Learning Label Specific Features for Multi-Label Classification

Jun Huang[1;2;3], Guorong Li[1;2;3;*], Qingming Huang[1;2;3;*], Xindong Wu[4;5]
[1] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing, 101480, China
[2] Key Lab of Big Data Mining and Knowledge Management, University of Chinese Academy of Sciences, Beijing, China
[3] Key Lab of Intelligent Information Processing,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China
[4] School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China
[5] Department of Computer Science, University of Vermont, USA
Email: {jun.huang, guorong.li}@vipl.ict.ac.cn, qmhuang@ict.ac.cn, xwu@uvm.edu

***Abstract*—Binary relevance (BR) is a well-known framework for multi-label classification. It decomposes multi-label classification into binary (one-vs-rest) classification subproblems, one for each label. The BR approach is a simple and straightforward way for multi-label classification, but it still has several drawbacks. First, it does not consider label correlations. Second, each binary classifier may suffer from the issue of class-imbalance. Third, it can become computationally unaffordable for data sets with many labels. Several remedies have been proposed to solve these problems by exploiting label correlations between labels and performing label space dimension reduction.**

Meanwhile, inconsistency, another potential drawback of BR, is often ignored by researchers when they construct multi-label classification models. Inconsistency refers to the phenomenon that if an example belongs to more than one class label, then during the binary training stage, it can be considered as both positive and negative example simultaneously. This will mislead binary classifiers to learn suboptimal decision boundaries. In this paper, we seek to solve this problem by learning label specific features for each label. We assume that each label is only associated with a subset of features from the original feature set, and any two strongly correlated class labels can share more features with each other than two uncorrelated or weakly correlated ones. The proposed method can be applied as a feature selection method for multi-label learning and a general strategy to improve multi-label classification algorithms comprising a number of binary classifiers. Comparison with the state-of-the-art approaches manifests competitive performance of our proposed method.

Instance	y_1	y_2	y_3	y_4
$\mathbf{x}_1$	1	1	1	1
$\mathbf{x}_2$	1	1	1	0
$\mathbf{x}_3$	1	1	0	0
$\mathbf{x}_4$	0	1	0	0
$\mathbf{x}_5$	0	1	1	1

$\longrightarrow$

Instance	y_1	$\bar{y_1}$
$\mathbf{x}_1$	1	0
$\mathbf{x}_2$	1	0
$\mathbf{x}_3$	1	0
$\mathbf{x}_4$	0	1
$\mathbf{x}_5$	0	1

Fig. 1. This is a toy example. The left table shows a data set which is composed of five instances and associated with four class labels. The right table shows the processed data set for training a binary (y_1 vs $\bar{y_1}$) classification model, where $\{\bar{y_1}\} = \{y_2, y_3, y_4\}$.

Instance	y_1	$\bar{y_1}$
$\mathbf{x}_1$	1	1
$\mathbf{x}_2$	1	1
$\mathbf{x}_3$	1	1
$\mathbf{x}_4$	0	1
$\mathbf{x}_5$	0	1

Fig. 2. Inconsistency of BR: During the binary training stage for one given class label (eg. y_1), instances which are associated with multiple labels can be considered as both positive and negative example simultaneously.

I. Introduction

In many real applications, one example can be assigned with multiple class labels simultaneously. For example, in image annotation, an image can be annotated as “sea water” and “sea bird”. Multi-label classification deals with examples having multiple class labels simultaneously. The task is to learn a classification model which can predict a set of possible labels for an unseen example. It has attracted significant attentions from researchers and has been applied to a variety of domains, such as text classification [1]–[3], image annotation [4]–[6], video annotation [7], [8], social networks [9], music emotion categorization [10], [11], etc.

A significant number of algorithms have been proposed to solve multi-label classification problems [12], [13]. A common approach for multi-label classification is to perform problem transformation, where a multi-label problem is transformed into one or more single-label subproblems. Binary Relevance (BR) [4] approach, one of the representative algorithms of problem transformation methods, considers each label as an independent binary (one-vs-rest) classification problem. In each binary classification problem, examples associating with a given class label are considered as positive examples, while those without this class label are considered as negative ones. Then, traditional single-label classification algorithms can be used to learn the binary classification model directly, such as Logistic Regression, Naive Bayesian, SVM, etc.

As an effective method for multi-label classification, the BR approach is theoretically simple and intuitive, and many advanced multi-label classification algorithms are built under this framework. However, it fails to capture the correlation information among different labels, which is critical for many applications where the semantics conveyed by different labels are correlated. On the other hand, it becomes computationally unaffordable for data sets with many labels, and each binary classifier may suffer from the issue of class-imbalance when the number of labels is large and label density is low. In the past decades, many well-established methods have been proposed to solve multi-label classification by incoporating label correlations [14]–[22]. For the problem of class-imbalance, several works have been proposed, such as [23]–[25]. To tackle

1550-4786/15 $31.00 © 2015 IEEE
DOI 10.1109/ICDM.2015.67

multi-label data with many labels, researchers try to perform label space dimension reduction (LSDR) [26]–[30].

Meanwhile, BR suffers from another important problem, inconsistency, and this problem is often ignored by researchers when they construct multi-label classification models. In each binary classification problem, if an example belongs to more than one class label, then during the binary training stage, it can be considered as both positive and negative examples simultaneously. For example, in Fig. 1, instance $\mathbf{x}_1$ belongs to labels y_1, y_2, y_3 and y_4 simultaneously. Then, we can learn that $\mathbf{x}_1 \in y_1$ and $\mathbf{x}_1 \in \bar{y_1}$. Similarly, for $\mathbf{x}_2$, we have $\mathbf{x}_2 \in y_1$ and $\mathbf{x}_2 \in \bar{y_1}$, and details for other instances are summarized in Fig. 2. That means these examples associated with more than one class label can be viewed as not only positive examples of label y_1, but also negative ones. However, in each binary classification, the events $\{x_i \in y_j\}$ and $\{x_i \in \bar{y_j}\}$ are mutually exclusive. This problem is called inconsistency [31].

We notice that these binary classification models utilize identical feature representation for each instance to discriminate all the class labels. In Fig. 2, each instance $\mathbf{x}_i$ is represented by all the features of the data set, not only discriminative to label y_1, but also to other class labels ($\{y_2, y_3, y_4\}$). It can lead to inconsistency when an example belongs to several class labels simultaneously. In the original feature space, the problem of inconsistency will mislead the binary classifiers to learn suboptimal decision boundaries. When training the binary (one-vs-rest) classifier for label y_j, if the instances are represented by the features which are only discriminative to y_j, the problem of inconsistency could be alleviated.

In multi-label learning, each class label may be decided by some specific characteristics of its own. In [32], it proposes to utilize label specific features to represent instances for predicting the corresponding class label. The label specific features are exploited by conducting clustering analysis on the positive and negative examples, and the new features are represented by distances between the original instances and instances in the centers of positive and negative examples. It can be viewed as a feature space transformation method, and it does not consider label correlations.

Motivated by the work in [32], we propose to learn label specific features for each label, named LLSF (*Learning Label Specific Features*). We assume that each label is only associated with a *subset* of relevant features from the original feature set for a given data set. These relevant features can be viewed as *Label Specific* features and should be *discriminative* to the corresponding class label. In multi-label learning, labels often have correlations with each other, and we try to incorporate this information into LLSF. We assume that any two strongly correlated class labels can *share* more features with each other than two uncorrelated or weakly correlated ones. Fig. 3 shows the learning structure of LLSF. It exploits different feature sets for the discrimination of different labels. LLSF can be applied as a feature selection method for multi-label learning and a general strategy to improve multi-label classification algorithms comprising a number of binary classifiers. Experimental results show competitive performance of LLSF.

The rest of this paper is organized as follows. Section II reviews previous works on multi-label learning. Section III presents details of the proposed method LLSF. Experimental results on eight multi-label benchmark data sets are shown in section IV. Finally, we conclude the paper in section V.

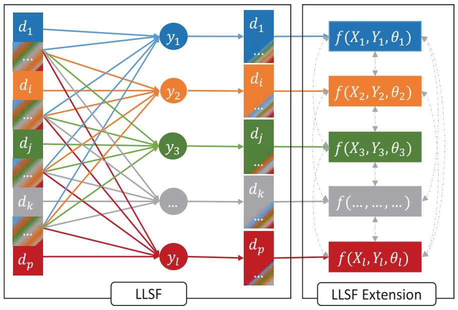


Fig. 3. Label specific features for each label are indicated in different colors, dashed arrows represent the output of one binary classifier can be used as input for the others, e.g., in CC, the predictions of preceding labels in the chain order will be augmented as features for discriminating subsequent labels.

II. Related work

In the past decades, many well-established methods have been proposed to solve multi-label learning problems in various domains. All these methods can be divided into two categories [12], [13]: Problem transformation methods (fitting data to algorithm) and algorithm adaption methods (fitting algorithm to data). Problem transformation methods transform a multi-label classification problem into one or more single-label classification (binary or multi-class classification) subproblems. Algorithm adaption methods modify traditional single-label learning algorithms for multi-label classification directly.

Binary Relevance (BR) [4] is a representative algorithm of problem transformation methods. The basic idea of BR is to decompose a multi-label learning problem into L independent binary (one-vs-rest) classification problems, where each binary classification problem corresponds to one label in the label space. The BR approach is a simple and straightforward way for multi-label learning, but it still has several disadvantages. First, it does not consider the correlation information among different labels, which is critical for many applications where the semantics conveyed by different labels are correlated. Second, the binary classifiers for each label may suffer from the issue of class-imbalance when the number of labels is large and label density is low. Third, BR approach can become computationally unaffordable for data sets with many labels. Last, as discussed in Section I, each binary classifier in BR suffers from the problem of inconsistency.

For the first problem, many methods have been proposed to solve multi-label classification by mining label correlations. Classifier Chains (CC) [14] is a novel chaining method that can model label correlations. The Classifier Chain model transforms the multi-label classification problem into a chain of binary classification problems. It involves l binary classifiers, and each binary classifier is trained one by one. The i-th classifier h_i is trained by using the results of labels $y_1, y_2, ..., y_{i-1}$ as additional input information. Ensemble Classifier Chains (EC-C) [14] is the ensemble framework of CC. The performance of CC is seriously constrained by the training order of labels and error propagation. On the other hand, it may not be appropriate that each label is dependent on all the preceding labels in a given order of labels. Probability Classifier Chains (PCC) [33] 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. Also, 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 and reducing the computational complexity. For example, PCC-beam [34] utilizes beam search to make inference tractable, and integrates beam search with training to determine a suitable order of labels. BCC [15] is a Bayesian Chain Classifiers for multi-label classification. A tree structured undirected label dependency graph is obtained from maximum weight undirected spanning tree, and each node in the tree has at most one parent node. OOCC [17] assigns a desirable order of labels to a new test instance which performs well in its k nearest neighbors in the training data . MCC [18] applies Monte Carlo technique to search the best order of labels at the training stage and predict label vector for each test instance in the inference stage. In each iteration of the training stage, the order which has lower loss will be accepted. In the inference stage, the predicting labels with the lowest exact-match loss will be accepted. LEAD [35] uses a bayesian network structure to encode the conditional dependencies of the labels as well as the feature set, with the feature set as the common parent of all labels. There are several works exploiting label correlations locally, such as ML-LOC [36] and GCC [37].

For the second problem, MLSC [23] is proposed to deal with concept drift and class imbalance in multi-label stream classification. k-Nearest Neighbor algorithm is utilized as an instantiation and a batch-incremental threshold technique is proposed to further deal with the class-imbalance problem. In [24], a novel inverse random under sampling (IRUS) method is proposed for the class-imbalance problem. The main idea is to severely under sample the majority class thus creating a large number of distinct training sets. For each training set, IRUS finds a decision boundary which separates the minority class from the majority class. By combining the multiple designs through fusion, IRUS constructs a composite boundary between the majority class and the minority class. COCOA [25] builds one binary-class imbalance learner for each label and also several multi-class imbalance learners coupling with other labels (random generated). The final prediction on each class label is obtained by aggregating the outputs yielded by the binary learner and the multi-class learners and the threshold is tuned by maximizing F-Measure.

For the third problem, multi-label data with large number of labels, the number of needed predictive models (eg. one-vs-rest) will be quite large, making the training costs unaffordable. To tackle this problem, recently academia has seen efforts made to perform label space dimension reduction (LSDR) [26]–[30]. For LSDR, each original high-dimensional label vector is encoded to a low dimensional code vector in a latent space. Then predictive models are trained from the data matrix to code vectors, whose quantity is much smaller and thus can significantly reduce the training costs. To predict an unseen example, a low-dimensional code vector is firstly obtained with the learnt predictive models on its features and then efficiently decoded for recovering its predicted label vector. If the learnt predictive models and the decoding process are effective enough, LSDR is expected to yield acceptable classification performance at much lower costs.

For the problem of inconsistency, Wang Hua [31] indicates that it can be avoided by using non-parametric classification methods, such as kNN. They propose a class balanced k nearest neighbor approach BKNN [31] for multi-label classification by emphasizing balanced usage of data from all class, and develop the classical linear discriminant analysis to reduce the dimensionality of multi-label data. In [32], LIFT is proposed, and it assumes that an example belongs to multiple labels simultaneously because each label is correlated with its corresponding features. LIFT exploits different features set for the discrimination of different labels by conducting clustering analysis on the positive and negative examples, and then performs training and testing by querying the clustering results. The label specific features of LIFT are represented by distances between the original instances and instances in the centers of positive and negative examples. LIFT can be viewed as a feature space transformation method, and it does not consider label correlations. Some feature selection, dimension reduction and subspace learning methods have been proposed for multi-label classification, such as [3], [38]–[41]. However, the learned low dimensional representations of these algorithms are still shared by all the class labels.

By surveying previous researches on multi-label classification, we realize that they are mostly constructed based on BR framework, and utilize identical feature representation to discriminate all the class labels, which, as analysed previously, may not well construct the optimal multi-label classification models. LIFT seems to be the only one which learns label specific features for multi-label classification, but it does not consider label correlations. Hence in this paper we propose LLSF, which not only learns label specific features, but also incorporates label correlations.

III. Learning Label Specific Features

In this section, we present how to model discriminant and sparsity of label specific features, and then introduce how to incorporate label correlation into this model. Last, we will give details of the proposed LLSF model and the accelerated approximal optimization method.

A. Notations

Let $\mathcal{X} = \mathbb{R}^p$ be the input space with p-dimensional and $\mathcal{Y} = \{y_1, y_2, ..., y_l\}$ be the finite set of l possible class labels. $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) | 1 \leqslant i \leqslant n\}$ is the training data set with n examples. The i-th example is denoted by a vector with m attribute values $\mathbf{x}_i = [\mathbf{x}_{i1}, \mathbf{x}_{i2}, ..., \mathbf{x}_{ip}]$, $\mathbf{x}_i \in \mathcal{X}$, and $\mathbf{y}_i = [\mathbf{y}_{i1}, \mathbf{y}_{i2}, ..., \mathbf{y}_{il}]$ is the ground truth labels of $\mathbf{x}_i$. Each element $\mathbf{y}_{ij} = 1$ if the label y_j is associated with $\mathbf{x}_i$, otherwise $\mathbf{y}_{ij} = 0$. We denote the input data as a matrix $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n \times p}$, and denote the output label matrix as $Y = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]^T \in \mathbb{R}^{n \times l}$.

B. Discriminant and Sparsity of Label Specific Features

As aforementioned, we assume that each class label is associated with a subset of features from the original feature set, and these features should be the most pertinent and discriminative to the corresponding class label. Contrasting to the original feature set, label specific features to each class label are sparse. We model the discriminant of label specific features by linear regression, and employ ℓ_1 norm on the regression parameters to model the sparsity of label specific features. The optimization problem is formulated as

$$\min_{W_i} \frac{1}{2}\|XW_i - Y_i\|_2^2 + \beta\|W_i\|_1 \tag{1}$$

where $W_i = [W_{i1}, W_{i2}, ..., W_{ip}]^T$ represents the regression parameter for the i-th label, and $Y_i = [\mathbf{y}_{1i}, \mathbf{y}_{2i}, ..., \mathbf{y}_{ni}]^T$ represents the i-th column of Y, $1 \leq i \leq l$.

If $W_{ij} = 0, 1 \leq j \leq p$, it indicates that the j-th feature has no effect on the discrimination of the i-th class label y_i. Therefore, the nonzero entities W_{ij} in each W_i indicate that the corresponding features are discriminative to label y_i, and these features can be considered as label specific features of label y_i, and the number of label specific features will be much smaller than p.

C. Incorporating Label Correlations

In multi-label learning, class labels often have correlations with each other. If two class labels are strongly correlated, features discriminative to one class label may be discriminative to another. On the contrary, features discriminative to one class label may not be discriminative to another if two class label are uncorrelated or weakly correlated. This indicates that two correlated labels can share more features in feature set than two uncorrelated or weakly correlated labels.

In LLSF, the label specific features are decided by nonzero entities in the coefficient vectors $W_i, 1 \leq i \leq l$. So, if labels y_i and y_j are strongly correlated, the similarity between W_i and W_j will be large, otherwise, the similarity will be small. After incorporating label correlations, the optimization is reformulated as

$$\min_{W_i} \frac{1}{2}\|XW_i - Y_i\|_2^2 + \frac{\alpha}{2}\sum_{j=1}^{l} R_{ij}{W_i}^T W_j + \beta\|W_i\|_1 \tag{2}$$

where $R_{ij} = 1 - C_{ij}$, and C_{ij} represents the correlation coefficient between labels y_i and y_j, and $C \in \mathbb{R}^{l\times l}$ is the label correlation matrix. In this paper, we utilize cosine similarity to calculate the correlation matrix of labels.

Considering all the binary classifiers simultaneously, the final optimization formulation can be written as

$$\min_{W} \frac{1}{2}\|XW - Y\|_F^2 + \frac{\alpha}{2}\text{Tr}(RW^T W) + \beta\|W\|_1 \tag{3}$$

where $W = [W_1, W_2, ..., W_l] \in \mathbb{R}^{p\times l}, R \in \mathbb{R}^{l\times l}$, $Y = [Y_1, Y_2, ..., Y_l] \in \mathbb{R}^{n\times l}$, $\alpha \geq 0$ and $\beta \geq 0$ are the model parameters.

D. Accelerated Proximal Gradient

Although the minimization problem (3) is a convex optimization problem, the objective function is non-smooth due to the non-smoothness of the ℓ_1 norm regularization term. We use the accelerated proximal gradient method to solve this non-smooth optimization problem.

In general accelerated proximal gradient method, given the following convex optimization problem

$$\min_{W\in\mathcal{H}} F(W) = f(W) + g(W) \tag{4}$$

where $\mathcal{H}$ is a real Hilbert space. Both $f(W)$ and $g(W)$ are convex and $f(W)$ is further Lipschitz continuous $\|\nabla f(W_1) - \nabla f(W_2)\| \leq L_f\|\Delta W\|$, where L_f is the Lipschitz constant.

Instead of directly minimizing $F(W)$, proximal gradient algorithms minimize a sequence of separable quadratic approximations to $F(W)$, denoted as $Q(W, W^{(t)})$.

$$\begin{aligned} Q(W, W^{(t)}) = f(W^{(t)}) + \langle\nabla f(W^{(t)}), W - W^{(t)}\rangle \\ + \tfrac{L_f}{2}\|W - W^{(t)}\|_F^2 + g(W) \end{aligned} \tag{5}$$

Let $G^{(t)} = W^{(t)} - \frac{1}{L_f}\nabla f(W^{(t)})$, then

$$\begin{aligned} W &= \arg\min\nolimits_W Q(W, W^{(t)}) \\ &= \arg\min\nolimits_W g(W) + \tfrac{L_f}{2}\|W - G^{(t)}\|_F^2 \end{aligned} \tag{6}$$

According to Eq. (3) and Eq. (4), $f(W)$ and $g(W)$ can be defined as

$$f(W) = \frac{1}{2}\|XW - Y\|_F^2 + \frac{\alpha}{2}\text{Tr}(RW^T W) \tag{7}$$

$$g(W) = \beta\|W\|_1 \tag{8}$$

E. Optimization of LLSF Model

According to Eq. (6), Eq. (7) and Eq. (8), the coefficient matrix W can be optimized by

$$\begin{aligned} W &= \arg\min\nolimits_W Q(W, W^{(t)}) \\ &= \arg\min\nolimits_W \tfrac{L_f}{2}\|W - G^{(t)}\|_F^2 + g(W) \\ &= \arg\min\nolimits_W \tfrac{1}{2}\|W - G^{(t)}\|_F^2 + \tfrac{\beta}{L_f}\|W\|_1 \end{aligned} \tag{9}$$

In [42], the work has shown that setting $W^{(t)} = W_t + \frac{b_{t-1}-1}{b_t}(W_t - W_{t-1})$ for a sequence b_t satisfying $b_{t+1}^2 - b_{t+1} \leq b_t^2$ can improve the convergence rate to $O(t^{-2})$, and W_t is the result of W at the t-th iteration.

Here $g(W)$ is ℓ_1 norm regularization, then in each iteration, W can be solved by the following optimization problem

$$W_{t+1} = S_\varepsilon[G^{(t)}] = \arg\min_W \varepsilon\|W\|_1 + \frac{1}{2}\|W - G^{(t)}\|_F^2 \tag{10}$$

where $S_\varepsilon[\cdot]$ is the soft-thresholding operator. For each element w_{ij} and $\varepsilon > 0$, the soft-thresholding operation is defined as

$$S_\varepsilon[w_{ij}] = \begin{cases} w_{ij} - \varepsilon & \text{if } w_{ij} > \varepsilon \\ w_{ij} + \varepsilon & \text{if } w_{ij} < -\varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

The overall accelerated proximal gradient algorithm of LLSF is summarized in Algorithm 1.

Algorithm 1: Learning Label Specific Features for Multi-Label Classification via Accelerated Proximal Gradient Method

```
Input: Training data matrix $X \in \mathbb{R}^{n \times p}$, label matrix
       $Y \in \mathbb{R}^{n \times l}$, and weighting parameters $\alpha, \beta, \gamma$;
Output: Model coefficient matrix $W \in \mathbb{R}^{p \times l}$.
Initialization:
$b_0, b_1 \leftarrow 1, W_0, W_1 \leftarrow (X^T X + \gamma I)^{-1} X^T Y$;
while not converged do
    $W^{(t)} \leftarrow W_t + \frac{b_{t-1}-1}{b_t}(W_t - W_{t-1})$;
    $G^{(t)} \leftarrow W^{(t)} - \frac{1}{L_f}\nabla f(W^{(t)})$;
    $W_{t+1} \leftarrow S_{\frac{\beta}{L_f}}(G^{(t)})$;
    $b_{t+1} \leftarrow \frac{1+\sqrt{4b_t^2+1}}{2}$;
    $t \leftarrow t + 1$;
```

Algorithm 2: Test of LLSF

```
Input: Test data matrix $X_{te} \in \mathbb{R}^{m \times p}$; Model coefficient
       matrix $W \in \mathbb{R}^{p \times l}$; Threshold $\tau$.
Output: Predict Label Matrix:$Y_{te}$; Score Matrix:$S_{te}$.
$S_{te} \leftarrow X_{te} W$;
$Y_{te} \leftarrow$ sign$(S_{te} - \tau)$;
```

F. Proof of Lipschitz Continuity

In this section, we will prove the Lipschitz continuity of Eq. (7). According to Eq. (7), we can calculate $\nabla f(W)$ as

$$\nabla f(W) = X^T X W - X^T Y + \alpha W R \tag{12}$$

Given W_1 and W_2, then we have

$$\begin{aligned} &\|\nabla f(W_1) - \nabla f(W_2)\|_F^2 \\ &\quad = \|X^T X \Delta W + \alpha \Delta W R\|_F^2 \\ &\quad \leq 2\|X^T X \Delta W\|_F^2 + 2\|\alpha \Delta W R\|_F^2 \\ &\quad \leq 2\|X^T X\|_2^2 \|\Delta W\|_F^2 + 2\|\alpha R\|_2^2 \|\Delta W\|_F^2 \\ &\quad = (2\|X^T X\|_2^2 + 2\|\alpha R\|_2^2)\|\Delta W\|_F^2 \\ &\quad = (2\sigma_{\max}^2(X^T X) + 2\sigma_{\max}^2(\alpha R))\|\Delta W\|_F^2 \end{aligned} \tag{13}$$

where $\Delta W = W_1 - W_2$, and $\sigma_{\max}(\cdot)$ indicates the maximum singular value of the matrix. Then we have

$$\begin{aligned} &\|\nabla f(W_1) - \nabla f(W_2)\|_F^2 \\ &\quad \leq (2\sigma_{\max}^2(X^T X) + 2\sigma_{\max}^2(\alpha R))\|\Delta W\|_F^2 \end{aligned} \tag{14}$$

Therefore, the Lipschitz constant is

$$L_f = \sqrt{2\sigma_{\max}^2(X^T X) + 2\sigma_{\max}^2(\alpha R)} \tag{15}$$

G. Testing

After training of LLSF, we can obtain the coefficient matrix W. It can be used as the parameters for linear regression directly. Given a test data represented by matrix X_{te}, the predict labels can be determined by sign$(S_{te} - \tau)$ with the given threshold τ, where $S_{te} = X_{te} W$. In our experiment, τ is set to be 0.5. The details of testing for LLSF can be summarized in Algorithm 2.

LLSF can be viewed as a feature selection method for multi-label classification, where features corresponding to the nonzero entities in W_i are considered as the label specific features to label y_i. It can be taken as the input of many multi-label classification algorithms comprising a number of binary classifiers, such as BSVM [4], CC [14], BCC [15], LIFT [32], CLR [43], etc. The procedures can be summarized in Algorithm 3.

Algorithm 3: LLSF as a Feature Selection Method

```
Input: Training data matrix $X \in \mathbb{R}^{n \times p}$,label matrix
       $Y \in \mathbb{R}^{n \times l}$, and weighting parameters $\alpha, \beta, \gamma$;
       Test data matrix $X_{te} \in \mathbb{R}^{m \times p}$; Binary classifier
       learning function $f(\cdot)$ and its parameters $\theta$;
Output: Predict Label Matrix:$Y_{te}$; Score Matrix:$S_{te}$.
Training:
Learning the model coefficient matrix $W$ of LLSF by
Algorithm 1;
for $i$ = 1 to $l$ do
    $id_i \leftarrow$ find$(W_i \neq 0)$;
    $X_{tr}^i \leftarrow X(:, id_i)$;
    $h_i \leftarrow f(X_{tr}^i, Y, \theta)$;
Testing:
for $i$ = 1 to $l$ do
    $X_{te}^i \leftarrow X_{te}(:, id_i)$;
    $\{Y_{te}^i, S_{te}^i\} \leftarrow h_i(X_{te}^i, Y_{te}, \theta)$;
$Y_{te} \leftarrow [Y_{te}^1, Y_{te}^2, ..., Y_{te}^l]$;
$S_{te} \leftarrow [S_{te}^1, S_{te}^2, ..., S_{te}^l]$;
```

IV. Experiments

A. Evaluation Metrics

To evaluate the performance of different algorithms for multi-label classification, we use seven common evaluation metrics [9], [12]–[14], [44] in multi-label classification to verify the performance. Given a test data set $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, where $\mathbf{y}_i \in \{0,1\}^l$ is the ground truth labels of the i-th example, and $\hat{\mathbf{y}}_i$ is its predicted labels.

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

$$\text{Hamming loss} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{l}\sum_{j=1}^{l}[\![\mathbf{y}_{ij} \neq \hat{\mathbf{y}}_{ij}]\!] \tag{16}$$

The smaller the value of Hamming loss, the better performance of the classifier, $[\![\cdot]\!]$ is an indication function.

Accuracy evaluates Jaccard similarity between the ground truth labels and the predicted labels.

$$\text{Accuracy} = \frac{1}{m}\sum_{i=1}^{m}\frac{|\mathbf{y}_i \wedge \hat{\mathbf{y}}_i|}{|\mathbf{y}_i \vee \hat{\mathbf{y}}_i|} \tag{17}$$

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

$$\text{Exact-Match} = \frac{1}{m}\sum_{i=1}^{m}[\![\mathbf{y}_i = \hat{\mathbf{y}}_i]\!] \tag{18}$$

Example based F_1-Measure is the integrated version of precision and recall for each example.

$$\mathrm{F}_1 = \frac{1}{m}\sum_{i=1}^{m}\frac{2p_i r_i}{p_i + r_i} \tag{19}$$

where p_i and r_i are the precision and recall for the i-th example.

Macro $\mathbf{F}_1$ is the integrated version of precision and recall for each label.

$$\text{Macro } \mathrm{F}_1 = \frac{1}{l}\sum_{i=1}^{l}\frac{2p_i r_i}{p_i + r_i} \tag{20}$$

where p_i and r_i are the precision and recall for the i-th 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.

$$\text{Micro } \mathrm{F}_1 = \frac{2\sum\limits_{j=1}^{l}\sum\limits_{i=1}^{m}\mathbf{y}_{ij}\hat{\mathbf{y}}_{ij}}{\sum\limits_{j=1}^{l}\sum\limits_{i=1}^{m}\mathbf{y}_{ij} + \sum\limits_{j=1}^{l}\sum\limits_{i=1}^{m}\hat{\mathbf{y}}_{ij}} \tag{21}$$

Average Precision evaluates the average fraction of relevant labels ranked higher than a particular label $y_j \in Y_i$.

$$\text{AvgPrecison} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{|\mathbf{y}_i|}\sum_{y_j \in \mathbf{y}_i}\frac{|\mathcal{R}(\mathbf{x}_i, y_j)|}{rank(\mathbf{x}_i, y_j)} \tag{22}$$

where $\mathcal{R}(\mathbf{x}_i, y_j) = \{y_q | rank(\mathbf{x}_i, y_q) \leq rank(\mathbf{x}_i, y_j), y_q \in \mathbf{y}_i\}$, and $rank(\mathbf{x}_i, y_j)$ indicates the rank of y_j for $\mathbf{x}_i$.

The above evaluation metrics include two types: example-based and label-based metrics [13]. The larger the value of them, the better the performance of the classifier for all of these metrics except hamming loss.

B. Comparison methods

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

BSVM [4]: It decomposes the multi-label classification problem into l independent binary (one-vs-rest) classification subproblems. In each binary classification problem, examples associating with a given class label are considered as positive examples, while those without this class label are considered as negative examples.

Classifier Chains (CC) [14]: CC transforms the multi-label learning problem into a chain of l binary classification problems. The chains order $y_{\pi(1)}, y_{\pi(2)}, ..., y_{\pi(l)}$ is generated randomly. Each binary classifier is trained one by one, and classifier $h_{\pi(i)}$ is trained by using $y_{\pi(1)}, ..., y_{\pi(i-1)}$ as additional features.

LIFT[1] [32]: It exploits different features set for the discrimination of different labels by conducting clustering analysis on the positive and negative instances. The ratio parameter r is set to be 0.1 for all data sets, which is suggested in [32].

TABLE I. DESCRIPTION OF DATA SETS

Data set	#Instance	#Feature	#Label	Cardinality	Domain
genbase	645	1186	27	1.252	biology
medical	978	1449	45	1.245	text
bibtex	7395	1836	159	2.402	text
rcv1(subset1)	6000	944	101	2.88	text
rcv1(subset2)	6000	944	101	2.634	text
rcv1(subset3)	6000	944	101	2.614	text
rcv1(subset4)	6000	944	101	2.484	text
rcv1(subset5)	6000	944	101	2.642	text

BKNN [31]: It is a class balanced k nearest neighbor approach B*K*NN [31] for multi-label classification by emphasizing balanced usage of data from all classes, and develop the classical linear discriminant analysis to reduce the dimensionality of multi-label data. Parameter α and b are set to be 0.5 and 5 respectively, which is suggested in [31]. The threshold h_i^{opt} for each label y_i is tuned from 0 to 1 stepped by 0.01 on training data by optimizing

$$h_i^{\text{opt}} = \arg\max_{h_i} \frac{2}{\frac{\alpha}{p_i(h_i)} + \frac{1-\alpha}{r_i(h_i)}} \tag{23}$$

where $p_i(h_i)$ and $p_i(h_i)$ indicate the precision and recall for the i-th label given threshold h_i.

LLSF[2] : The proposed method in this paper, which learns label specific features for multi-label classification. Parameters α, β and γ are set to be 0.1, 0.1 and 0.01 respectively. The threshold τ is simply set to be 0.5. The Lipschitz constant L_f is calculated by Eq. (15).

LLSF-BSVM: LLSF is utilized as a feature selection method. Features corresponding to the nonzero entities in each $W_i (1 \leq i \leq l)$ are considered as label specific features of label y_i. The data feature matrix composed of these label specific features of each label y_i is set to be the training data for each binary classifier of BSVM. Parameters α, β and γ for LLSF are set to be 0.5, 0.1 and 0.01 respectively.

LLSF-CC: It is an extension of LLSF-BSVM. The chains order $y_{\pi(1)}, y_{\pi(2)}, ..., y_{\pi(l)}$ is generated randomly. Each classifier $h_{\pi(i)}$ is trained by using $y_{\pi(1)}, y_{\pi(2)}, ..., y_{\pi(i-1)}$ as augmented features with the label specific features of label $y_{\pi(i)}$. Parameters α, β and γ for LLSF are set to be 0.5, 0.1 and 0.01 respectively.

For fair comparison, we utilize libsvm [45] as binary learner for each binary (one-vs-rest) classifier for BSVM, CC, LIFT, LLSF-SVM and LLSF-CC, where we set the kernel function as linear kernel, and set the parameter C as 1.

C. Data sets

We conduct experiments on eight multi-label benchmark data sets, and the detailed characteristics of these data sets are summarized in Table I. All of these data sets can be downloaded from mulan[3] and lamda[4]. The column named "Cardinality" means label cardinality of the data set. It indicates the average number of labels per example.

[1] Code of LIFT is available at: http://cse.seu.edu.cn/people/zhangml

[2] The programme code of LLSF, LLSF-BSVM and LLSF-CC are public available at https://github.com/JunHuangUCAS/LLSF

[3] http://mulan.sourceforge.net/datasets.html

[4] http://lamda.nju.edu.cn/Data.ashx#data

TABLE II. Experimental results of each comparing algorithm (mean±std rank) on eight data sets in terms of Hamming Loss. The smaller the value the better the performance.

Data set	BKNN		LIFT		LLSF		BSVM		LLSF-BSVM		CC		LLSF-CC	
cgenbase	0.003±0.001	6.5	0.003±0.001	6.5	**0.001±0.000**	**3**	**0.001±0.001**	**3**	**0.001±0.000**	**3**	**0.001±0.000**	**3**	**0.001±0.000**	**3**
medical	0.023±0.002	7	0.011±0.002	6	0.010±0.001	3.5	0.010±0.002	3.5	**0.009±0.001**	**1**	0.010±0.001	3.5	0.010±0.002	3.5
bibtex	0.029±0.001	7	0.012±0.000	4	0.012±0.000	4	0.012±0.000	4	**0.011±0.000**	**1**	0.012±0.000	4	0.012±0.000	4
rcv1(subset1)	0.043±0.005	7	**0.026±0.000**	**2.5**	**0.026±0.001**	**2.5**	**0.026±0.001**	**2.5**	**0.026±0.001**	**2.5**	0.028±0.000	5.5	0.028±0.000	5.5
rcv1(subset2)	0.044±0.006	7	**0.023±0.001**	**2.5**	**0.023±0.001**	**2.5**	**0.023±0.000**	**2.5**	**0.023±0.001**	**2.5**	0.025±0.001	5.5	0.025±0.001	5.5
rcv1(subset3)	0.043±0.003	7	**0.023±0.001**	**2.5**	**0.023±0.000**	**2.5**	**0.023±0.000**	**2.5**	**0.023±0.001**	**2.5**	0.024±0.001	5	0.025±0.001	6
rcv1(subset4)	0.046±0.007	7	0.019±0.000	4.5	**0.018±0.000**	**2**	**0.018±0.001**	**2**	**0.018±0.000**	**2**	0.020±0.001	6	0.019±0.001	4.5
rcv1(subset5)	0.046±0.008	7	0.023±0.000	3.5	**0.022±0.000**	**1.5**	**0.022±0.000**	**1.5**	0.023±0.000	3.5	0.024±0.000	5.5	0.024±0.001	5.5
Avg.Rank	6.938	7	4	4	2.688	2.5	2.688	2.5	2.25	1	4.75	6	4.688	5
Total Order	LLSF-BSVM ≻ LLSF ⪰ BSVM ≻ LIFT ≻ LLSF-CC ≻ CC ≻ BKNN													

TABLE III. Experimental results of each comparing algorithm (mean±std rank) on eight data sets in terms of accuracy. The larger the value the better the performance.

Data set	BKNN		LIFT		LLSF		BSVM		LLSF-BSVM		CC		LLSF-CC	
genbase	0.967±0.014	6	0.966±0.014	7	**0.993±0.006**	**1**	0.985±0.007	4.5	0.988±0.004	2.5	0.985±0.008	4.5	0.988±0.005	2.5
medical	0.520±0.036	7	0.688±0.044	6	0.760±0.022	2	0.737±0.039	5	0.753±0.018	3	0.746±0.022	4	**0.769±0.044**	**1**
bibtex	0.219±0.009	7	0.316±0.003	5	0.299±0.014	6	0.346±0.012	3	**0.373±0.007**	**1**	0.324±0.009	4	0.358±0.008	2
rcv1(subset1)	0.267±0.030	4	0.238±0.008	7	0.250±0.013	5	0.244±0.010	6	0.293±0.014	3	0.331±0.008	2	**0.355±0.008**	**1**
rcv1(subset2)	0.255±0.034	7	0.298±0.020	5.5	0.298±0.010	5.5	0.317±0.007	4	0.360±0.015	3	0.411±0.013	2	**0.432±0.008**	**1**
rcv1(subset3)	0.233±0.023	7	0.286±0.018	6	0.292±0.009	5	0.309±0.005	4	0.346±0.018	3	0.412±0.015	2	**0.425±0.008**	**1**
rcv1(subset4)	0.203±0.023	7	0.361±0.006	6	0.396±0.012	5	0.409±0.016	4	0.429±0.004	3	0.504±0.018	2	**0.516±0.008**	**1**
rcv1(subset5)	0.234±0.041	7	0.293±0.010	6	0.300±0.011	5	0.312±0.011	4	0.353±0.006	3	0.414±0.012	2	**0.425±0.020**	**1**
Avg.Rank	6.5	7	6.063	6	4.313	4.5	4.313	4.5	2.688	2	2.813	3	1.313	1
Total Order	LLSF-CC ≻ LLSF-BSVM ≻ CC ≻ LLSF ⪰ BSVM ≻ LIFT ≻ BKNN													

TABLE IV. Experimental results of each comparing algorithm (mean±std rank) on eight data sets in terms of exact-match. The larger the value the better the performance.

Data set	BKNN		LIFT		LLSF		BSVM		LLSF-BSVM		CC		LLSF-CC	
genbase	0.933±0.029	7	0.946±0.021	6	**0.986±0.011**	**1**	0.968±0.016	5	0.976±0.011	3.5	0.977±0.008	2	0.976±0.011	3.5
medical	0.322±0.074	7	0.616±0.077	6	0.667±0.023	4	0.660±0.037	5	0.669±0.028	3	0.672±0.025	2	**0.696±0.055**	**1**
bibtex	0.092±0.007	7	0.168±0.006	6	0.175±0.013	5	0.196±0.009	3.5	0.211±0.003	2	0.196±0.005	3.5	**0.221±0.007**	**1**
rcv1(subset1)	0.011±0.009	7	0.073±0.007	6	0.077±0.013	5	0.079±0.007	4	0.101±0.011	3	0.215±0.007	2	**0.226±0.008**	**1**
rcv1(subset2)	0.011±0.006	7	0.168±0.018	6	0.184±0.009	5	0.208±0.009	4	0.216±0.014	3	0.328±0.004	2	**0.340±0.005**	**1**
rcv1(subset3)	0.005±0.008	7	0.168±0.020	6	0.182±0.006	5	0.203±0.004	3.5	0.203±0.017	3.5	0.328±0.018	2	**0.336±0.016**	**1**
rcv1(subset4)	0.001±0.002	7	0.238±0.008	6	0.287±0.012	5	0.297±0.016	4	0.308±0.011	3	0.422±0.022	2	**0.427±0.007**	**1**
rcv1(subset5)	0.006±0.008	7	0.167±0.008	6	0.174±0.011	5	0.192±0.013	4	0.200±0.005	3	0.324±0.010	2	**0.326±0.020**	**1**
Avg.Rank	7	7	6	6	4.375	5	4.125	4	3	3	2.188	2	1.313	1
Total Order	LLSF-CC ≻ CC ≻ LLSF-BSVM ≻ BSVM ≻ LLSF ≻ LIFT ≻ BKNN													

TABLE V. Experimental results of each comparing algorithm (mean±std rank) on eight data sets in terms of Example-based F_1 Measure. The larger the value the better the performance.

Data set	BKNN		LIFT		LLSF		BSVM		LLSF-BSVM		CC		LLSF-CC	
genbase	0.976±0.010	6	0.971±0.013	7	**0.995±0.005**	**1**	0.990±0.006	4	0.991±0.003	2.5	0.987±0.008	5	0.991±0.003	2.5
medical	0.587±0.028	7	0.712±0.036	6	0.791±0.023	2	0.763±0.041	5	0.780±0.019	3	0.771±0.021	4	**0.793±0.041**	**1**
bibtex	0.280±0.011	7	0.374±0.003	4	0.350±0.013	6	0.403±0.013	3	**0.432±0.007**	**1**	0.373±0.010	5	0.408±0.008	2
rcv1(subset1)	0.380±0.030	2.5	0.303±0.011	7	0.318±0.013	5	0.309±0.011	6	0.366±0.017	4	0.380±0.010	2.5	**0.408±0.009**	**1**
rcv1(subset2)	0.368±0.038	4	0.349±0.021	6	0.343±0.012	7	0.361±0.006	5	0.415±0.016	3	0.444±0.017	2	**0.466±0.010**	**1**
rcv1(subset3)	0.338±0.027	5	0.335±0.018	7	0.337±0.010	6	0.352±0.006	4	0.404±0.018	3	0.444±0.016	2	**0.460±0.006**	**1**
rcv1(subset4)	0.308±0.026	7	0.408±0.008	6	0.439±0.013	5	0.451±0.015	4	0.475±0.003	3	0.536±0.018	2	**0.549±0.010**	**1**
rcv1(subset5)	0.342±0.045	7	0.345±0.012	6	0.352±0.011	5	0.360±0.010	4	0.413±0.006	3	0.449±0.013	2	**0.463±0.022**	**1**
Avg.Rank	5.688	6	6.125	7	4.625	5	4.375	4	2.813	2	3.063	3	1.313	1
Total Order	LLSF-CC ≻ LLSF-BSVM ≻ CC ≻ BSVM ≻ LLSF ≻ BKNN ≻ LIFT													

TABLE VI. Experimental results of each comparing algorithm (mean±std rank) on eight data sets in terms of Macro F_1. The larger the value the better the performance.

Data set	BKNN		LIFT		LLSF		BSVM		LLSF-BSVM		CC		LLSF-CC	
genbase	0.667±0.043	5	0.612±0.029	6	**0.777±0.040**	**1**	0.679±0.072	4	0.734±0.019	2	0.555±0.037	7	0.726±0.013	3
medical	0.294±0.029	6	0.251±0.022	7	**0.356±0.022**	**1**	0.339±0.031	4	0.347±0.005	2.5	0.324±0.012	5	0.347±0.032	2.5
bibtex	0.249±0.007	6	0.286±0.006	4	0.198±0.011	7	0.304±0.012	2	**0.332±0.009**	**1**	0.260±0.006	5	0.301±0.006	3
rcv1(subset1)	**0.320±0.004**	**1**	0.193±0.010	4	0.129±0.009	7	0.184±0.012	6	0.212±0.017	2.5	0.190±0.011	5	0.212±0.015	2.5
rcv1(subset2)	**0.290±0.010**	**1**	0.174±0.017	5	0.107±0.009	7	0.172±0.008	6	0.215±0.010	3	0.205±0.009	4	0.233±0.009	2
rcv1(subset3)	**0.287±0.020**	**1**	0.163±0.013	5	0.103±0.003	7	0.154±0.006	6	0.205±0.011	3	0.189±0.010	4	0.209±0.014	2
rcv1(subset4)	**0.265±0.010**	**1**	0.160±0.012	6	0.102±0.003	7	0.167±0.009	5	0.200±0.006	3	0.181±0.011	4	0.217±0.012	2
rcv1(subset5)	**0.304±0.011**	**1**	0.182±0.011	5	0.105±0.004	7	0.169±0.009	6	0.240±0.003	2	0.202±0.013	4	0.235±0.017	3
Avg.Rank	2.75	3	5.25	6	5.5	7	4.875	5	2.375	1	4.75	4	2.5	2
Total Order	LLSF-BSVM ≻ LLSF-CC ≻ BKNN ≻ CC ≻ BSVM ≻ LIFT ≻ LLSF													

TABLE VII. EXPERIMENTAL RESULTS OF EACH COMPARING ALGORITHM (MEAN±STD RANK) ON EIGHT DATA SETS IN TERMS OF MICRO F_1. THE LARGER THE VALUE THE BETTER THE PERFORMANCE.

Data set	BKNN	LIFT	LLSF	BSVM	LLSF-BSVM	CC	LLSF-CC
genbase	0.967±0.015 7	0.970±0.013 6	**0.994±0.005 1**	0.985±0.009 5	0.990±0.005 3	0.990±0.004 3	0.990±0.005 3
medical	0.626±0.022 7	0.785±0.033 6	0.816±0.016 2	0.805±0.036 4	**0.818±0.017 1**	0.803±0.019 5	0.814±0.034 3
bibtex	0.286±0.005 7	0.432±0.007 4	0.402±0.013 6	0.458±0.012 3	**0.481±0.013 1**	0.426±0.006 5	0.461±0.010 2
rcv1(subset1)	**0.418±0.030 1**	0.352±0.012 7	0.365±0.014 5	0.355±0.010 6	0.405±0.020 3	0.388±0.008 4	0.414±0.012 2
rcv1(subset2)	0.392±0.037 4	0.383±0.019 6	0.373±0.015 7	0.385±0.007 5	0.433±0.008 2	0.418±0.016 3	**0.446±0.012 1**
rcv1(subset3)	0.373±0.023 5	0.370±0.019 6	0.367±0.009 7	0.377±0.005 4	0.423±0.020 2	0.420±0.008 3	**0.437±0.011 1**
rcv1(subset4)	0.336±0.028 7	0.419±0.010 6	0.443±0.010 5	0.454±0.016 4	0.473±0.004 3	0.481±0.016 2	**0.499±0.010 1**
rcv1(subset5)	0.375±0.042 7	0.386±0.011 6	0.392±0.009 5	0.394±0.012 4	0.439±0.007 2	0.432±0.012 3	**0.454±0.017 1**
Avg.Rank	5.625 6	5.875 7	4.75 5	3.625 4	2.125 2	3.5 3	1.75 1
Total Order	LLSF-CC $\succ$ LLSF-BSVM $\succ$ CC $\succ$ BSVM $\succ$ LLSF $\succ$ BKNN $\succ$ LIFT						

TABLE VIII. EXPERIMENTAL RESULTS OF EACH COMPARING ALGORITHM (MEAN±STD RANK) ON EIGHT DATA SETS IN TERMS OF AVERAGE PRECISION. THE LARGER THE VALUE THE BETTER THE PERFORMANCE.

Data set	BKNN	LIFT	LLSF	BSVM	LLSF-BSVM	CC	LLSF-CC
genbase	0.985±0.007 7	**0.996±0.004 2.5**	**0.996±0.004 2.5**	**0.996±0.003 2.5**	0.994±0.003 5.5	0.996±0.004 2.5	0.994±0.003 5.5
medical	0.734±0.016 7	0.887±0.017 4.5	**0.901±0.016 1**	0.897±0.029 3	0.898±0.012 2	0.881±0.017 6	0.887±0.028 4.5
bibtex	0.552±0.013 7	0.586±0.002 5	**0.609±0.017 1**	0.588±0.015 4	0.604±0.013 2	0.566±0.008 6	0.590±0.014 3
rcv1(subset1)	**0.636±0.011 1**	0.605±0.006 2	0.603±0.010 3	0.589±0.008 5	0.591±0.008 4	0.529±0.009 7	0.544±0.009 6
rcv1(subset2)	**0.638±0.007 1**	0.624±0.011 3	0.630±0.007 2	0.617±0.006 5	0.619±0.011 4	0.570±0.011 6	0.566±0.011 7
rcv1(subset3)	**0.636±0.011 1**	0.617±0.012 3.5	0.627±0.008 2	0.611±0.007 5	0.617±0.017 3.5	0.562±0.013 7	0.577±0.010 6
rcv1(subset4)	0.687±0.008 4	0.680±0.006 5	**0.712±0.004 1**	0.691±0.006 2	0.689±0.005 3	0.643±0.014 7	0.659±0.005 6
rcv1(subset5)	**0.645±0.009 1**	0.625±0.010 3	0.639±0.006 2	0.617±0.009 5	0.620±0.008 4	0.576±0.008 7	0.583±0.014 6
Avg.Rank	3.625 4	3.563 3	1.813 1	3.938 5	3.438 2	6.063 7	5.5 6
Total Order	LLSF $\succ$ LLSF-BSVM $\succ$ LIFT $\succ$ BKNN $\succ$ BSVM $\succ$ LLSF-CC $\succ$ CC						

D. Multi-Label Classification Results

With the given parameters, we then repeatedly run each method 5 times on 5 sets of randomly partitioned training (80%) and test (20%) data. Tables II to VIII report the average results of each compared algorithm on these eight data sets in terms of different evaluation metrics. Each result is composed of $mean$, std and $rank$. When two or more algorithms obtain the same performance on one data set for a given evaluation metric, the value of $rank$ for these algorithms are assigned with the average result of them.

Previous works suggest that Hamming loss does not require modeling dependencies between labels. As shown in Table II, LLSF-CC and CC are inferior to many of the compared methods in terms of Hamming Loss. However, in case of Exact-Match, we can see that LLSF-CC and CC achieve better performance than all the compared methods. From Table VI, we can see that BKNN obtains good performance on Macro F_1, which is better than CC, BSVM, LIFT and LLSF. The optimal threshold of BKNN is tuned according to the mixture results of precision and recall for each label, which is calculated by Eq. (23). Essentially, BKNN tries to optimize Macro F_1.

We further observe that LLSF-CC achieves better or tied performance than LLSF-BSVM for data sets genbase and medical (the number of class labels of them are smaller than 50) in most cases, but inferior to LLSF-BSVM for the other data sets (the number of class labels of them are larger than 100, see Table I). LLSF-CC is a classifier chain model, where each binary classifier adds predictions of preceding labels in the chain as additional features with the label specific features. However, the label specific features are sparse after the processing of LLSF, and the number of label specific features is much smaller than p. So the percentage of label specific features to the label spaces is smaller than before. If the predictions of preceding labels in the chain are wrong, these incorrect predictions will mislead LLSF-CC when predicting subsequent labels in the chain order.

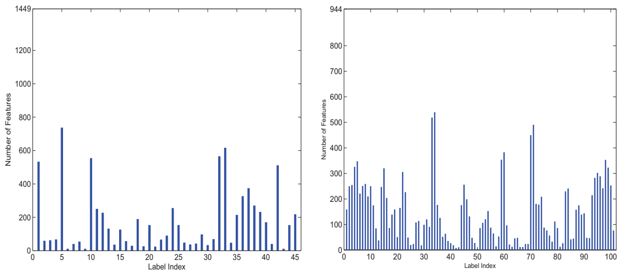
TABLE IX. RESULTS OF PAIRWISE COMPARISON APPLIED TO LLSF AND LLSF EXTENDED METHODS WITH OTHER COMPARED METHODS.

Algorithm	BKNN	LIFT	BSVM	CC	Total
LLSF	6 / 0 / 1	6 / 0 / 1	1 / 2 / 4	2 / 0 / 5	15 / 2 / 11
LLSF-BSVM	7 / 0 / 0	7 / 0 / 0	7 / 0 / 0	6 / 0 / 1	27 / 0 / 1
LLSF-CC	6 / 0 / 1	5 / 0 / 2	5 / 0 / 2	7 / 0 / 0	23 / 0 / 5

From Table II to Table VIII, we can summarize the overall pairwise comparison results of LLSF and LLSF extended methods with other compared methods as Table IX. Each cell is composed of three numbers: from left to right, how many times that LLSF or LLSF extended method is better / tied / worse than the compared method in terms of the seven evaluation metrics. For example, in the second row and the second column, "6 / 0 / 1" means that LLSF wins BKNN 6 times, ties 0 time and loses 1 time. As can be observed from Table IX, first of all, LLSF obtains better performance than the other compared algorithm overall. Especially, LLSF achieves better performance than BKNN and LIFT in terms of the seven evaluation metrics except for Macro F_1. It indicates that LLSF can solve the problem of inconsistency better than BKNN and LIFT. Moreover, LLSF-BSVM and LLSF-CC always obtain better performance than BSVM and CC on these eight data sets in terms of each evaluation metric respectively. It demonstrates the effectiveness of label specific features which are learned by LLSF. These results confirm that the proposed method is superior across different data sets and different evaluation metrics in a statistical sense.

E. Coefficient Matrix W of LLSF

As discussed in Section I and Section III, in multi-label classification, we assume that each label is associated with a subset of relevant features from the original feature set, and two strongly correlated class labels can share more features than two uncorrelated or weakly correlated ones. In LLSF model, for each $W_i = [W_{1i}, W_{2i}, ..., W_{pi}]^T, 1 \leq i \leq l$, if an element in W_i is nonzero, it indicates the corresponding feature



(a) result of medical (b) result of rcv1(subset1)

Fig. 4. The number of label specific features which is learned by LLSF for each class label with parameters $\alpha = 0.5, \beta = 0.1$ and $\gamma = 0.01$.

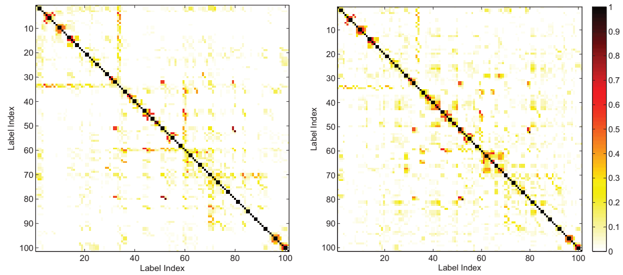
is discriminative to label y_i. So the corresponding features to the nonzero entities in each W_i are considered as label specific features of label y_i, and $\|W_i\|_0$ represents the number of label specific features of label y_i.

We count the number of label specific features for each class label learned by LLSF on two data sets: medical and rcv1(subset1), and the results are shown in Fig. 4. The horizontal axis of each sub-figure indicates the index of class label, e.g., number 10 indicates the 10-th class label y_{10}. The vertical axis of each sub-figure indicates the number of label specific features of each class label. We find that each label is only associated with a subset of relevant features from the original features set. The larger the value of α and β, the more sparse of the label specific features for each label. The experimental results shown in Section IV-D demonstrate that these label specific features are strongly discriminative to the corresponding class label.

On the other hand, we assume that two strongly correlated class labels can share more features than two uncorrelated or weakly correlated ones. In our experiment, the affinity matrix of Y and W are calculated by cosine similarity. The elements are defined as $C_{ij}^Y = \frac{<Y_i,Y_j>}{\|Y_i\|\times\|Y_j\|}$ and $C_{ij}^W = \frac{<W_i,W_j>}{\|W_i\|\times\|W_j\|}$, where Y_i and W_i are the i-th column of label matrix Y and coefficient matrix W respectively. So if label y_i and y_j are strongly correlated, then the corresponding coefficients W_i and W_j will be very similar, and the cosine similarity between them will be large. If label y_i and y_j are weakly correlated or uncorrelated, they will share less features, and the similarity between the corresponding coefficients W_i and W_j will be small.

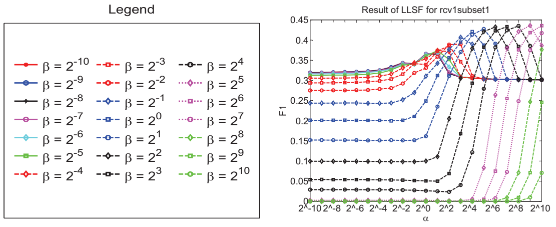
Fig. 5 shows the affinity matrices of training label matrix Y and the learned coefficient matrix W for data rcv1(subset1). The horizontal axis and vertical axis of each sub-figure indicate the index of each label. From Fig.5, we can see that the affinity matrix of W is surprisingly consistent with the affinity matrix of Y. That means if the similarity between Y_i and Y_j in the training data is large, then the similarity between the learned coefficients W_i and W_j will be large. This experimental result verifies the correctness of the hypothesis that two strongly correlated class labels can share more features than two uncorrelated or weakly correlated ones.

F. Parameter Sensitivity Analysis



(a) affinity matrix of label matrix (b) affinity matrix of coefficient matrix

Fig. 5. Label and coefficient affinity matrix of rcv1(subset1): The left sub-figure shows the label affinity matrix which is calculated on the training label matrix, and the right sub-figure shows the coefficient affinity matrix which is calculated on the coefficient matrix W which is learned by LLSF with parameters $\alpha = 0.5, \beta = 0.1$ and $\gamma = 0.01$.



(a) Legend (b) F_1 of LLSF

Fig. 6. Parameter sensitivity analysis on rcv1(subset1)

We conduct parameter sensitivity analysis for LLSF on rcv1(subset1) data set over the trade-off parameters α and β, where α controls the correlation between labels and β controls the sparsity of label specific features, and they are searched from $\{2^{-10}, 2^{-9}, ..., 2^{9}, 2^{10}\}$. The averaged F_1 over 5-fold cross validation for different values of α and β are depicted in Fig. 6. We observe that the performance of LLSF are first improved and then degraded with the increasing of α, and the results of LLSF is degraded with the increasing of β. It is suggested to set $\alpha \geq \beta$.

V. Conclusion

In this paper, we propose to solve the inconsistency problem by learning label specific features(LLSF) for multi-label classification. We assume that each label is associated with a subset of relevant features from the original feature set, and any two strongly correlated class labels can share more features with each other than two uncorrelated or weakly correlated ones. Then, instances which are represented by these label specific features will be only discriminative to the corresponding class label, and the problem of inconsistency could be alleviated.

Our proposed method LLSF can be utilized as a feature selection method for multi-label learning. The learned label specific features for each label can be applied as input to existing multi-label classification algorithms comprising a number of binary classifiers. We compare our method LLSF and its extended versions with several well-established multi-label classification algorithms over eight multi-label benchmark data sets. Comparison results manifest competitive performance of our proposed method, and verify the correctness of the hypotheses of our proposed method.

Acknowledgments

This work was supported in part by National Basic Research Program of China (973 Program): 2012CB316400 and 2015CB351802, in part by the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the Ministry of Education, China: IRT13059, and in part by National Natural Science Foundation of China: 61303153, 61332016, 61025011 and 61229301.

References

[1] N. Ueda and K. Saito, "Parametric mixture models for multi-labeled text," in *NIPS*, 2003, pp. 721–728.

[2] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda, "Maximal margin labeling for multi-topic text categorization," in *NIPS*, 2005, pp. 649–656.

[3] K. Yu, S.-P. Yu, and V. Tresp, "Multi-label informed latent semantic indexing," in *ACM SIGIR*, 2005, pp. 258–265.

[4] M. R. Boutell, J.-B. Luo, X.-P. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[5] Y. Luo, D.-C. Tao, C. Xu, D.-C. Li, and C. Xu, "Vector-valued multi-view semi-supervised learning for multi-label image classification," in *AAAI*, 2013, pp. 647–653.

[6] F.-M. Sun, J.-H. Tang, H.-J. Li, G.-J. Qi, and T. S. Huang, "Multi-label image categorization with sparse factor representation," *Image Processing, IEEE Transactions on*, vol. 23, no. 3, pp. 1028–1037, 2014.

[7] F. Kang, R. Jin, and R. Sukthankar, "Correlated label propagation with application to multi-label learning," in *CVPR*, 2006, pp. 1719–1726.

[8] G.-J. Qi, X.-S. Hua, Y. Rui, J.-H. Tang, T. Mei, and H.-J. Zhang, "Correlative multi-label video annotation," in *ACM MM*, 2007, pp. 17–26.

[9] X. Wang and G. Sukthankar, "Multi-label relational neighbor classification using social context features," in *KDD*, 2013, pp. 464–472.

[10] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. P. Vlahavas, "Multi-label classification of music into emotions," in *International Society for Music Information Retrieval*, 2008, pp. 325–330.

[11] B. Wu, E.-H. Zhong, A. Horner, and Q. Yang, "Music emotion recognition by multi-label multi-layer multi-instance multi-view learning," in *ACM MM*, 2014, pp. 117–126.

[12] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," *Data Mining and Knowledge Discovery Handbook*, pp. 667–685, 2010.

[13] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 26, no. 8, pp. 1819–1837, 2014.

[14] R. Jesse, P. Bernhard, H. Geoff, and F. Eibe, "Classifier chains for multi-label classification," in *ECML*, 2009, pp. 254–269.

[15] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, and P. Larrañaga, "Bayesian chain classifiers for multidimensional classification," in *IJCAI*, 2011, pp. 2192–2197.

[16] Y.-H. Guo and S.-C. Gu, "Multi-label classification using conditional dependency networks," in *IJCAI*, 2011, pp. 1300–1305.

[17] P. Da Silva, E. Gonçalves, A. Plastino, and A. A. Freitas, "Distinct chains for different instances: an effective strategy for multi-label classifier chains," in *ECMLPKDD*, 2014, pp. 453–468.

[18] J. Read, L. Martino, and D. Luengo, "Efficient monte carlo methods for multi-dimensional learning with classifier chains," *Pattern Recognition*, vol. 47, no. 3, pp. 1535 – 1546, 2014.

[19] X. Li, F.-P. Zhao, and Y.-H. Guo, "Multi-label image classification with a probabilistic label enhancement model," in *UAI*, 2014.

[20] K. Dembczyński, W. Waegeman, W.-W. Cheng, and E. Hüllermeier, "On label dependence and loss minimization in multi-label classification," *Machine Learning*, vol. 88, no. 1-2, pp. 5–45, 2012.

[21] B. Fu, G.-D. Xu, Z.-H. Wang, and L.-B. Cao, "Leveraging supervised label dependency propagation for multi-label learning," in *ICDM*, 2013, pp. 1061–1066.

[22] L. li Xu, Z. Wang, Z.-F. Shen, Y.-B. Wang, and E.-H. Chen, "Learning low-rank label correlations for multi-label classification with missing labels," in *ICDM*, 2014, pp. 1067–1072.

[23] E. S. Xioufis, M. Spiliopoulou, G. Tsoumakas, and I. Vlahavas, "Dealing with concept drift and class imbalance in multi-label stream classification," in *IJCAI*, 2011, pp. 1583–1588.

[24] M. A. Tahir, J. Kittler, and F. Yan, "Inverse random under sampling for class imbalance problem and its application to multi-label classification," *Pattern Recognition*, vol. 45, no. 10, pp. 3738 – 3750, 2012.

[25] M.-L. Zhang, Y.-K. Li, and X.-Y. Liu, "Towards class-imbalance aware multi-label learning," in *IJCAI*, 2015.

[26] J. Langford, T. Zhang, D. J. Hsu, and S. M. Kakade, "Multi-label prediction via compressed sensing," in *NIPS*, 2009, pp. 772–780.

[27] Y.-N. Chen and H.-T. Lin, "Feature-aware label space dimension reduction for multi-label classification," in *NIPS*, 2012, pp. 1529–1537.

[28] T.-Y. Zhou, D.-C. Tao, and X.-D. Wu, "Compressed labeling on distilled labelsets for multi-label learning," *Machine Learning*, vol. 88, no. 1-2, pp. 69–126, 2012.

[29] F. Tai and H.-T. Lin, "Multilabel classification with principal label space transformation," *Neural Computing*, vol. 24, no. 9, pp. 2508–2542, 2012.

[30] Z.-J. Lin, G.-G. Ding, M.-Q. Hu, and J.-M. Wang, "Multi-label classification via feature-aware implicit label space encoding," in *ICML*, 2014, pp. 325–333.

[31] H. Wang, C. H. Q. Ding, and H. Huang, "Multi-label classification: Inconsistency and class balanced k-nearest neighbor," in *AAAI*, 2010.

[32] M.-L. Zhang and L. Wu, "Lift: Multi-label learning with label-specific features," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 1, pp. 107–120, 2015.

[33] K. Dembczyński, W. Cheng, and E. Hüllermeier, "Bayes optimal multilabel classification via probabilistic classifier chains," in *ICML*, 2010, pp. 1609–1614.

[34] K. Abhishek, V. Shankar, M. A. Krishna, and E. Charles, "Beam search algorithms for multilabel learning," *Maching Learning*, vol. 92, no. 1, pp. 65–89, 2013.

[35] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *KDD*, 2010, pp. 999–1008.

[36] S.-J. Huang and Z.-H. Zhou, "Multi-label learning by exploiting label correlations locally," in *AAAI*, 2012.

[37] J. Huang, G.-R. Li, S.-H. Wang, and Q.-M. Huang, "Group sensitive classifier chains for multi-label classification," in *ICME*, 2015.

[38] S.-W. Ji, L. Tang, S.-P. Yu, and J.-P. Ye, "Extracting shared subspace for multi-label classification," in *KDD*, 2008, pp. 381–389.

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

[40] X.-N. Kong and P. S. Yu, "Multi-label feature selection for graph classification," in *ICDM*, 2010, pp. 274–283.

[41] Y. Zhang and Z.-H. Zhou, "Multilabel dimensionality reduction via dependence maximization," *Knowledge Discovery from Data, ACM Transactions on*, vol. 4, no. 3, pp. 14:1–14:21, 2010.

[42] Z.-C. Lin, A. Ganesh, J. Wright, L.-Q. Wu, M.-M. Chen, and Y. Ma, "Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix," in *UIUC Technical Report UILU-ENG-09-2214*, 2009.

[43] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker, "Multilabel classification via calibrated label ranking," *Machine Learning*, vol. 73, no. 2, pp. 133–153, 2008.

[44] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hllermeier, "Optimizing the f-measure in multi-label classification: Plug-in rule approach versus structured loss minimization," in *ICML*, 2013, pp. 1130–1138.

[45] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, 2011.