed of the k nearest neighbors of example $\mathbf{x}$
$\mathbf{c}_{\mathbf{x}}^{j} = [c_{\mathbf{x}}^{1j}, c_{\mathbf{x}}^{2j}, \ldots, c_{\mathbf{x}}^{kj}]^T$	the vector of variables for the membership of the examples $\mathbf{x}_i (1 \le i \le k)$ to label y_j, $\mathbf{x}_i \in \mathcal{N}(\mathbf{x})$, $c_{\mathbf{x}}^{ij} \in \{0, 1\}$ and $c_{\mathbf{x}}^{ij} = y_{ij}, 1 \le i \le k$
$\bar{\mathbf{c}}_{\mathbf{x}}^{j} = [\bar{c}_{\mathbf{x}}^{1j}, \bar{c}_{\mathbf{x}}^{2j}, \ldots, \bar{c}_{\mathbf{x}}^{kj}]^T$	$\bar{c}_{\mathbf{x}}^{ij} = 1 - c_{\mathbf{x}}^{ij}, 1 \le i \le k$
$\mathbf{P} \in \mathbb{R}^{n \times q}$	$\mathbf{P}$ stores the index of the most positive correlated label for each ground truth label to which each training example $\mathbf{x}_i$ belongs
$\mathbf{N} \in \mathbb{R}^{n \times q}$	$\mathbf{N}$ stores the index of the most negative correlated label for each ground truth label to which each training example $\mathbf{x}_i$ belongs
$\mathcal{NC}_P(\mathbf{x}_t)$	the positive pairwise label correlations shared by the examples in $\mathcal{N}(\mathbf{x}_t)$
$\mathcal{NC}_N(\mathbf{x}_t)$	the negative pairwise label correlations shared by the examples in $\mathcal{N}(\mathbf{x}_t)$
$\mathbf{pa}_i^+(\mathbf{x}_t)$	the set of positive correlated labels of label y_i in $\mathcal{NC}_P(\mathbf{x}_t)$
$\mathbf{pa}_i^-(\mathbf{x}_t)$	the set of negative correlated labels of label y_i in $\mathcal{NC}_N(\mathbf{x}_t)$

of y_j. In our proposed model, the most positive correlated class label of y_j for $\mathbf{x}_i$ is explored according to the posterior probability of label y_j given by another ground truth label of example $\mathbf{x}_i$ and the k nearest neighbors $\mathcal{N}(\mathbf{x}_i)$, where the posterior probability is calculated by Eq. (1),

$$\mathbf{P}_{ij} = \arg\max_{\forall\ l:l\neq j\ \&\ y_{il}=1} p(y_j = 1 | y_l = 1, \mathcal{N}(\mathbf{x}_i)) \tag{1}$$

If two class labels are strongly correlated, the posterior probability will be large; otherwise, it will be small. We assume that the strong positive pairwise label correlation only exists among the ground truth labels to which each training example belongs. When calculating $\mathbf{P}_{ij}$ by Eq. (1), y_l and y_j should be associated with $\mathbf{x}_i$, i.e. $y_{il} = 1$ and $y_{ij} = 1$. Because, if two labels are strongly correlated, they must be often co-occurred with each other, and thus the value of Eq. (1) will be larger. This constraint will make our algorithm exploit local pairwise label correlation efficiently. It is worth noting that $\mathbf{P}_{ij}$ may not equal to $\mathbf{P}_{il}$, i.e label y_j may dependent on y_l, but y_l may not dependent on y_j. For example, in image annotation, if we know one image is annotated as "ship", it will be annotated as "sea" with a higher probability, but not the vice versa.

If label y_l is chosen as the strongest positive correlated class label of label y_j for $\mathbf{x}_i$, it results in a maximum conditional probability of $p(y_j = 1 | y_l = 1, \mathcal{N}(\mathbf{x}_i))$. The probability $p(y_j = 1 | y_l = 1, \mathcal{N}(\mathbf{x}_i))$ is calculated by Eq. (2),

$$p(y_j = 1 | y_l = 1, \mathcal{N}(\mathbf{x}_i)) = \frac{\mathbf{c}_{\mathbf{x}_i}^{j\,T} \mathbf{c}_{\mathbf{x}_i}^{l}}{\|\mathbf{c}_{\mathbf{x}_i}^{l}\|_1} \tag{2}$$

Thus, the probability $p(y_j = 0 | y_l = 1, \mathcal{N}(\mathbf{x}_i)) = 1 - p(y_j = 1 | y_l = 1, \mathcal{N}(\mathbf{x}_i))$.

3.2.2. Discovering local negative label correlations

For the negative label correlations, if class label y_l is not associated with example $\mathbf{x}_i$, then y_j might be associated with $\mathbf{x}_i$ with a higher probability. Therefore, this negative label correlation can help the annotation of y_j. The most negative correlated class label of y_j for $\mathbf{x}_i$ is explored according to Eq. (3),

$$\mathbf{N}_{ij} = \arg\max_{\forall\ l:l\neq j\ \&\ \bar{y}_{il}=1} p(y_j = 1 | y_l = 0, \mathcal{N}(\mathbf{x}_i)) \tag{3}$$

If two class labels are strongly negative correlated, the posterior probability will be large; otherwise, it will be small. We assume that the strong negative pairwise label correlation only exists between a label associated to it and a label it does not have. When calculating $\mathbf{N}_{ij}$ by Eq. (3), y_j should be associated with $\mathbf{x}_i$, and y_l should not be associated with $\mathbf{x}_i$, i.e. $y_{ij} = 1$ and $y_{il} = 0$. Because, if two labels are strongly negative correlated, they might not be co-occurred with each other, and thus the value of Eq. (3) will be larger. The probability $p(y_j = 1 | y_l = 0, \mathcal{N}(\mathbf{x}_i))$ is calculated by Eq. (4),

$$p(y_j = 1 | y_l = 0, \mathcal{N}(\mathbf{x}_i)) = \frac{\mathbf{c}_{\mathbf{x}_i}^{j\,T} \bar{\mathbf{c}}_{\mathbf{x}_i}^{l}}{\|\bar{\mathbf{c}}_{\mathbf{x}_i}^{l}\|_1} \tag{4}$$

where $\bar{\mathbf{c}}_{\mathbf{x}_i}^{l} = \mathbf{1} - \mathbf{c}_{\mathbf{x}_i}^{l}$, and $\mathbf{1}$ is a $k \times 1$ column vector with all its elements being 1. Consequently, the probability $p(y_j = 0 | y_l = 0, \mathcal{N}(\mathbf{x}_i)) = 1 - p(y_j = 1 | y_l = 0, \mathcal{N}(\mathbf{x}_i))$. All the procedures of exploring the local positive and negative pairwise label correlations are summarized in Algorithm 1.

Algorithm 1: Exploring Local Positive and Negative Pairwise Label Correlations.

Input: $\mathcal{D}$: the training data set, $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) | 1 \le i \le n\}, \mathbf{y}_i \in \{0, 1\}^q$; k: the number of nearest neighbors;
Output: $\mathbf{P}$ and $\mathbf{N}$: local pairwise label correlation matrix;

```
for i = 1 to n do
    D_t = D − x_i;
    find the k nearest neighbors N(x_i) of x_i from D_t;
    forall the y_ij = 1, 1 ≤ j ≤ q do
        Pr_max = ||c^j_{x_i}||_1 / k;
        forall the y_il = 1, 1 ≤ l ≤ q do
            if j ≠ l and p(y_j = 1 | y_l = 1, N(x_i)) > Pr_max then
                P_ij = l;
                Pr_max = p(y_j = 1 | y_l = 1, N(x_i));
        Pr_max = ||c^j_{x_i}||_1 / k;
        forall the ȳ_il = 1, 1 ≤ l ≤ q do
            if p(y_j = 1 | y_l = 0, N(x_i)) > Pr_max then
                N_ij = l;
                Pr_max = p(y_j = 1 | y_l = 0, N(x_i));
return P and N;
```

3.3. LPLC Model

Once the local positive and negative pairwise label correlations are obtained for each training example, it can be incorporated into multi-label classification models to improve the performance. We assume that similar examples may share the same label correlations. In the test stage, we first find the k nearest neighbors of each test example, and the corresponding local pairwise label correlations of these nearest neighbors. The test example will share these local pairwise label correlations with its k nearest neighbors. Fig. 3 shows the learning structure of the proposed method LPLC.

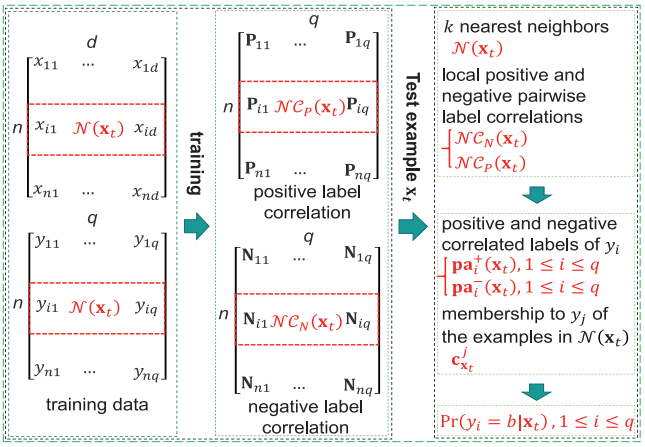


Fig. 3. Framework of the proposed method LPLC.

Given a test example $\mathbf{x}_t$, $\mathcal{N}(\mathbf{x}_t)$ is its k nearest neighbors. $\mathcal{NC}_P(\mathbf{x}_t)$ and $\mathcal{NC}_N(\mathbf{x}_t)$ are the local positive and negative pairwise label correlations of $\mathcal{N}(\mathbf{x}_t)$, and can be obtained from **P** and **N**, respectively. The test example $\mathbf{x}_t$ shares the same local pairwise label correlations with its k nearest neighbors $\mathcal{N}(\mathbf{x}_t)$. $\mathbf{pa}_i^+(\mathbf{x}_t)$ is the set of positive correlated labels of label y_i in the local pairwise label correlations $\mathcal{NC}_P(\mathbf{x}_t)$, and $\mathbf{pa}_i^-(\mathbf{x}_t)$ is the set of negative correlated labels of label y_i in the local pairwise label correlations $\mathcal{NC}_N(\mathbf{x}_t)$. To determine the prediction of the ith class label for $\mathbf{x}_t$, the proposed LPLC algorithm uses the following Maximum a Posteriori (MAP) rule Eq. (5),

$$\begin{aligned} h_i(\mathbf{x}_t) &= \arg\max_{b\in\{0,1\}} \Pr(y_i = b|\mathbf{x}_t) \\ &= \frac{1}{Z}\left(\alpha \sum_{y_c\in\mathbf{pa}_i^+(\mathbf{x}_t)} p(y_i = b, y_c = 1|\mathcal{N}(\mathbf{x}_t)) \right. \\ &\quad \left. + (1-\alpha) \sum_{y_c\in\mathbf{pa}_i^-(\mathbf{x}_t)} p(y_i = b, y_c = 0|\mathcal{N}(\mathbf{x}_t))\right) \\ &= \frac{1}{Z}\left(\alpha \sum_{y_c\in\mathbf{pa}_i^+(\mathbf{x}_t)} p(y_i = b|y_c = 1, \mathcal{N}(\mathbf{x}_t))p(y_c = 1|\mathcal{N}(\mathbf{x}_t)) \right. \\ &\quad \left. + (1-\alpha) \sum_{y_c\in\mathbf{pa}_i^-(\mathbf{x}_t)} p(y_i = b|y_c = 0, \mathcal{N}(\mathbf{x}_t))p(y_c = 0|\mathcal{N}(\mathbf{x}_t))\right) \end{aligned} \tag{5}$$

where Z is the normalizing constant which is needed to ensure that the density integrates to one, i.e. $Z = \Pr(y_i = 1|\mathbf{x}_t) + \Pr(y_i = 0|\mathbf{x}_t)$. Parameter $\alpha \in [0, 1]$ controls the tradeoff between the positive and negative correlations. The probability $p(y_c = b|\mathcal{N}(\mathbf{x}_t))$ can be calculated by Eq. (6),

$$p(y_c = b|\mathcal{N}(\mathbf{x}_t)) = \frac{k(1-b)+(-1)^{(1-b)}\|\mathbf{c}_{\mathbf{x}_t}^c\|_1}{k} \tag{6}$$

After computing the probabilities $\Pr(y_i = b|\mathbf{x}_t), 1 \le i \le q$. The LPLC classifier can be written as Eq. (7),

$$h(\mathbf{x}_t) = \{h_1(\mathbf{x}_t), h_2(\mathbf{x}_t), ..., h_q(\mathbf{x}_t)\} \tag{7}$$

where each classifier $h_i(\mathbf{x}_t)$ is defined as Eq. (8),

$$h_i(\mathbf{x}_t) = \arg\max_{b\in\{0,1\}} \Pr(y_i = b|\mathbf{x}_t) \tag{8}$$

The prediction procedures of LPLC can be summarized as Algorithm 2.

The complexity of LPLC has two parts: training and test (see Algorithms 1 and 2). In the training stage, LPLC needs to find k nearest neighbors for each training example. This leads to $\mathcal{O}(n^2)$ time complexity and $\mathcal{O}(n^2)$ memory consumption. To find the most positive and negative correlated class labels for each training example, it leads to $\mathcal{O}(ncq)$ time complexity and $\mathcal{O}(2nq)$ memory consumption, where c indicates the *Cardinality* (i.e. the average number of labels per instance) of a data set and it could be calculated according to Eq. (9), and q is the number of class labels of the data set. In the test stage, LPLC needs to find k nearest neighbors for each test example. This leads to $\mathcal{O}(mn)$ time complexity and $\mathcal{O}(mn)$ memory consumption. where m is the number of test examples. To predict the class labels for the test examples, it leads to $\mathcal{O}(mq)$ time complexity and $\mathcal{O}(mq)$ memory consumption,

Algorithm 2: LPLC Prediction.

Input: $\mathcal{D}$: the training data set,
$\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)|1 \le i \le n\}, \mathbf{y}_i \in \{0, 1\}^q$; $\mathbf{x}_t$: a test example;
k: the number of nearest neighbors; **P** and **N**: local pairwise label correlation matrices;
Output: $\mathbf{y}_t$: the set of predicted labels for $\mathbf{x}_t$;

```
find the k nearest neighbors N(x_t) for x_t;
get the local positive pairwise label correlations NC_P(x_t) of the examples in N(x_t) from P;
get the local negative pairwise label correlations NC_N(x_t) of the examples in N(x_t) from N;
for i = 1 to q do
    get the correlated labels pa_i^+(x_t) of y_i from NC_P(x_t);
    get the correlated labels pa_i^-(x_t) of y_i from NC_N(x_t);
    calculate Pr(y_i = b|x_t) by Eq. (5);
    h_i(x_t) = arg max_{b∈{0,1}} Pr(y_i = b|x_t);
y_t = [h_1(x_t), h_2(x_t), ..., h_q(x_t)];
```

4. Experiments

In this section, we empirically evaluate the effectiveness of our proposed method LPLC. We compare our method with five well-established multi-label learning algorithms over twelve multi-label benchmark data sets from various domains and scales. Sensitivity analysis for LPLC over the parameters k and α is conducted.

4.1. Experiment setup

4.1.1. Data sets

Twelve multi-label benchmark data sets are studied in this paper. The detailed characteristics of these data sets are summarized in Table 2. Data sets are ordered by the number of instances, and all these data sets can be downloaded from mulan[1], lamda[2], and meka[3]. In Table 2, for each data set, "# Instances" indicates the total number of examples, "# Features" means the number of features, and "# Labels" means the total number of class labels. "Cardinality" indicates the average number of labels per instance of a data set, which is calculated by Eq. (9).

$$\text{Cardinality} = \frac{1}{n}\sum_{i=1}^{n} |\mathbf{y}_i|_1 \tag{9}$$

where n is the number of total instances of a data set, $\mathbf{y}_i$ is a label set of the ith example.

[1] http://mulan.sourceforge.net/datasets.html
[2] http://lamda.nju.edu.cn/Data.ashx#data
[3] http://meka.sourceforge.net/

Table 2
Description of data sets.

ID	Data set	# Instances	# Features	# Labels	Cardinality	Domain
1	flags	194	19	7	3.392	image
2	cal500	502	68	174	26.044	music
3	emotions	593	72	6	1.869	music
4	yeast	2417	103	14	4.237	biology
5	corel5k	5000	499	374	3.522	image
6	rcv1subset1	6000	944	101	2.880	text
7	rcv1subset2	6000	944	101	2.634	text
8	corel16k001	13766	500	153	2.859	image
9	corel16k002	13761	500	164	2.882	image
10	delicious	16015	500	983	19.020	text
11	bookmark	87856	2150	208	2.028	text
12	imdb	120919	1001	28	2.000	text

4.2. Evaluation criteria

To evaluate the performance of different algorithms for multi-label classification, we use six common evaluation criteria in multi-label classification. Given a testing data set $\mathcal{D}_{test} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{m}$, where $\mathbf{y}_i \in \{0, 1\}^q$ is a ground truth label of the i-th test example, and $\hat{\mathbf{y}}_i$ is a predicted label.

- **Hamming Loss** evaluates how many times an instance-label pair is misclassified, i.e., a label not belonging to the instance is predicted or a label belonging to the instance is not predicted,

$$\text{Hamming Loss} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{q}|\mathbf{y}_i \Delta \hat{\mathbf{y}}_i| \tag{10}$$

  where Δ represents the symmetric difference of two sets.
- **Accuracy** evaluates Jaccard similarity between the ground truth labels and the predicted labels,

$$\text{Accuracy} = \frac{1}{m}\sum_{i=1}^{m}\frac{|\mathbf{y}_i \wedge \hat{\mathbf{y}}_i|}{|\mathbf{y}_i \vee \hat{\mathbf{y}}_i|} \tag{11}$$

- **Exact-Match** evaluates how many times the ground truth labels and the predicted labels are exactly matched,

$$\text{Exact-Match} = \frac{1}{m}\sum_{i=1}^{m}[\![\mathbf{y}_i = \hat{\mathbf{y}}_i]\!] \tag{12}$$

- $\mathbf{F_1}$ is the integrated version of precision and recall for each example,

$$\text{F}_1 = \frac{1}{m}\sum_{i=1}^{m}\frac{2p_i r_i}{p_i + r_i} \tag{13}$$

  where p_i and r_i are the precision and recall for the ith example.
- **Macro $\mathbf{F_1}$** is the integrated version of precision and recall for each label,

$$\text{Macro F}_1 = \frac{1}{q}\sum_{i=1}^{q}\frac{2p_i r_i}{p_i + r_i} \tag{14}$$

  where p_i and r_i are the precision and recall for the ith label.
- **Micro $\mathbf{F_1}$** is an extended version of the single label F_1 Measure to multi-label classification, and it treats every entry of the label vector as an individual instance regardless of label distinction,

$$\text{Micro F}_1 = \frac{2\sum_{j=1}^{q}\sum_{i=1}^{m} y_{ij}\hat{y}_{ij}}{\sum_{j=1}^{q}\sum_{i=1}^{m} y_{ij} + \sum_{j=1}^{q}\sum_{i=1}^{m}\hat{y}_{ij}} \tag{15}$$

The above evaluation criteria include label-based evaluation criteria (e.g., *Macro-F_1* and *Micro-F_1*) and example-based evaluation criteria (e.g., *Hamming Loss, Accuracy, Exact-Match* and F_1). These evaluation criteria are widely used in multi-label literatures, and they evaluate the performance of multi-label algorithms from various aspects. For all of these criteria except hamming loss, the larger the value of them, the better the performance of the classifier.

4.3. Comparison algorithms

We compare our proposed method LPLC with the following state-of-the-art multi-label classification algorithms.

1. **ECC** [31]: Ensemble Classifier Chains. It is an ensemble version of CC, where the ensemble size m is set to be 10. The chains order $y_{\pi(1)}, y_{\pi(2)}, \ldots, y_{\pi(l)}$ for each CC is generated randomly. Libsvm [42] is utilized as base binary learner for each binary (one-vs-rest) classifier of ECC, where the kernel function is set as linear kernel, and the parameter C is tuned in $\{10^{-4}, 10^{-3}, \ldots, 10^{4}\}$.
2. **MCC**[4] [38]: Efficient monte carlo methods for multi-dimensional learning with classifier chains. SVM fitted with logistic models in the SMO implementation with polynomial kernel is utilized as base binary learner for each binary (one-vs-rest) classifier of MCC, and C is tuned in $\{10^{-4}, 10^{-3}, \ldots, 10^{4}\}$.
3. **ML-LOC**[5] [2]: Multi-label learning by exploiting label correlations locally. For ML-LOC, $m = 15$, and λ_1, λ_2 are searched in $\{10^{-4}, 10^{-3}, \ldots, 10^{4}\}$.
4. **ML-*k*NN**[6] [22]: A lazy learning approach to multi-label learning. k is searched in $\{3, 5, \ldots, 21\}$.
5. **LLSF**[7] [30]: Learning *label specific features* for multi-label classification. Parameters α, β are both searched in $\{2^{-10}, 2^{-9}, \ldots, 2^{10}\}$, and ρ is searched in {0.1, 1, 10}.
6. **LPLC**[8]: The proposed method in this paper. Parameter α is searched in $\{0.1, 0.2, \ldots, 1\}$, and k is searched in $\{3, 5, \ldots, 21\}$.

All the comparing algorithms are summarized in Table 3. The column named "Type of Correlation" indicates which type of label correlation the corresponding algorithm incorporates.

4.4. Parameter sensitivity analysis

In our proposed method LPLC, there are two important parameters, i.e., k is the number of nearest neighbors, and α controls the tradeoff between positive and negative correlation. To study the

[4] Source code: http://meka.sourceforge.net/
[5] Source code: http://lamda.nju.edu.cn/Data.ashx
[6] Source code: http://cse.seu.edu.cn/PersonalPage/zhangml/
[7] Source code: http://www.escience.cn/people/huangjun/index.html.
[8] Source code will be available at: http://www.escience.cn/people/huangjun/index.html.

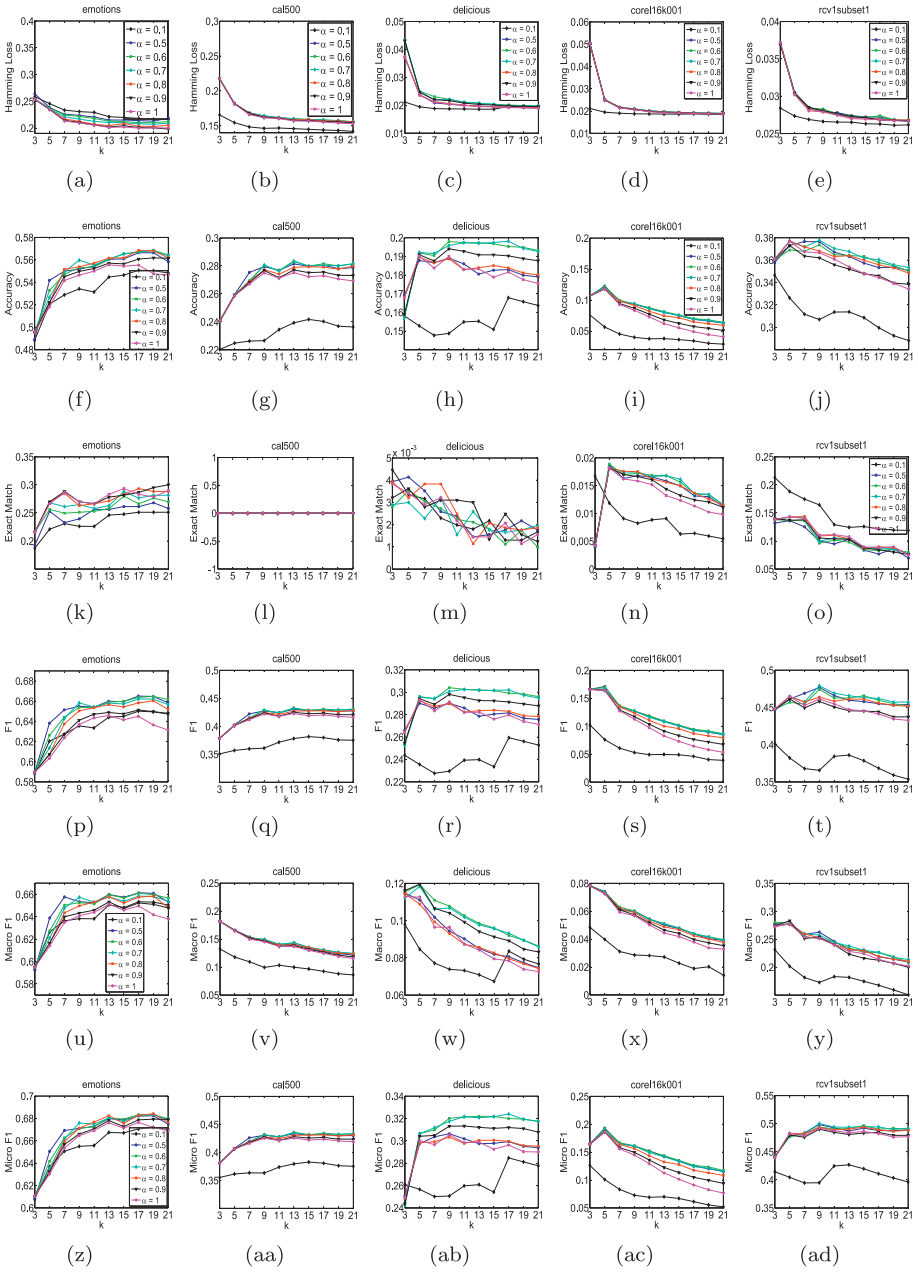


Fig. 4. Parameter sensitivity analysis of LPLC with respect to k and α.

Table 3
Comparing algorithms.

Method	Type of correlation	Publication
ECC	High-order, global, positive	[31]
MCC	High-order, global, positive	[38]
ML-LOC	High-order, local, positive	[2]
ML-kNN	First-order	[22]
LLSF	Second-order, global, positive	[30]
LPLC	Second-order, local, positive and negative	proposed

Table 4
Summary of the Friedman Statistics $F_F(k = 6, N = 10)$ and the critical value in terms of each evaluation criterion (k: # Comparing Algorithms; N: # Data sets).

Evaluation criterion	F_F	Critical value ($\alpha = 0.05$)
Hamming Loss	9.2609	2.4221
Accuracy	11.4148	
Exact-Match	5.5564	
F_1	11.2442	
Macro F_1	23.7784	
Micro F_1	16.7563	

sensitivity of LPLC with respect to k and α, we conduct parameter sensitivity analysis for LPLC on five data sets, e.g., emotions, cal500, rcv1subset1, corel16k001, and delicious. Parameter k is searched in {3, 5, ..., 21}, and α is tuned in $\{0.1, 0.5, 0.6, \ldots, 1\}$. Fig. 4 illustrates the experimental results of LPLC in terms of each evaluation criterion. The horizontal axis of each sub-figure indicates different values of k, and the vertical axis represents the result of each evaluation criterion.

Based on the experimental results, the following observations can be made:

Table 5
Experimental results of each comparing algorithm (mean± std) on ten data sets in terms of *Hamming Loss, Accuracy and Exact Match*.

Data set	*Hamming Loss* ↓					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.284± 0.027	0.288± 0.022	0.273± 0.048	**0.262**± 0.027	0.267± 0.015	0.279± 0.010
cal500	0.140± 0.001	**0.138**± 0.005	0.204± 0.006	**0.138**± 0.002	0.144± 0.005	0.155± 0.002
emotions	0.202± 0.004	0.214± 0.016	0.215± 0.021	0.210± 0.008	**0.191**± 0.013	0.197± 0.011
yeast	0.197± 0.007	0.228± 0.008	0.218± 0.012	**0.193**± 0.003	0.203± 0.007	0.202± 0.008
corel5k	**0.009**± 0.000	0.012± 0.000	0.017± 0.000	**0.009**± 0.000	0.012± 0.000	0.011± 0.000
rcv1subset1	**0.026**± 0.000	0.029± 0.000	0.034± 0.000	**0.026**± 0.000	0.029± 0.000	0.030± 0.000
rcv1subset2	0.024± 0.000	0.025± 0.001	0.033± 0.004	**0.022**± 0.000	0.025± 0.001	0.027± 0.000
corel16k001	**0.019**± 0.000	0.021± 0.001	0.020± 0.000	**0.019**± 0.000	0.023± 0.000	0.025± 0.000
corel16k002	**0.017**± 0.000	0.019± 0.001	0.030± 0.000	0.019± 0.000	0.021± 0.000	0.024± 0.000
delicious	**0.018**± 0.000	0.031± 0.000	0.029± 0.000	**0.018**± 0.000	0.019± 0.000	0.022± 0.000
bookmark	**0.009**± 0.000	0.016± 0.000	DNF	DNF	**0.009**± 0.000	0.017± 0.000
imdb	**0.071**± 0.000	0.085± 0.000	0.120± 0.000	DNF	0.085± 0.000	0.081± 0.003
Data set	*Accuracy* ↑					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.555± 0.035	0.560± 0.027	0.580± 0.069	0.568± 0.028	0.581± 0.038	**0.607**± 0.016
cal500	0.201± 0.007	0.194± 0.011	0.201± 0.005	0.204± 0.007	0.263± 0.014	**0.276**± 0.007
emotions	0.541± 0.018	0.540± 0.034	0.553± 0.042	0.497± 0.029	0.518± 0.038	**0.565**± 0.016
yeast	0.509± 0.015	0.490± 0.019	0.491± 0.025	0.510± 0.000	0.500± 0.017	**0.542**± 0.013
corel5k	0.041± 0.005	0.118± 0.009	0.104± 0.003	0.039± 0.002	**0.144**± 0.007	0.121± 0.009
rcv1subset1	0.274± 0.008	0.339± 0.008	0.320± 0.006	0.256± 0.003	0.351± 0.005	**0.377**± 0.004
rcv1subset2	0.281± 0.007	**0.411**± 0.013	0.332± 0.041	0.308± 0.009	0.356± 0.007	0.383± 0.012
corel16k001	0.026± 0.002	0.091± 0.016	0.106± 0.001	0.034± 0.004	**0.144**± 0.005	0.117± 0.001
corel16k002	0.026± 0.004	0.095± 0.025	0.100± 0.002	0.029± 0.004	**0.132**± 0.004	0.122± 0.005
delicious	0.129± 0.005	0.154± 0.001	0.168± 0.001	0.134± 0.003	**0.201**± 0.002	0.194± 0.001
bookmark	0.236± 0.004	0.271± 0.001	DNF	DNF	0.257± 0.003	**0.292**± 0.002
imdb	0.006± 0.000	0.083± 0.001	0.143± 0.002	DNF	**0.243**± 0.002	0.211± 0.006
Data set	*Exact-Match* ↑					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.098± 0.030	**0.191**± 0.014	0.140± 0.086	0.115± 0.038	0.139± 0.055	0.123± 0.030
cal500	0.000± 0.000	0.000± 0.000	0.000± 0.000	0.000± 0.000	0.000± 0.000	0.000± 0.000
emotions	0.292± 0.031	**0.309**± 0.041	0.299± 0.060	0.261± 0.014	0.292± 0.027	0.303± 0.023
yeast	0.174± 0.015	0.164± 0.026	0.189± 0.021	**0.199**± 0.004	0.151± 0.019	0.186± 0.026
corel5k	0.005± 0.001	0.015± 0.003	0.007± 0.004	0.007± 0.000	0.008± 0.001	**0.016**± 0.002
rcv1subset1	0.090± 0.011	**0.222**± 0.007	0.137± 0.007	0.095± 0.005	0.051± 0.006	0.143± 0.009
rcv1subset2	0.156± 0.010	**0.328**± 0.004	0.156± 0.079	0.207± 0.019	0.173± 0.014	0.210± 0.009
corel16k001	0.004± 0.001	**0.019**± 0.004	0.017± 0.002	0.004± 0.000	0.014± 0.002	0.018± 0.001
corel16k002	0.004± 0.001	0.018± 0.003	**0.019**± 0.001	0.004± 0.000	0.013± 0.001	0.017± 0.002
delicious	0.001± 0.000	0.001± 0.002	**0.003**± 0.002	0.002± 0.000	0.000± 0.000	**0.003**± 0.000
bookmark	0.211± 0.003	0.199± 0.001	DNF	DNF	0.208± 0.004	**0.228**± 0.002
imdb	0.004± 0.000	0.060± 0.000	0.021± 0.002	DNF	0.077± 0.002	**0.111**± 0.002

- After incorporating negative label correlations (i.e. $\alpha < 1$), the result of LPLC is better than only considering positive label correlations (i.e. when $\alpha = 1$, LPLC only considering positive label correlations) in most cases. These results clearly justify the effectiveness of exploiting negative label correlations.
- If α is too small, e.g., when $\alpha = 0.1$, the effect of positive label correlation is much less than negative label correlation, then the performance of LPLC is poor. The main reason might be the sparsity of the label matrix of multi-label data, the times that a label pair not co-occurred is much larger than that of co-occurred. Thus, the reliability of negative label correlation, learned by LPLC, is weaker than that of positive label correlation. We suggest that α should be greater than or equal to 0.5, especially for the data set with large number of labels but small cardinality.
- Given α, the performance of LPLC is improved and then degraded with the increase of k on the data set with small number of labels or large number of labels and large cardinality (e.g., emotions, cal500, and delicious, where the detailed characteristics of them are all summarized in Table 2); while on the data set with large number of labels but small cardinality (e.g., rcv1subset1 and corel16k001), the performance of LPLC degrades quickly with the increase of k. One reason might be that the smaller the cardinality, the more sparse the label matrix, thus the reliability of negative label correlation becomes lower.

According to the parameter sensitivity analysis, we can see that the best parameter setting for different data sets are different. Thus, in the following experiments, we search the best configuration for the parameters for each data set by 5-fold cross validation on the training data.

4.5. Comparison results

In our experiment, on each data set, we randomly split each data set into the training (80%) and test (20%) sets for 10 times, and report the average results as well as standard deviations over the 10 repetitions. The best experimental results of each comparing algorithm over the data sets are reported in Tables 5 and 6. "DNF" indicates the experiment is not finished, and ↓(↑) indicates the smaller (larger) the value, the better the performance. Best results are highlighted in bold.

To analyze the relative performance among the comparing algorithms systematically, Friedman test [43] is employed to conduct performance analysis here which is regarded as the favorable statistical test for comparisons among multiple algorithms

Table 6
Experimental results of each comparing algorithm (mean± std) on ten data sets in terms of F_1, *Macro* F_1, *and Micro* F_1.

Data set	F_1 ↑					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.679± 0.034	0.676± 0.026	0.700± 0.053	0.695± 0.025	0.698± 0.033	**0.732**± 0.012
cal500	0.330± 0.009	0.319± 0.016	0.325± 0.009	0.334± 0.010	0.410± 0.017	**0.423**± 0.008
emotions	0.625± 0.018	0.617± 0.035	0.638± 0.041	0.576± 0.033	0.590± 0.040	**0.650**± 0.013
yeast	0.615± 0.015	0.600± 0.016	0.595± 0.026	0.612± 0.001	0.609± 0.015	**0.648**± 0.009
corel5k	0.056± 0.006	0.167± 0.013	0.152± 0.003	0.052± 0.003	**0.208**± 0.011	0.168± 0.011
rcv1subset1	0.346± 0.007	0.390± 0.010	0.395± 0.008	0.321± 0.006	0.456± 0.006	**0.465**± 0.003
rcv1subset2	0.331± 0.008	0.444± 0.017	0.405± 0.026	0.349± 0.005	0.426± 0.006	**0.450**± 0.013
corel16k001	0.035± 0.003	0.124± 0.022	0.148± 0.002	0.047± 0.006	**0.204**± 0.007	0.164± 0.002
corel16k002	0.036± 0.006	0.131± 0.036	0.138± 0.004	0.040± 0.006	**0.186**± 0.006	0.172± 0.006
delicious	0.201± 0.007	0.243± 0.001	0.263± 0.001	0.218± 0.005	**0.306**± 0.003	0.298± 0.001
bookmark	0.246± 0.004	0.304± 0.002	DNF	DNF	0.277± 0.003	**0.322**± 0.002
imdb	0.007± 0.000	0.092± 0.001	0.200± 0.002	DNF	**0.311**± 0.002	0.251± 0.010
Data set	*Macro* F_1 ↑					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.534± 0.041	0.573± 0.020	**0.635**± 0.069	0.546± 0.012	0.615± 0.033	0.623± 0.048
cal500	0.060± 0.002	0.039± 0.002	**0.163**± 0.001	0.044± 0.002	0.068± 0.007	0.125± 0.004
emotions	0.633± 0.016	0.629± 0.039	**0.655**± 0.031	0.612± 0.030	0.625± 0.036	0.654± 0.017
yeast	0.364± 0.013	0.314± 0.021	0.371± 0.016	0.346± 0.007	0.357± 0.013	**0.430**± 0.006
corel5k	0.021± 0.003	0.025± 0.003	**0.044**± 0.002	0.017± 0.000	0.039± 0.002	0.040± 0.003
rcv1subset1	0.184± 0.005	0.202± 0.011	0.247± 0.010	0.141± 0.007	0.251± 0.004	**0.278**± 0.006
rcv1subset2	0.142± 0.005	0.205± 0.009	0.216± 0.013	0.131± 0.005	0.205± 0.008	**0.241**± 0.006
corel16k001	0.027± 0.003	0.041± 0.004	0.031± 0.001	0.020± 0.001	0.064± 0.004	**0.073**± 0.002
corel16k002	0.029± 0.004	0.040± 0.003	**0.084**± 0.001	0.019± 0.001	0.067± 0.003	0.077± 0.004
delicious	0.064± 0.002	0.113± 0.002	**0.131**± 0.002	0.066± 0.000	0.093± 0.003	0.104± 0.002
bookmark	0.135± 0.002	**0.197**± 0.002	DNF	DNF	0.142± 0.002	0.187± 0.001
imdb	0.011± 0.002	0.035± 0.000	**0.106**± 0.001	DNF	0.078± 0.001	0.019± 0.002
Data set	*Micro* F_1 ↑					
	MLkNN	ECC	MCC	ML-LOC	LLSF	LPLC
flags	0.699± 0.026	0.699± 0.027	0.721± 0.059	0.715± 0.025	0.727± 0.020	**0.748**± 0.013
cal500	0.327± 0.009	0.313± 0.016	0.330± 0.010	0.329± 0.010	0.409± 0.018	**0.426**± 0.007
emotions	0.663± 0.018	0.649± 0.029	0.665± 0.034	0.637± 0.022	0.651± 0.030	**0.677**± 0.016
yeast	0.639± 0.013	0.612± 0.014	0.617± 0.024	0.640± 0.002	0.633± 0.013	**0.667**± 0.009
corel5k	0.081± 0.008	0.180± 0.013	0.155± 0.003	0.078± 0.003	**0.244**± 0.012	0.197± 0.012
rcv1subset1	0.386± 0.007	0.395± 0.008	0.402± 0.006	0.365± 0.001	**0.495**± 0.005	0.482± 0.002
rcv1subset2	0.353± 0.009	0.418± 0.016	0.387± 0.019	0.380± 0.006	0.443± 0.003	**0.456**± 0.008
corel16k001	0.050± 0.003	0.143± 0.021	0.168± 0.001	0.065± 0.009	**0.243**± 0.007	0.186± 0.002
corel16k002	0.053± 0.007	0.148± 0.032	0.145± 0.002	0.056± 0.009	**0.233**± 0.008	0.190± 0.007
delicious	0.220± 0.007	0.240± 0.002	0.264± 0.002	0.237± 0.005	**0.343**± 0.003	0.313± 0.002
bookmark	0.272± 0.004	0.250± 0.003	DNF	DNF	**0.308**± 0.003	0.248± 0.001
imdb	0.010± 0.000	0.091± 0.001	0.217± 0.001	DNF	**0.330**± 0.002	0.241± 0.009

over a number of data sets. Given k comparing algorithms and N data sets, let r_i^j be the rank of the jth algorithm on the ith data set, where average ranks are assigned in case of ties. Let $R_j = \frac{1}{N}\sum_i r_i^j$ be the average rank for the jth algorithm. Under the null-hypothesis, which states that all the comparing algorithms perform equivalently. The Friedman statistic F_F will be distributed according to the F-distribution with $(k-1)$ numerator degrees of freedom and $(k-1)(N-1)$ denominator degrees of freedom,

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{16}$$

where $\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j^k R_j^2 - \frac{k(k+1)^2}{4}\right]$.

Table 4provides the Friedman statistics F_F and the corresponding critical values in terms of each evaluation criterion[9]. As shown in Table 4, at significance level $\alpha = 0.05$, the null hypothesis is clearly rejected in terms of each evaluation criterion. Consequently, we can proceed with a post-hoc test [43] to analyze the relative performance among the comparing algorithms. The Bonferroni–Dumn test [43] is employed to test whether our proposed method LPLC achieves competitive performance against the comparing algorithms, where LPLC is considered as the control algorithm. The performance between two classifiers is significantly different if their corresponding average ranks differ by at least one critical difference,

$$\mathrm{CD} = q_\alpha\sqrt{\frac{k(k+1)}{6N}} \tag{17}$$

For Bonferroni–Dumn test, $q_\alpha = 2.576$ at significance level $\alpha = 0.05$, and thus $\mathrm{CD} = 2.1552$ $(k = 6, N = 10)$. Fig. 5 shows the CD diagrams on each evaluation criterion. In each sub-figure, any comparing algorithm whose average rank is within one CD to that of LPLC is connected. Otherwise, any algorithm not connected with LPLC is considered to have significant different performance between them.

Based on these experimental results, the following observations can be made:

- As *first-order* algorithms try to optimize *Hamming Loss*. On *Hamming Loss* (see Fig. 5(a)), it can be seen that these algorithms, which incorporate *second-order* (e.g., LLSF and LPLC) or *high-order* (e.g., ECC, MCC) label correlations, obtain worse performance than the *first-order* algorithm MLkNN.

[9] As MLLOC does not finish the experiments on bookmark and imdb, and MCC does not finish the experiment on bookmark. For simplicity, we only conduct the statistic test on the results of the other ten data sets. Thus, here k is 6, and N equals to 10.

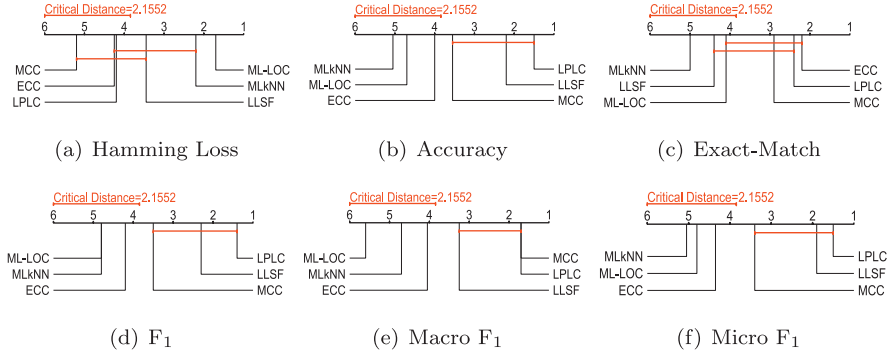


Fig. 5. Comparison of LPLC (control algorithm) against other comparing algorithms with the Bonferroni-Dumn test. Groups of classifiers that are not significantly different from LPLC (at $\alpha = 0.05$) are connected.

- While on *Exact-Match* (see Fig. 5(c)), these algorithms, which incorporate *second-order* (e.g., LLSF and LPLC) or *high-order* (e.g., ECC, MCC, and ML-LOC) label correlations, obtain better performance than the *first-order* algorithm ML*k*NN. As previous works suggest that optimizing *Exact-Match* need to model label correlations.
- LPLC achieves comparable performance against all the comparing algorithms in terms of *Hamming Loss* and *Exact-Match*.
- LPLC significantly outperforms ML-LOC, ML*k*NN and ECC, and achieves statistically superior or at least comparable performance against LLSF and MCC in terms of *Accuracy*, F_1, *Macro* F_1 and *Micro* F_1.

To summarize, our proposed method LPLC achieves a competitive performance against other well-established multi-label classification algorithms.

5. Conclusion

In this paper, we propose a simple and effective Bayesian model for multi-label classification by exploiting local positive and negative pairwise label correlations. LPLC tries to find the positive and negative correlated class label for each ground truth label of all the training examples. The experimental results show that a multi-label classifier can benefit from both positive and negative label correlation among labels. We should set a lower weight to the negative label correlation, especially for the data sets with large number of labels but small cardinality. LPLC achieves statistically superior or comparable performance against the state-of-art methods in terms of each evaluation criterion.

Acknowledgment

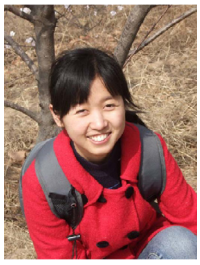
This work was supported in part by National Basic Research Program of China (973 Program): 2015CB351802, in part by National Natural Science Foundation of China: 61303153, 61332016, 61620106009, 61572488 and 61672497, in part by 863 program of China: 2014AA015202, and in part by Bureau of Frontier Sciences and Education (CAS): QYZDJ-SSW-SYS013.

References

[1] M.-L. Zhang, Z.-H. Zhou, A review on multi-label learning algorithms, IEEE Trans. Knowl. Data Eng. 26 (8) (2014) 1819–1837.
[2] S.-J. Huang, Z.-H. Zhou, Multi-label learning by exploiting label correlations locally, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2012, pp. 949–955.
[3] J. Huang, G.-R. Li, S.-H. Wang, W.-G. Zhang, Q.-M. Huang, Group sensitive classifier chains for multi-label classification, in: Proceedings of the IEEE International Conference on Multimedia Expo, 2015, pp. 1–6.
[4] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, H. Adam, Large-scale object classification using label relation graphs, in: Proceedings of the European Conference on Comp. Vision, 2014, pp. 48–64.
[5] M.-L. Zhang, Z.-H. Zhou, Multilabel neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (10) (2006) 1338–1351.
[6] A.K. McCallum, Multi-label text classification with a mixture model trained by em, in: Proceedings of the AAAI'99 Workshop on Text Learn., 1999.
[7] H. Kazawa, T. Izumitani, H. Taira, E. Maeda, Maximal margin labeling for multi-topic text categorization, in: Proceedings of the Neural Information Processing Systems, 2005, pp. 649–656.
[8] F.-M. Sun, J.-H. Tang, H.-J. Li, G.-J. Qi, T.S. Huang, Multi-label image categorization with sparse factor representation, IEEE Trans. Image Process 23 (3) (2014) 1028–1037.
[9] Z. He, C. Chen, J. Bu, P. Li, D. Cai, Multi-view based multi-label propagation for image annotation, Neurocomputing 168 (2015) 853–860.
[10] X. Jia, F.-M. Sun, H.-J. Li, Y.-D. Cao, X. Zhang, Image multi-label annotation based on supervised nonnegative matrix factorization with new matching measurement, Neurocomputing 219 (2017) 518–525.
[11] S. Xia, P. Chen, J. Zhang, X. Li, B. Wang, Utilization of rotation-invariant uniform lbp histogram distribution and statistics of connected regions in automatic image annotation based on multi-label learning, Neurocomputing (2016), doi:10.1016/j.neucom.2016.09.087.
[12] F. Kang, R. Jin, R. Sukthankar, Correlated label propagation with application to multi-label learning, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006, pp. 1719–1726.
[13] G.-J. Qi, X.-S. Hua, Y. Rui, J.-H. Tang, T. Mei, H.-J. Zhang, Correlative multi-label video annotation, in: Proceedings of the 15th ACM International Conference on Multimedia, 2007, pp. 17–26.
[14] X. Wang, G. Sukthankar, Multi-label relational neighbor classification using social context features, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2013, pp. 464–472.
[15] K. Trohidis, G. Tsoumakas, G. Kalliris, I.P. Vlahavas, Multi-label classification of music into emotions, in: Proceedings of the International Society of Music Information Retrieval, 2008, pp. 325–330.
[16] B. Wu, E.-H. Zhong, A. Horner, Q. Yang, Music emotion recognition by multi-label multi-layer multi-instance multi-view learning, in: Proceedings of the 22nd ACM International Conference on Multimedia, 2014, pp. 117–126.
[17] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, in: Data Mining and Knowledge Discovery Handbook, 2010, pp. 667–685.
[18] M.R. Boutell, J.-B. Luo, X.-P. Shen, C.M. Brown, Learning multi-label scene classification, Pattern Recognit. 37 (9) (2004) 1757–1771.
[19] K. Brinker, E. Hüllermeier, Case-based multilabel ranking, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2007, pp. 702–707.
[20] X.-T. Lin, X.-W. Chen, Mr.knn: Soft relevance for multi-label classification, in: Proceedings of the Conference on Information and Knowledge Management, 2010, pp. 349–358.
[21] A. Veloso, W. Meira, M. Gonçalves, M. Zaki, Multi-label lazy associative classification, in: Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases, 2007, pp. 605–612.
[22] M.-L. Zhang, Z.-H. Zhou, Ml-knn: A lazy learning approach to multi-label learning, Pattern Recognit. 40 (7) (2007) 2038–2048.
[23] J. Huang, G. Li, S. Wang, Q. Huang, Categorizing social multimedia by neighborhood decision using local pairwise label correlation, in: Proceedings of the IEEE International Conference on Data Mining Workshop, 2014, pp. 913–920.
[24] H. Liu, X.-D. Wu, S.-C. Zhang, Neighbor selection for multilabel classification, Neurocomputing 182 (2016) 187–196.
[25] A. Clare, R.D. King, Knowledge discovery in multi-label phenotype data, in: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, 2001, pp. 42–53.
[26] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, H. Blockeel, Decision trees for hierarchical multi-label classification, Mach. Learn. 73 (2) (2008) 185–214.
[27] H. Jing, S.D. Lin, Neural conditional energy models for multi-label classification, in: Proceedings of IEEE International Conference on Data Mining, 2014, pp. 240–249.
[28] A. Elisseeff, W. Jason, A kernel method for multi-labelled classification, in: Proceedings of the 14th International Conference on Neural Information Processing Systems, 2001, pp. 681–687.

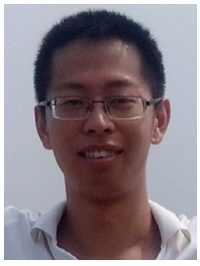
[29] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Mach. Learn. 73 (2) (2008) 133–153.
[30] J. Huang, G.-R. Li, Q.-M. Huang, X.-D. Wu, Learning label specific features for multi-label classification, in: Proceedings of the IEEE International Conference on Data Mining, 2015, pp. 181–190.
[31] R. Jesse, P. Bernhard, H. Geoff, F. Eibe, Classifier chains for multi-label classification, in: Proceedings of the European Conference on Machine Learning, 2009, pp. 254–269.
[32] K. Dembczyński, W. Cheng, E. Hüllermeier, Bayes optimal multilabel classification via probabilistic classifier chains, in: Proceedings of the International Conference on Machine Learning, 2010, pp. 1609–1614.
[33] W. Bi, J. Kwok, Bayes-optimal hierarchical multilabel classification, IEEE Trans. Knowl. Data Eng. 27 (11) (2015) 2907–2918.
[34] A. Alali, M. Kubat, Prudent: A pruned and confident stacking approach for multi-label classification, IEEE Trans. Knowl. Data Eng. 27 (9) (2015) 2480–2493.
[35] J. Huang, G.-R. Li, Q.-M. Huang, X.-D. Wu, Learning label-specific features and class-dependent labels for multi-label classification, IEEE Trans. Knowl. Data Eng. 28 (12) (2016) 3309–3323.
[36] J.H. Zaragoza, L.E. Sucar, E.F. Morales, C. Bielza, P. Larrañaga, Bayesian chain classifiers for multidimensional classification, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2011, pp. 2192–2197.
[37] K. Abhishek, V. Shankar, M.A. Krishna, E. Charles, Beam search algorithms for multilabel learning, Mach. Learn. 92 (1) (2013) 65–89.
[38] J. Read, L. Martino, D. Luengo, Efficient monte carlo methods for multi-dimensional learning with classifier chains, Pattern Recognit. 47 (3) (2014) 1535–1546.
[39] L. Sun, M. Kudo, Polytree-augmented classifier chains for multi-label classification, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2015, pp. 3834–3840.
[40] L.-Q. Wang, Z.-C. Zhao, F. Su, Efficient multi-modal hypergraph learning for social image classification with complex label correlations, Neurocomputing 171 (2016) 242–251.
[41] F. Briggs, X. Fern, R. Raich, Context-aware miml instance annotation: exploiting label correlations with classifier chains, Knowl. Inf. Syst. 43 (1) (2015) 53–79.
[42] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27:1–27:27.
[43] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res 7 (Jan) (2006) 1–30.

Jun Huang received the M.S. degree in computer science from the Anhui University of Technology, Ma'anshan, China, in 2011. Now, he is a Ph.D. student in School of Computer and Control Engineering, University of Chinese Academy of Sciences (UCAS). Before joining UCAS, he was a lecturer with Anhui University of Technology. His research interests include machine learning and data mining.



Guorong Li received her B.S. degree in computer science from Renmin University of China, in 2006 and Ph.D. degree in computer science from the Graduate University of Chinese Academy of Sciences in 2012. Now, she is an associate professor within the University of Chinese Academy of Sciences. Her research interests include object tracking, pattern recognition, cross-media analysis and multi-label learning.

Shuhui Wang received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, in 2006 and the Ph.D. degree from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2012. He is a researcher with Institute of Computing Technology, Chinese Academy of Sciences, and Key Laboratory of Intelligent Information Processing, Chinese Academy of Sciences. His research interests include semantic image analysis, image and video retrieval, and large-scale web multimedia data mining.



Zhe Xue received the B.S. degree in Electronic Engineering from Civil Aviation University of China, Tianjin, China, in 2010. He is currently pursuing the Ph.D. degree in University of Chinese Academy of Sciences, Beijing, China. His research interests include machine learning, computer vision and multimedia data mining.

Qingming Huang is currently a professor and deputy dean in the School of Computer and Control Engineering, University of Chinese Academy of Sciences (UCAS). His research interests include multimedia computing, image/video processing, pattern recognition and computer vision. He has published more than 300 academic papers in international journals such as IEEE Transactions on Image Processing, IEEE Transactions on Multimedia, IEEE Transactions on CSVT, and top level international conferences including ACM Multimedia, ICCV, CVPR, VLDB, IJCAI, etc.