Generalized *k*-Labelsets Ensemble for Multi-Label and Cost-Sensitive Classification

Hung-Yi Lo, Shou-De Lin, and Hsin-Min Wang *Senior Member, IEEE*

Abstract—Label powerset (LP) method is one category of multi-label learning algorithm. This paper presents a basis expansions model for multi-label classification, where a basis function is a LP classifier trained on a random k-labelset. The expansion coefficients are learned to minimize the global error between the prediction and the ground truth. We derive an analytic solution to learn the coefficients efficiently. We further extend this model to handle the cost-sensitive multi-label classification problem, and apply it in social tagging to handle the issue of the noisy training set by treating the tag counts as the misclassification costs. We have conducted experiments on several benchmark datasets and compared our method with other state-of-the-art multi-label learning methods. Experimental results on both multi-label classification and cost-sensitive social tagging demonstrate that our method has better performance than other methods.

Index Terms—Multi-label classification, cost-sensitive learning, ensemble method, labelset, hypergraph, social tag, tag count.

1 Introduction

Multi-label classification has attracted a great deal of attention in recent years. Different from single label classification, in multi-label classification, an instance could be associated with a set of labels jointly, instead of only one single label. For example, in image classification, an image may possess several concepts, such as "sea" and "sunset".

Label powerset (LP) [1] method is a kind of multi-label learning algorithm, which reduces the multi-label classification problem to a *single-label multi-class* classification problem by treating each distinct combination of labels in the training set as a different class. Given a test instance, the multi-class LP classifier predicts the most probable class, which can be transformed to a set of labels. Table 1 shows an example of multi-label dataset with transformed multi-class label based on the concept of LP. In contrast to the binary relevance approach, which loses the label dependency information while learning a binary classifier for each label independently, the LP method exploits conditional label dependency information by learning the joint label distribution [2]. However, one major concern for this model is that, when the number of labels increases, the number of potential classes increases proportionally, and each class will be associated with very few training instances. Moreover, LP can only predict labelsets observed in the training data. In [3], a method called Random *k*-Labelsets (RA*k*EL) is proposed to overcome the drawback of the traditional LP method. RA*k*EL randomly selects a number of label *subsets* from the original set of labels and uses the LP method to train the corresponding multi-class classifiers. The final prediction of RA*k*EL is made by voting of the LP classifiers in the ensemble. This method can not only reduce the number of classes, but also allow each class to have more training instances. Experimental results have shown an improvement of RA*k*EL over LP.

TABLE 1
An Example Multi-Label Dataset with Transformed Multi-Class Labels

Instance	Label Set	Transformed Class
1	Rock,Guitar	1
2	Rock, Guitar, Drum	2
3	Rock, Guitar, Vocal	3
4	Country, Guitar	4
5	Rock, Guitar, Drum	2
6	R&B, Vocal	5
7	Country, Guitar	4
8	Vocal	6

As RA*k*EL is considered an ensemble-based multi-label classification method, it has to follow the theory states [4] that "a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the individual classifiers are accurate and diverse". Here the term *accurate* implies that such classifier has a lower error rate than random guessing. In RA*k*EL, the diversity of classifiers is achieved by randomly selecting label subsets. The major limitation of this kind of ensemble method is that such "committee-of-diverse-experts" heuristics do not directly optimize for the learning objective. In contrast, AdaBoost.M2 [5] is an ensemble method for single-label classification, which directly optimizes a learning objective. Breiman

- *H.-Y. Lo is with the Department of Information Technology and Communication, Shih Chien University, Kaohsiung 845, Taiwan. E-mail: hungyi@mail.kh.usc.edu.tw.*
- *S.-D. Lin is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan. E-mail: sdlin@csie.ntu.edu.tw.*
- *H.-M. Wang is with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan. E-mail: whm@iis.sinica.edu.tw.*

Digital Object Indentifier 10.1109/TKDE.2013.112

[6] shows that the training procedure of AdaBoost is a form of gradient optimization to minimize the objective function $J(F) = \sum_i \exp(-y_i F(\boldsymbol{x}_i))$, where y_i is the class label of the instance $\boldsymbol{x}_i$ and $F(\cdot)$ is a hypothesis classifier of AdaBoost. Both theoretical and experimental results show the superiority of AdaBoost. Our work tries to follow a similar idea to improve RA*k*EL by re-designing the learning function to optimize a global error objective function.

Another limitation of RA*k*EL is that it assumes every base classifier in the ensemble is equally important. However, such assumption is problematic because the individual LP classifiers are trained on different randomly selected *k*-labelsets, some of them may have worse predictive performance than others or can be even redundant. Researchers have shown that properly determining the weights of the base classifiers in an ensemble can improve the prediction performance [5], [7]. For example, in AdaBoost, the coefficient α_t of a base classifier $f_t(\boldsymbol{x})$ is determined analytically and is proportional to the predictive performance of $f_t(\boldsymbol{x})$. We believe that learning the weights of base classifiers in an ensemble can improve the prediction performance for multi-label classification.

1.1 Generalized *k*-Labelsets Ensemble (GLE)

Inspired by the success of the LP-based methods, this paper presents a novel machine learning model for multi-label classification using the idea of *basis expansions* model [8, Chapter 5] of the following form:

$$H(\boldsymbol{x}) = \sum_{m=1}^{M} \beta_m h_m(\boldsymbol{x}), \tag{1}$$

where the basis functions $h_m(\boldsymbol{x})$ exploit the LP classifiers trained on random *k*-labelsets and β_m are their coefficients. In general, the basis expansions model treats the classifiers $h_m(\boldsymbol{x})$ as *dictionary functions* and uses the linear combination of these functions to approximate the target classifier during the learning procedure [8, Chapter 5]. Intuitively, given sufficient dictionary functions, the basis expansions model is a flexible representation for the target multi-label classifier. In our proposed GLE, the coefficients β_m are learned to minimize the *global error* between the prediction of $H(\boldsymbol{x})$ and the ground truth. The coefficients β_m of the corresponding base classifiers $h_m(\boldsymbol{x})$ become more significant for base classifiers with better performance, and are not redundant given others. Another interpretation of our method is from the Bayesian framework. The GLE can be considered as a kind of Bayesian model combination, that is, $P(\mathcal{Y}_i|\boldsymbol{x}) = \sum_m P(\mathcal{Y}_i|\boldsymbol{x}, h_m)P(h_m)$, where the $\mathcal{Y}_i$ is a predicted label set. In this aspect, the coefficients β_m are approximating the prior probabilities $P(h_m)$ for the model h_m.

When exploiting the basis expansions model, controlling the model complexity to avoid the overfitting problem is very important [8, Chapter 5]. We propose a regularized objective function based on the two-norm of $\boldsymbol{\beta}$ and a hypergraph Laplacian regularizer. The hypergraph can capture the high-order relation among instances and multiple labels jointly. We derive an analytic solution to learn the coefficients efficiently. The RA*k*EL [3] can be considered as a special case of our GLE when the coefficients are assigned *uniformly*.

We have also extended the GLE approach to handle cost-sensitive multi-label classification (CSML) cases. In the CSML problem, the misclassification cost can be different for each instance-label pair and is given before the training process. For example, in the image classification problem, the misclassification cost of the label "sunset" for an image whose subject is sunset should be higher than the misclassification cost of an additional label "gull". The CSML learning problem was first proposed in our previous paper [9], but has not been well studied yet. The previous solution uses the cost information independently for each label or each label set and might lose the label correlation information. In this paper, we extend the GLE to optimize a cost-weighted objective function for the misclassification costs of all label-instance pairs jointly.

1.2 Applications to Cost-Sensitive Social Tag Prediction Considering Tag Count

Tags are free text labels annotated on data in different format, such as images, music tracks, and websites. In some cases, the tags are objective and assigned by experts, such as part-of-speech tagging, or semantic role labelling. In other cases, the tags are more or less subjective while different persons might compose different tag sets for an object. In the Web 2.0 era, there are more and more online services that allow or even encourage users to tag objects such as images, or music tracks, and consequently create a large set of subjective tagging data. The emergence of human computing [10] also creates mass of opportunity for users to tag media. In general, we call such tagging behavior "collective tagging" and the subjective tags produced through such process "collective tags". The collective tags usually capture different aspects of the content. For example, the types of music tags may contain artist, genre, mood, and instrumentation. Given a feature representation for the content of the resources, some previous researches [11], [12] assumed that the tags are independent and, thus, transformed the social tag prediction problem into many independent binary classification problems, each for an individual tag. This strategy inevitably misses the *co-occurrence information* of multiple tags. It is generally believed that multi-label classification is more suitable for the task [9], [13]–[15].

Unlike the classification labels annotated by domain experts, the information provided in social tags may contain *noise* or *errors*. Consider that the *tag count* indicates the number of users who have annotated the given resource with the tag. We believe that not only the tags of an object but also the tag count information

should be considered in automatic social tagging because the count reflects the *confidence degree* of the tag. Important, confident, and relevant tags (with respect to the resources, such as music tracks or websites) are usually assigned by many different users. Previous research [16] also stated that, if many users annotate the same tag to some visually similar images, these tags are likely to reflect the semantic concept of the image content. Our idea is to use the tag count information to train a *cost-sensitive classifier* that minimizes the training error associated with tag counts. In other words, our goal is to correctly predict the important, confident, and relevant tags. We will use this task as a case study to demonstrate the usefulness of our cost-sensitive multi-label model.

1.3 Contribution

The contribution of this paper is fourfold.

1) We propose a novel label-powerset-based multi-label classification method, called Generalized *k*-Labelsets Ensemble. The GLE exploits random *k*-labelset LP classifiers as its base learners. The coefficients of these base classifiers are learned to minimize the global error and regularized by their two-norm and a hypergraph Laplacian regularizer. We derive an analytic solution to learn the coefficients efficiently.
2) We extend the GLE for cost-sensitive multi-label classification. To the best of our knowledge, this is the first method which minimizes the misclassification costs for all labels jointly.
3) We analyze the effect of tag count in using multi-label classification for social tag prediction. The observation inspires us to formulate the social tag prediction task as a cost-sensitive multi-label classification problem by treating the tag counts as the misclassification costs.
4) We have conducted experiments using a broad range of multi-label benchmark datasets and compared our method with other state-of-the-art multi-label learning methods. The results in terms of five popular evaluation metrics show that our method has better or competitive performance against other methods. We have also conduct experiments on cost-sensitive social tag annotation and retrieval. The results in terms of two cost-sensitive metrics show that GLE has better performance against other methods.

The remainder of this paper is organized as follows. In Section 2, we review the related work. Then, we present the Generalized *k*-Labelsets Ensemble for both multi-label classification and cost-sensitive multi-label classification in Section 3. Then, we discuss the results of multi-label classification in Section 4. In Section 5, we analyze the effect of tag count in social tag prediction and discuss the results of cost-sensitive social tagging. Finally, Section 6 contains some concluding remarks.

2 Related Work

Multi-label classification methods can be grouped into two categories: *algorithm adaptation* and *problem transformation* [1]. The *algorithm adaptation* methods extend some specific learning algorithms for single-label classification to solve the multi-label classification problem. Zhang and Zhou [17] extended the famous back-propagation algorithm for multi-label learning (BPMLL). Some algorithms are extended from the instance-based learning, such as multi-label K-nearest neighbor (MLKNN) [18] and instance-based learning by logistic regression (IBLR) [19]. Wang et al. [20] extended the discrete hidden Markov random field for multi-label classification. Liu et al. [21] proposed a Non-negative Matrix Factorization based approach which exploits both label correlation and unlabeled data.

The *problem transformation* methods transform the multi-label classification problem to one or many single-label classification tasks. Binary relevance and label powerset are two popular problem transformation approaches. The binary relevance method trains a binary classifier for each label independently. Sun et al. [22] proposed a two stage learning method to improve the binary relevance method based on hypergraph spectral learning. They exploited hypergraph spectral learning for feature transformation in the first stage and trained binary relevance classifiers in the second stage. The label powerset method treats each distinct combination of labels as a different class. Our proposed method belongs to the LP-based method. Rokach and Itach design methods to select the *k*-Labelsets in RA*k*EL using a greedy algorithm [23] which solves a set coverage problem. Their method assumes that every base classifier in the ensemble is equally important. Zhang and Zhang [24] proposed another two stage method, called LEAD. The LEAD algorithm exploits Bayesian network to encode the conditional dependencies of the labels in the first stage, and trains binary relevance classifiers with incorporating their parental labels as additional features in the second stage. Zhang [25] proposed to use different feature sets for each binary relevance classifier. We note that these algorithms can hardly be applied to *cost-sensitive* multi-label classification without significant modification.

General ensemble learning methods consist of two steps: learning the base classifiers and assigning weights for the base classifiers. Some methods, such as Random Forest [26] and RA*k*EL [3], only focus on the first step and assign equal weights for every base classifier. Ensemble selection [7] is another ensemble learning method which focuses on the latter step and its success has been proved by winning ACM KDD Cup 2009[1]. It performs an iterative search algorithm to determine the weights of classifiers for optimizing any objective function. AdaBoost is an ensemble method which learns both the base classifiers and their weights. In [27], Schapire and

1. http://www.kddcup-orange.com

Singer have shown an upper bound of training error for the AdaBoost classifier. In each iteration of its training procedure, the upper bound of training error is used as the criterion for choosing the base classifier and its associated weight. Our GLE is inspired by AdaBoost. GLE learns both the base classifiers and their weights. The learning method for the weights directly optimizes the global error.

Social tag prediction is one major application of multi-label classification. Previous researches have empirically studied different multi-label learning methods for tag annotation on music [14] and social bookmark [13]. However, these papers do not use the tag count information. Our previous study [9] is specific to the Music Information Retrieval Evaluation eXchange (MIREX)[2] audio tag prediction task and the method proposed in it does not use the cost information for all labels jointly. Part of this work appears in a conference paper [28]. This paper extends the technique for cost-sensitive multi-label classification.

3 Generalized *k*-Labelsets Ensemble

In this section, we start from introducing the concept of multi-label classification and cost-sensitive multi-label classification. Then, we review the concept of exploiting the hypergraph representation for multi-label classification, and describe the proposed Generalized *k*-Labelsets Ensemble for multi-label classification. Finally, we extend GLE for cost-sensitive multi-label classification.

3.1 Multi-Label Classification and Cost-Sensitive Multi-Label Classification

Let $\boldsymbol{x} \in \mathbb{R}^d$, which is a d-dimensional input space, and $\mathcal{Y} \subseteq \mathcal{L} = \{\lambda_1, \lambda_2, ..., \lambda_K\}$, which is a finite set of K possible labels. To facilitate the discussion, hereafter, $\mathcal{Y}$ is represented by a vector $\boldsymbol{y} = (y_1, y_2, ..., y_K) \in \{1, -1\}^K$, in which $y_j = 1 \Leftrightarrow \lambda_j \in \mathcal{Y}, y_j = -1 \Leftrightarrow \lambda_j \notin \mathcal{Y}$. We denote the labels of the whole instances by $\boldsymbol{Y} \in \mathbb{R}^{N \times K}$, where the i-th row of $\boldsymbol{Y}$ is $\boldsymbol{y}_i$; and denote the whole instances by $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, where the i-th row of $\boldsymbol{X}$ is $\boldsymbol{x}_i$. Given a training set $(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^N$ that contains N samples, the goal of multi-label classification is to learn a classifier $H : \mathbb{R}^d \rightarrow 2^K$ such that $H(\boldsymbol{x})$ predicts which labels should be assigned to an unseen sample $\boldsymbol{x}$.

In cost-sensitive multi-label learning, for each instance $\boldsymbol{x}_i$, a misclassification cost c_j is coupled to each label λ_j belonging to the label set of that instance. We give a more generalized definition as follow. We are given a training set $(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{c}_i)_{i=1}^N$ that contains N samples, where the j-th component c_{ij} denotes the cost to be paid when the label y_{ij} is misclassified. More specifically, c_{ij} is a *false negative cost* when $y_{ij} = 1$, and a *false positive cost* when $y_{ij} = -1$. We denote the misclassification costs of the whole instances by $\boldsymbol{C} \in \mathbb{R}^{N \times K}$, where the i-th row of $\boldsymbol{C}$ is $\boldsymbol{c}_i$. The goal of multi-label classification is to learn a classifier $H : \mathbb{R}^d \rightarrow 2^K$ such that $H(\boldsymbol{x})$ minimizes the expected misclassification cost on an unseen sample $\boldsymbol{x}$. In the next subsection, we describe the hypergraph representation for multi-label classification.

2. http://www.music-ir.org/mirex/

3.2 Multi-Label Learning with Hypergraphs

Hypergraph is a generalization of the traditional graph in which an edge can connect arbitrary non-empty subsets of the vertex set [29]. We denote a hypergraph by $G = (V, E)$, where V is the vertex set and E is the edge set, where each edge is a subset of V. Traditional graph is a special case of the hypergraph, in which every edge connects exactly two vertices, and is also called "2-graph". Given a multi-label dataset, the instances with their labels can be represented as one single hypergraph. More specifically, the vertex is a data point and the hyperedge is a label that connects the instances associated with it. A normalized hypergraph Laplacian $\boldsymbol{L} \in \mathbb{R}^{|V| \times |V|}$ is a commonly used technique for capturing the relationship among nodes in the hypergraph and has been used in spectral clustering [30]. Sun et al. proposed the hypergraph spectral multi-label learning algorithm [22] to learn a low-dimensional feature transformation $\boldsymbol{W}$, such that the data points sharing many common labels tend to be close to each other in the transformed space. The optimization problem for learning $\boldsymbol{W}$ is formulated as follows:

$$\begin{array}{ll} \min\limits_{\boldsymbol{W}} & \text{trace}(\boldsymbol{W}^T \boldsymbol{X} \boldsymbol{L} \boldsymbol{X}^T \boldsymbol{W}) \\ \text{s.t.} & \boldsymbol{W}^T \boldsymbol{X} \boldsymbol{X}^T \boldsymbol{W} = I, \end{array} \tag{2}$$

where $\boldsymbol{W}^T \boldsymbol{X}$ is the transformed data.

In this paper, we exploit the relation information encoded in the hypergraph Laplacian $\boldsymbol{L}$ in a different manner. We add a hypergraph Laplacian regularizer into the objective function for learning the coefficients in the ensemble. There are several different methods for learning the normalized hypergraph Laplacian matrix, but Agarwal et al. have shown that these methods lead to similar results [29]. In this paper, we use the clique expansion algorithm [22], [29] for calculating the normalized hypergraph Laplacian.

The clique expansion algorithm constructs a 2-graph from the original hypergraph by replacing each hyperedge with a clique, that is, maintaining an edge for each pair of vertices in the hyperedge. For a hypergraph, the vertex-edge incidence matrix $J \in \mathbb{R}^{|V| \times |E|}$ is defined as: $J(v, e) = 1$ if $v \in e$ and 0 otherwise. We denote the weight associated with the hyperedge e by $w(e)$. We denote the diagonal matrix whose diagonal entries are $w(e)$ by W_H. In the application of hypergraph for multi-label classification, the weights are set to uniform for each hyperedge. We denote the vertex degree in the expanded 2-graph by $d_c(u)$, where u is a vertex of the 2-graph, and denote the diagonal matrix whose diagonal entries are $d_c(u)$ by D_c. The normalized hypergraph Laplacian can be calculated by

$$\boldsymbol{L} = I - D_c^{-1/2} J W_H J^T D_c^{-1/2}. \tag{3}$$

3.3 Generalized *k*-Labelsets Ensemble for Multi-Label Classification

As introduced in [3], a k-labelset is a labelset $\mathcal{R} \subseteq \mathcal{L}$ with $|\mathcal{R}| = k$. GLE first trains M LP-based classifiers using randomly selected k-labelsets from the original set of labels. Then, GLE uses the base classifiers as dictionary functions and learns a linear combination of these functions. Algorithms 1 and 2 describe the training and classification processes, respectively. In Algorithm 2, the prediction of a multi-class LP classifier, g_m, for a sample $\boldsymbol{x}$ is denoted by $g_m(\boldsymbol{x}) \in \{1, 2, \ldots, Z\}$, where Z is the number of classes, that is, the number of different labelsets used for training the LP classifier. Note that Z is upper bounded by $min(N, 2^k)$, and is usually much smaller in practice [3]. The i, j-th element in Q_m is calculated by $q(g_m(\boldsymbol{x}_i), j)$, which is defined as:

$$q(g_m(\boldsymbol{x}_i), j) = \begin{cases} 1, & \text{if } j \in \mathcal{R}_m \\ & \text{and } j \text{ is positive in } g_m(\boldsymbol{x}_i), \\ -1, & \text{if } j \in \mathcal{R}_m \\ & \text{and } j \text{ is negative in } g_m(\boldsymbol{x}_i), \\ 0, & \text{if } j \notin \mathcal{R}_m. \end{cases} \tag{4}$$

For example, when $k = 2$, the classes 1, 2, 3, and 4 correspond to $(1,1), (1,-1), (-1,1)$, and $(-1,-1)$, respectively. If label j is not included in $\mathcal{R}_m$, $q(g_m(\boldsymbol{x}_i), j)$ is 0. If label j corresponds to the first label of $\mathcal{R}_m$, $q(1,j), q(2,j), q(3,j)$, and $q(4,j)$ will be $1, 1, -1$, and -1, respectively. We note that the function $q(g_m(\boldsymbol{x}), j)$ is used to generate the $h_m(\boldsymbol{x})$ in the final classifier (1) by gathering the predictions on all labels j.

The weight coefficients $\boldsymbol{\beta}$ for the base classifiers are learned by solving a minimization problem formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \tfrac{1}{2}||\boldsymbol{Y} - \sum_{m=1}^{M} \beta_m Q_m||_F^2 + \tfrac{\gamma}{2}||\boldsymbol{\beta}||_2^2 \\ & + \tfrac{\nu}{2}\text{trace}\left(\Big(\sum_{m=1}^{M} \beta_m Q_m \Big)^T \boldsymbol{L} \Big(\sum_{m=1}^{M} \beta_m Q_m \Big) \right), \end{aligned} \tag{5}$$

where $||\cdot||_F$ is the Frobenius norm of a matrix, $\boldsymbol{L}$ is the normalized hypergraph Laplacian, and $Q_m \in \mathbb{R}^{N \times K}$ is a transformed prediction of g_m which will be described in more detail later. The first term in the objective function aims to minimize the global error between the prediction of $H(\boldsymbol{x})$ and the multi-label ground truth $\boldsymbol{Y}$. The second term is a two-norm regularization term of the coefficients $\boldsymbol{\beta}$. The third term is a hypergraph regularization term.

The trade-off between fitting the training data and regularization can be controlled by the parameters γ and ν. A larger parameter γ will lead to smoother coefficients $\boldsymbol{\beta}$. The hypergraph Laplacian $\boldsymbol{L} \in \mathbb{R}^{N \times N}$ captures the high order labelling relationship among different instances. Following the spectral graph theory [22], [31], our idea behind the hypergraph regularization term is that the prediction on two instances, that is, two rows in $\sum_{m=1}^{M} \beta_m Q_m$, should be similar if they have high similarity according to the hypergraph.

Algorithm 1 The training process of GLE

- **Input:** number of models M, size of labelset k, learning parameters γ and ν, set of labels $\mathcal{L}$, and the training set $\mathcal{D} = (\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^N$
- **Output:** an ensemble of LP classifiers g_m, the corresponding k-labelsets $\mathcal{R}_m$ and coefficients β_m

1) Initialize $\mathcal{S} \leftarrow \mathcal{L}^k$
2) **for** $m \leftarrow 1$ **to** min(M,$|\mathcal{L}^k|$) **do**
   - $\mathcal{R}_m \leftarrow$ a k-labelset randomly selected from $\mathcal{S}$
   - train the LP classifier g_m based on $\mathcal{D}$ and $\mathcal{R}_m$
   - calculate a transformed prediction of g_m using (4)
   - $\mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{R}_m$
3) **end**
4) Learn $\boldsymbol{\beta}$ using (10)

Algorithm 2 The classification process of GLE

- **Input:** number of models M, a test sample $\boldsymbol{x}$, an ensemble of LP classifiers g_m, and the corresponding k-labelsets $\mathcal{R}_m$ and coefficients β_m
- **Output:** the multi-label classification vector $\boldsymbol{r} = (r_1, r_2, ..., r_K)$

1) **for** $j \leftarrow 1$ **to** K **do**
   a) $r_j = 0$
   b) **for each** g_m, **if** $j \in \mathcal{R}_m$ **do**
      - $r_j = r_j + \beta_m \cdot q(g_m(\boldsymbol{x}), j)$
   c) **end**
2) **end**

To solve the optimization problem (5), we start from rewriting the first term of the objective function by vectorizing Q_m and $\boldsymbol{Y}$. We denote the prediction of the base classifiers by $\hat{Q} \in \mathbb{R}^{(L \cdot N) \times M}$ whose columns are vectorized from Q_m by reshaping Q_m into $\mathbb{R}^{(L \cdot N)}$; and vectorize $\boldsymbol{Y}$ into $\hat{\boldsymbol{Y}} \in \mathbb{R}^{(L \cdot N)}$. Then, the first term of the objective function can be rewritten as $\frac{1}{2}||\hat{\boldsymbol{Y}} - \hat{Q}\boldsymbol{\beta}||_2^2$. To further simplify the third term of the objective function, we let $Q_{m,j}$ be the j-th column vector in Q_m. We perform operations on the third term as follows:

$$\begin{aligned} & \tfrac{\nu}{2}\text{trace}\left(\Big(\sum_{m=1}^{M} \beta_m Q_m \Big)^T \boldsymbol{L} \Big(\sum_{m=1}^{M} \beta_m Q_m \Big) \right) \\ = \; & \tfrac{\nu}{2}\text{trace} \begin{pmatrix} \boldsymbol{\beta}^T \boldsymbol{P}_{1,1} \boldsymbol{\beta} & \cdots & \boldsymbol{\beta}^T \boldsymbol{P}_{1,L} \boldsymbol{\beta} \\ \vdots & \ddots & \vdots \\ \boldsymbol{\beta}^T \boldsymbol{P}_{L,1} \boldsymbol{\beta} & \cdots & \boldsymbol{\beta}^T \boldsymbol{P}_{L,L} \boldsymbol{\beta} \end{pmatrix} \\ = \; & \tfrac{\nu}{2} \sum_{j=1}^{L} \boldsymbol{\beta}^T \boldsymbol{P}_{j,j} \boldsymbol{\beta} \end{aligned} \tag{6}$$

where $\boldsymbol{P}_{i,j} \in \mathbb{R}^{M \times M}$ is generated as

$$\boldsymbol{P}_{i,j} = \begin{pmatrix} Q_{1,i}^T \boldsymbol{L} Q_{1,j} & \cdots & Q_{1,i}^T \boldsymbol{L} Q_{M,j} \\ \vdots & \ddots & \vdots \\ Q_{M,i}^T \boldsymbol{L} Q_{1,j} & \cdots & Q_{M,i}^T \boldsymbol{L} Q_{M,j} \end{pmatrix}. \tag{7}$$

Let $\sum_{j=1}^{L} \boldsymbol{P}_{j,j}$ in (6) be denoted as $\boldsymbol{\rho}$, the loss function in (5) can be rewritten as

$$L(\boldsymbol{\beta}) = \frac{1}{2}(\hat{\boldsymbol{Y}} - \hat{Q}\boldsymbol{\beta})^T(\hat{\boldsymbol{Y}} - \hat{Q}\boldsymbol{\beta}) + \frac{\gamma}{2}\boldsymbol{\beta}^T\boldsymbol{\beta} + \frac{\nu}{2}\boldsymbol{\beta}^T\boldsymbol{\rho}\boldsymbol{\beta}. \quad (8)$$

We take derivatives of equation (8) with respect to $\boldsymbol{\beta}$ and set them equal to zero,

$$\frac{\partial L}{\partial \boldsymbol{\beta}} = -\hat{Q}^T\hat{\boldsymbol{Y}} + \hat{Q}^T\hat{Q}\boldsymbol{\beta} + \gamma \cdot I\boldsymbol{\beta} + \frac{\nu}{2}(\boldsymbol{\rho} + \boldsymbol{\rho}^T)\boldsymbol{\beta} = 0. \quad (9)$$

where I is the identity matrix. Hence, the optimization problem (5) has a unique solution $\boldsymbol{\beta}^*$:

$$\boldsymbol{\beta}^* = (\hat{Q}^T\hat{Q} + \gamma \cdot I + \frac{\nu}{2}(\boldsymbol{\rho} + \boldsymbol{\rho}^T))^{-1}\hat{Q}\hat{\boldsymbol{Y}}. \quad (10)$$

The computational cost of training GLE depends on the training speed of the LP base classifier. We note that the matrix to be inverted in (10) is a M-dimensional square matrix and typically the number of models M is set from 15 to 250. Hence, solving such equation is computationally feasible and learning $\boldsymbol{\beta}$ does not increase much overhead.

3.4 Generalized *k*-Labelsets Ensemble for Cost-Sensitive Multi-Label Learning

GLE can also be extended for cost-sensitive multi-label classification. We follow the same training and classification procedures as shown in Algorithms 1 and 2, and modify the objective function for learning the coefficients $\boldsymbol{\beta}$ in (5). Recall that the first term in the objective function of (5) is the global error between the multi-label classifier prediction, $\sum_{m=1}^{M} \beta_m Q_m$, and the multi-label ground truth $\boldsymbol{Y}$. We modify it to a *cost-weighted global error* by multiplying the global error with the multi-label misclassification cost matrix $\boldsymbol{C}$. The optimization problem of learning the coefficients $\boldsymbol{\beta}$ for cost-sensitive multi-label classification can then be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \tfrac{1}{2}||\boldsymbol{C} \circ \Big(\boldsymbol{Y} - \sum_{m=1}^{M} \beta_m Q_m\Big)||_F^2 + \tfrac{\gamma}{2}||\boldsymbol{\beta}||_2^2 \\ & + \tfrac{\nu}{2}\text{trace}\Bigg(\Big(\sum_{m=1}^{M} \beta_m Q_m\Big)^T \boldsymbol{L}\Big(\sum_{m=1}^{M} \beta_m Q_m\Big)\Bigg), \end{aligned} \quad (11)$$

where $\circ$ is the dot product of two matrices. Similarly, the optimization problem (11) has a unique solution $\boldsymbol{\beta}^*$,

$$\boldsymbol{\beta}^* = \Big(\boldsymbol{C} \circ (\hat{Q}^T\hat{Q}) + \gamma \cdot I + \frac{\nu}{2}(\boldsymbol{\rho} + \boldsymbol{\rho}^T)\Big)^{-1}\hat{Q}\hat{\boldsymbol{Y}}. \quad (12)$$

4 COST-INSENSITIVE EXPERIMENTS

We compare the proposed methods with other multi-label classification algorithms on ten datasets. Six of them come from the social tagging domain[3]. In the following subsections, we describe the datasets, the evaluation criteria for multi-label classification, and the experimental setup, and then discuss the experimental results.

3. The second part of this paper focuses on cost-sensitive multi-label classification. However, cost-sensitive multi-label datasets are not as easy to obtain. As we will show in Section 5, the tag counts in social tagging can be used as the misclassification costs, and the social tag prediction problem can be treated as a cost-sensitive multi-label classification problem.

TABLE 2
The 45 Tags Used in The MIREX Audio Tag Classification Evaluation

metal	instrumental	horns	piano	guitar
ambient	saxophone	house	loud	bass
fast	keyboard	electronic	noise	british
solo	electronica	beat	80s	dance
jazz	drum machine	strings	pop	r&b
female	rock	voice	rap	male
slow	vocal	quiet	techno	drum
funk	acoustic	distortion	organ	soft
country	hip hop	synth	trumpet	punk

4.1 Datasets

We conduct experiments on ten datasets belonging to different domains. The datasets include *scene*, *enron*, *cal500*, *majorminer*, *medical*, *bibtex*, and four versions of *delicious* (from *dlc1* to *dlc4*). Four of these datasets, including *scene*, *enron*, *medical*, and *bibtex*, can be downloaded from the MULAN library [32] website. These four datasets have been used in [3]. More details of the other six datasets are described below.

MajorMiner [33] is a web-based game[4] for collecting music tags. Players on MajorMiner listen to short music clips (each about 10 seconds long) and label them with relevant words and phrases. We consider 45 tags, as listed in Table 2, which have been used in the audio tag classification task in MIREX. We download all the audio clips that are associated with these 45 tags from the website of the MajorMiner's game. We extract audio features with respect to various musical information, including dynamics, rhythm, timbre, pitch, and tonality. More details can be found in [9].

The *cal500* [34], [35] dataset consists of 500 popular Western songs from 500 different artists. Each song has been annotated by at least three humans, using a semantic vocabulary of 149 tags that describe genres, instruments, vocal characteristics, emotions, acoustic characteristics, and song usages. For each annotated tag, a weight score (can be considered equivalent to the tag count) is available to indicate the level of agreement over all annotators. The dataset is publicly available[5]. To alleviate the label sparsity problem as mentioned by the provider of *cal500* [35], we use the 91 tags with at least 50 positively associated songs. The feature extraction on the *cal500* dataset is similar to that used on the *majorminer* dataset. There is another version of *cal500* on the MULAN website. We do not adopt this version since the soft decision label which will be used in Section 5, is not available on it.

4. http://www.majorminer.org/
5. http://cosmal.ucsd.edu/cal/projects/AnnRet/

The *delicious* dataset is generated from the famous social bookmarking website del.icio.us. The URLs with social tags were crawled in February 2008 [36]. In addition, the source web page of each URL was also collected and represented by the boolean bag-of-words model. The crawled data contain 56,804 URLs, and each of them is annotated by at least 20 users. To reduce the noise introduced by synonym terms, a semi-automatic normalization procedure has been performed to group different tag strings denoting the same concept to an identical tag [36]. The resulting data contains 33,490 different tags. According to a recent experimental study of multi-label classification methods [37], the training time of some algorithms on such a large dataset might last for more than one week. To reduce the computational cost of training multi-label classifiers, the *delicious* data is divided into 4 subset versions: from *dlc1* to *dlc4*. We first randomly select four disjoint tag subsets, whose numbers are 39, 58, 77, and 96, respectively, from the tags with at least 300 positively associated URLs. For each tag subset, we randomly select 8000 URLs which are positively associated to at least one tag in the corresponding tag subset. Since the web pages contain many typos and noisy information, we perform feature selection on the four subset versions separately. We calculate the signal-to-noise (s2n) ratio of each feature-label pair, and select the features whose s2n scores (with respect to any one of the labels) are higher than a threshold. The resulting numbers of features for the four subset versions are 325, 483, 611, and 735, respectively. There is another version of the *delicious* data on the MULAN website. We do not adopt that version since the tag counts are not available on it.

The statistics of the ten datasets are shown in Table 3. The "cardinality" is defined as the average number of labels per instance, while the "density" is the same number normalized by L. The "distinct" counts the number of distinct label sets, while the "normalized distinct" is the same number divided by the number of instances. We note that the size of the first five datasets are medium or small, that is, the number of instances is smaller than 3000; and the size of the last five datasets is large, that is, the number of instances is larger than 7000. We also note that the cardinality of the *cal500* dataset is 34.474, which is much larger than that of the other datasets. This fact makes predicting the whole label set of an instance in *cal500* correctly a very difficult task.

4.2 Evaluation Metrics For Multi-Label Classification

We use five popular evaluation metrics for multi-label classification: the Hamming loss, ranking loss, subset $0/1$ loss, one error, and average precision [1], [37]. Hamming loss calculates the percentage of labels whose relevance is predicted incorrectly. Ranking loss evaluates the average fraction of label pairs, that is, a positive label versus a negative label, that are not correctly ordered. Subset $0/1$ loss evaluates a multi-label prediction as a whole. It evaluates the percentage of predicted label sets that do not exactly match the true label sets. One error evaluates the number of times the top-ranked label is not relevant. For these four metrics, the smaller the result value is, the better the algorithm performs. Average precision evaluates that, for each relevant label, the percentage of relevant labels among all labels that are ranked above it.

4.3 Experimental Setup

For the cost-insensitive multi-label classification part, we compare the performance of GLE with that of six state-of-the-art multi-label learning algorithms: RA*k*EL, BR (Binary Relevance) [1], CC (Classifier Chains) [38], MLKNN [18], IBLR [19], and BPMLL [17]. We select these methods according to a recent extensive experimental study of multi-label classification [37]. We implement GLE using MATLAB. We exploit the linear multi-class SVM implemented in LIBSVM for the LP classifiers in GLE, which is based on the *one-against-one* approach. The parameter C in SVM is set to 1. Based on our observation, the best selected parameters k and M are usually the same for both RA*k*EL and GLE. We select k and M using cross-validation for RA*k*EL and use Hamming loss as the model selection criterion. We apply the selected k and M for GLE and the parameters are listed in Table 4. Then, the parameters γ and ν in GLE are selected using cross-validation with respect to the five different evaluation metrics, respectively. As mentioned in the Section 3.3, the computational cost of obtaining $\boldsymbol{\beta}^*$ is very small. After training the LP classifiers, we can obtain different $\boldsymbol{\beta}$ with thousands of different parameter pair, that is, γ and ν, within one second[6]. The cross-validation used for selecting γ and ν is not used for generating the final result.

The BR, CC, MLKNN, and IBLR are implemented in the MULAN package. For BR, we exploit the linear logistic regression with probability output score as the base classifier. For CC, we exploit the linear SVM as the base classifier and the parameter C is set to 1. For MLKNN and IBLR, the size of the neighborhood is set to 10 [18], [19]. We use the MATLAB implementation of BPMLL, which is provided by the authors of BPMLL. For BPMLL, the number of hidden neurons is set to 20% of the dimensionality, and the number of training epochs is selected using cross-validation. The implementation of RA*k*EL are similar to that of GLE. We perform three-fold cross-validation sixty times for the five medium-scale datasets and three times for the five large-scale datasets; and calculate the mean of the results.

4.4 Experimental Results

The experimental results of multi-label classification are summarized in Table 5. The numbers in parentheses

6. We have conducted experiments with different γ and ν. The results suggest that we can obtain good enough results by tuning γ and set ν to 1, and thus alleviate the burden of parameter selection.

TABLE 3
Statistics of The Multi-Label Datasets

Data set	Instances	Features	Feature Type	Labels	Cardinality	Density	Distinct	Normalized Distinct
scene	2407	294	numeric	6	1.074	0.179	15	0.006
enron	1702	1001	nominal	53	3.378	0.064	753	0.442
cal500	500	140	numeric	91	34.474	0.198	500	1.000
majorminer	2472	177	numeric	45	4.119	0.092	1553	0.628
medical	978	1449	nominal	45	1.245	0.028	94	0.096
bibtex	7395	1836	nominal	159	2.402	0.015	2856	0.386
dlc1	8000	325	nominal	39	1.663	0.043	1024	0.128
dlc2	8000	483	nominal	58	2.027	0.035	1995	0.249
dlc3	8000	611	nominal	77	2.529	0.033	3286	0.411
dlc4	8000	735	nominal	96	2.586	0.027	3704	0.463

TABLE 4
Selected Parameters k and M of GLE and RAkEL for The Multi-Label Datasets

Dataset	k	M
scene	4	15
enron	16	250
cal500	10	250
majorminer	14	250
medical	14	250
bibtex	24	250
dlc1	26	250
dlc2	32	250
dlc3	32	250
dlc4	32	250

represent the rank of the algorithm among the compared algorithms. The average rankings of our method GLE on ten datasets using five different metrics are 1.8, 2.4, 1.5, 1.4, and 1.9, respectively. On four of the five metrics, GLE achieves the best performance. GLE performs slightly worse than MLKNN only in terms of ranking loss; however, the difference is very small. We observe that RAkEL performs closely to GLE in terms of Hamming loss; but in terms of the other four metrics, GLE performs much better than RAkEL. Among the five metrics, the improvement of GLE is more significant on Hamming loss, subset $0/1$ error, and one error. Generally speaking, GLE has better or competitive performance against the other state-of-the-art methods. We have run the pairwise t-test at the 5% significance level on the experimental results. We use $\bullet/\circ$ to indicate whether GLE is statistically superior/inferior to the compared algorithm in Table 5. When the difference is not significant, no marker is given. There are 246 cases in which GLE performs significantly better than the compared method and only 34 cases in which GLE performs significantly worse.

Since both GLE and RAkEL are LP-based methods, we calculate the relative improvement of GLE over RAkEL for each dataset, respectively. Then, we show the average relative improvement over all datasets for the five evaluation metrics in Table 6, respectively. In each iteration of the GLE and RAkEL training phase, they use the same randomly selected k-Labelsets for the LP classifiers. We have also shown the relative improvement of two

TABLE 6
Relative Improvement of GLE and Its Two Simplified Versions Over RAkEL in Terms of Five Different Evaluation Metrics (In $\%$)

	Hamming Loss	Ranking Loss	Subset $0/1$ Loss	One Error	Average Precision
$\gamma = 0$	-9.11	26.44	-5.19	-28.29	2.16
$\nu = 0$	0.20	51.26	1.30	3.87	10.25
GLE	0.22	55.34	1.83	8.15	11.80

simplified versions of GLE, that is, without two-norm regularization ($\gamma = 0$) or without hypergraph regularization ($\nu = 0$), over RAkEL in Table 6. We observe that the relative improvement of GLE over RAkEL is more significant in terms of ranking loss than the other metrics. GLE achieves around 10% relative improvements over RAkEL in terms of both one error and average precision, but the improvement is small in terms of Hamming loss and subset $0/1$ loss. The simplified version of GLE without two-norm regularization performs even worse than RAkEL in terms of Hamming loss, subset $0/1$ loss and one error. The simplified version of GLE without hypergraph regularization performs better than RAkEL but worse than GLE.

We further compare GLE and RAkEL by varying parameters k and M. They use the same randomly selected k-Labelset for the LP classifiers. We show the average relative improvement of GLE over RAkEL in terms of five different evaluation metrics in Figure 1. The results for the selected parameter k in Table 4 are indicated by the blue curves in Figure 1. We have also tested larger k and smaller k, and the results are indicated by the red and green curves, respectively. We incrementally increase the number of models M as indicated by the horizontal axis. Since the number of instances and the number of labels for the *scene* dataset are much smaller than that of the other datasets, the ten steps of M are from 6 to 15. For the other nine datasets, the ten steps of M are from 25 to 250 with an increment of 25. From the results, we observe that GLE performs better than RAkEL in most parameter settings, especially when the number of models increases. The major reason could be that, when the number of models is small, it is less likely

TABLE 5
Experimental Results in Terms of Five Different Evaluation Metrics. The Numbers in Parentheses Represent the Rank of the Algorithm Among the Compared Algorithms. The Average Rank is the Average of the Ranks Across All Datasets. •/○ indicates whether GLE is statistically superior/inferior to the compared algorithm (the pairwise t-test at the 5% significance level).

	GLE	RA*k*EL	BR	CC	MLKNN	IBLR	BPMLL
Hamming Loss							
scene	0.0973 (3)	0.0968 (2) ○	0.1622 (6) •	0.1069 (4) •	0.0896 (1) ○	0.1472 (5) •	0.2454 (7) •
enron	0.0488 (1)	0.0488 (2) •	0.1105 (7) •	0.0601 (4) •	0.0531 (3) •	0.1083 (6) •	0.0693 (5) •
cal500	0.1641 (2)	0.1634 (1) ○	0.3473 (6) •	0.2962 (4) •	0.2775 (3) •	0.3632 (7) •	0.3077 (5) •
majorminer	0.0810 (1)	0.0812 (2) •	0.1118 (6) •	0.0815 (3) •	0.0851 (4) •	0.1191 (7) •	0.0926 (5) •
medical	0.0099 (1)	0.0100 (2) •	0.0206 (5) •	0.0103 (3) •	0.0163 (4) •	0.4917 (7) •	0.0277 (6) •
bibtex	0.0125 (2)	0.0124 (1)	0.0803 (7) •	0.0152 (4) •	0.0137 (3) •	0.0738 (6) •	0.0158 (5) •
dlc1	0.0386 (2)	0.0389 (3) •	0.0477 (5) •	0.0382 (1) ○	0.0397 (4) •	0.0536 (7) •	0.0508 (6) •
dlc2	0.0325 (2)	0.0326 (3)	0.0532 (6) •	0.0324 (1)	0.0334 (4) •	0.0638 (7) •	0.0354 (5) •
dlc3	0.0307 (2)	0.0308 (3) •	0.0711 (6) •	0.0303 (1) ○	0.0316 (4) •	0.0771 (7) •	0.0339 (5) •
dlc4	0.0246 (2)	0.0246 (3) •	0.0756 (6) •	0.0242 (1) ○	0.0256 (4) •	0.1577 (7) •	0.0282 (5) •
Average Rank	1.8	2.2	6.0	2.6	3.4	6.6	5.4
Ranking Loss							
scene	0.1031 (2)	0.1546 (3) •	0.1959 (5) •	0.3057 (6) •	0.0813 (1) ○	0.1688 (4) •	0.5008 (7) •
enron	0.0902 (1)	0.2472 (4) •	0.2894 (5) •	0.4880 (7) •	0.0954 (2) •	0.3058 (6) •	0.1175 (3) •
cal500	0.1527 (1)	0.2249 (2) •	0.3503 (5) •	0.5633 (7) •	0.2476 (3) •	0.3829 (6) •	0.2959 (4) •
majorminer	0.1249 (1)	0.1746 (4) •	0.1692 (3) •	0.6772 (7) •	0.1487 (2) •	0.1758 (5) •	0.3821 (6) •
medical	0.0550 (2)	0.1100 (4) •	0.0658 (3) •	0.2042 (6) •	0.0465 (1) ○	0.1186 (5) •	0.2897 (7) •
bibtex	0.1976 (2)	0.4199 (6) •	0.2309 (5) •	0.5989 (7) •	0.2184 (3) •	0.2264 (4) •	0.0774 (1) ○
dlc1	0.1777 (4)	0.6470 (6) •	0.1272 (1) ○	0.8371 (7) •	0.1567 (3) ○	0.1395 (2) ○	0.2020 (5) •
dlc2	0.1736 (4)	0.7602 (6) •	0.1411 (1) ○	0.8749 (7) •	0.1556 (2) ○	0.1608 (3) ○	0.2019 (5) •
dlc3	0.2194 (4)	0.7746 (6) •	0.1894 (1) ○	0.8522 (7) •	0.1961 (2) ○	0.2141 (3) ○	0.2496 (5) •
dlc4	0.2138 (3)	0.7548 (6) •	0.1993 (2) ○	0.8212 (7) •	0.1911 (1) ○	0.2597 (5) •	0.2447 (4) •
Average Rank	2.4	4.7	3.1	6.8	2.0	4.3	4.7
Subset 0/1 Loss							
scene	0.2743 (2)	0.2853 (3) •	0.4589 (6) •	0.3330 (4) •	0.2565 (1) ○	0.4122 (5) •	0.8339 (7) •
enron	0.7759 (1)	0.7816 (2) •	0.9051 (5) •	0.8447 (3) •	0.8888 (4) •	0.9152 (6) •	0.9963 (7) •
cal500	1.0000 (1)	1.0000 (1)	1.0000 (1)	1.0000 (1)	1.0000 (1)	1.0000 (1)	1.0000 (1)
majorminer	0.9081 (1)	0.9113 (2) •	0.9602 (5) •	0.9434 (4) •	0.9358 (3) •	0.9701 (6) •	0.9910 (7) •
medical	0.2206 (1)	0.2266 (2) •	0.3422 (5) •	0.2769 (3) •	0.3464 (6) •	0.2824 (4) •	0.9819 (7) •
bibtex	0.7271 (2)	0.7270 (1)	0.9289 (7) •	0.8092 (3) •	0.8780 (5) •	0.9216 (6) •	0.8344 (4) •
dlc1	0.7871 (4)	0.8056 (5) •	0.7443 (1) ○	0.8971 (7) •	0.7817 (2)	0.7863 (3)	0.8669 (6) •
dlc2	0.8500 (1)	0.8854 (4) •	0.8505 (2)	0.9279 (6) •	0.8512 (3)	0.9045 (5) •	0.9457 (7) •
dlc3	0.8822 (1)	0.9038 (3) •	0.9338 (5) •	0.9282 (4) •	0.9007 (2) •	0.9517 (7) •	0.9491 (6) •
dlc4	0.8776 (1)	0.8957 (2) •	0.9426 (6) •	0.9175 (5) •	0.8981 (4) •	0.8960 (3) •	0.9696 (7) •
Average Rank	1.5	2.5	4.3	4.0	3.1	4.6	5.9
One Error							
scene	0.2543 (2)	0.2565 (3) •	0.4312 (6) •	0.2850 (4) •	0.2346 (1) ○	0.3841 (5) •	0.8119 (7) •
enron	0.2873 (2)	0.2776 (1) ○	0.5343 (6) •	0.4100 (4) •	0.3195 (3) •	0.5364 (7) •	0.4944 (5) •
cal500	0.1100 (2)	0.2028 (4) •	0.3700 (7) •	0.2842 (5) •	0.0777 (1) ○	0.3496 (6) •	0.1309 (3) •
majorminer	0.3693 (1)	0.3736 (2) •	0.6830 (5) •	0.4721 (4) •	0.4525 (3) •	0.7234 (6) •	0.8349 (7) •
medical	0.1458 (1)	0.1485 (2) •	0.2680 (6) •	0.1896 (3) •	0.2638 (5) •	0.1989 (4) •	0.9756 (7) •
bibtex	0.3889 (1)	0.3892 (2)	0.8177 (7) •	0.5511 (4) •	0.6026 (5) •	0.7932 (6) •	0.5413 (3) •
dlc1	0.5725 (2)	0.5777 (4)	0.5410 (1) ○	0.7684 (7) •	0.5747 (3)	0.6111 (5) •	0.6919 (6) •
dlc2	0.5974 (1)	0.7096 (4) •	0.6450 (3) •	0.8099 (7) •	0.6158 (2) •	0.7641 (5) •	0.7688 (6) •
dlc3	0.6375 (1)	0.6945 (3) •	0.7605 (5) •	0.7515 (4) •	0.6440 (2)	0.8192 (7) •	0.7921 (6) •
dlc4	0.5755 (1)	0.6433 (4) •	0.7725 (6) •	0.7076 (5) •	0.6229 (3) •	0.6194 (2) •	0.8070 (7) •
Average Rank	1.4	2.9	5.2	4.7	2.8	5.3	5.7
Average Precision							
scene	0.8415 (3)	0.8420 (2) ○	0.7213 (6) •	0.7991 (4) •	0.8600 (1) ○	0.7541 (5) •	0.4211 (7) •
enron	0.6658 (1)	0.6456 (2) •	0.4155 (6) •	0.4661 (5) •	0.6200 (3) •	0.3986 (7) •	0.5053 (4) •
cal500	0.6131 (2)	0.6030 (3) •	0.4996 (6) •	0.5245 (5) •	0.6351 (1) ○	0.4934 (7) •	0.5774 (4) •
majorminer	0.5988 (1)	0.5932 (2) •	0.4531 (4) •	0.3526 (6) •	0.5412 (3) •	0.4271 (5) •	0.2206 (7) •
medical	0.8611 (1)	0.8548 (2) •	0.7887 (6) •	0.7956 (3) •	0.7938 (4) •	0.7922 (5) •	0.1222 (7) •
bibtex	0.5222 (1)	0.5153 (2) •	0.2234 (7) •	0.3776 (4) •	0.3376 (5) •	0.2437 (6) •	0.4451 (3) •
dlc1	0.4749 (4)	0.4160 (5) •	0.5379 (1) ○	0.3044 (7) •	0.4952 (3) ○	0.4978 (2) ○	0.3933 (6) •
dlc2	0.4210 (3)	0.2849 (6) •	0.4335 (1) ○	0.2038 (7) •	0.4311 (2) ○	0.3638 (4) •	0.3076 (5) •
dlc3	0.3568 (2)	0.2780 (4) •	0.3194 (3) •	0.2268 (7) •	0.3594 (1)	0.2754 (5) •	0.2525 (6) •
dlc4	0.3567 (1)	0.2975 (3) •	0.2885 (4) •	0.2508 (6) •	0.3544 (2)	0.2707 (5) •	0.2174 (7) •
Average Rank	1.9	3.1	4.4	5.4	2.5	5.1	5.6

to select a significantly better or worse k-Labelset, so it is reasonable to assume that all LP classifiers are equally important.

5 Cost-Sensitive Social Tag Prediction

In this section, we analyze the effect and importance of tag counts in multi-label classification for social tag prediction. The analysis inspired us to apply cost-sensitive multi-label learning for social tag prediction by treating the tag counts as costs. We describe two evaluation metrics for cost-sensitive multi-label classification.

5.1 Effect of Tag Count in Using Multi-Label Classification for Social Tag Prediction

Previous research generally used tag-count as a measure for the existence of a tag. For example, in the audio tag prediction task in MIREX, the tag count is transformed into 1 (with a tag) or 0 (without a tag) based on a certain pre-determined threshold. In [33], the authors tried different thresholds to transform the tag counts into binary decisions for training, but found it difficult to improve the classification accuracy. Part of the reason is that such coarse-grained discretization does not fully use the count information.

In order to demonstrate the importance of the tag count information, we analyze the difference between the prediction results of the high count tags and low count tags in terms of the false negative rate (FNR) on the six social tag prediction datasets: *cal500*, *majorminer*, and the four subset versions of the *delicious* data (from *dlc1* to *dlc4*). The results are shown in Table 7. The high count tags include tags whose counts are larger than a pre-defined threshold. The thresholds for the *cal500* and *majorminer* datasets are set to 5 and 6, respectively; and the threshold for the four *delicious* datasets is set to 50. The low count tags are tags whose counts are equal to the respective smallest tag count in that dataset. We use binary relevance SVM (BRSVM) and IBLR for training the multi-label classifiers. The results are obtained from three-fold cross-validation.

From Table 7, we observe that, for all of the six datasets, both BRSVM and IBLR have significantly higher false negative rates on the low count tags than on the high count tags. For example, for the *cal500* dataset, the FNR is 37.61 % for the high count tags and 70.94 % for the low count tags, by using IBLR. We have also conducted the same experiment by using other multi-label classifiers such as GLE and MLKNN, and the results are similar to that of BRSVM and IBLR. It is clear that the low count tags are more difficult to recognize than the high count tags. This observation inspires us to further study the relationship of the tag counts and the annotated resources.

We show some example URLs in the *delicious* dataset associated with the "art", "c", "education", and "web2.0" tags as the high count tags or low count tags in Tables 8 and 9, respectively. For each tag, we select three URLs which are repeatedly annotated with the tag, i.e., the tag is a high count tag for these URLs. The URLs are shown in Table 8. We also select three URLs which are annotated with the tag only a few times, i.e., the tag is a low count tag for the URLs. The URLs are shown in Table 9. The count of the tag and a short description for the URL are also shown in the table. For comparison, we also show two other high count tags annotated to the same URL in the last column in Table 8 and 9.

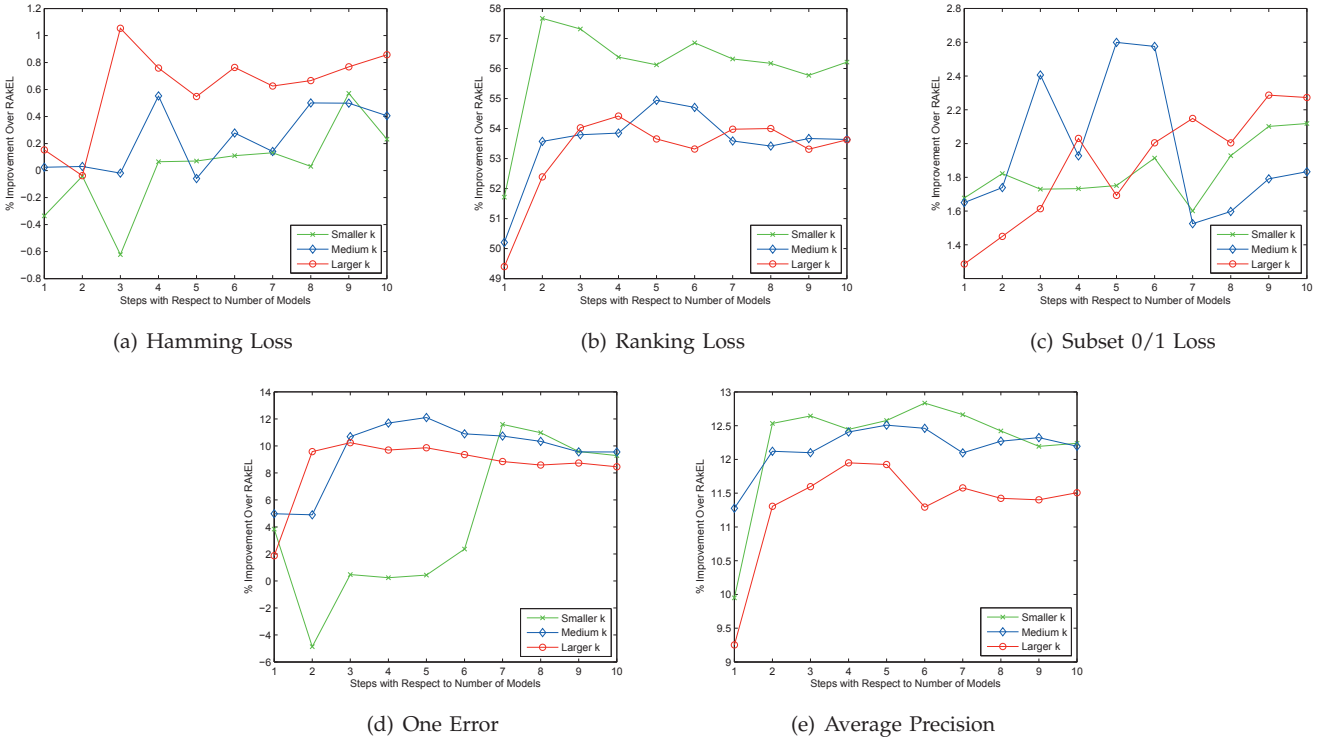
TABLE 7
Comparison of Prediction Results of High Count Tags and Low Count Tags in Terms of False Negative Rate (in %)

Dataset	Tag Count Type	Number of Tag-Instance Pairs	BRSVM	IBLR
cal500	High	308	38.92	37.61
	Low	3395	75.74	70.94
majorminer	High	686	31.49	38.96
	Low	4418	54.99	55.29
dlc1	High	1497	65.60	55.85
	Low	1951	89.95	82.32
dlc2	High	1658	69.54	59.83
	Low	2541	91.46	83.35
dlc3	High	2142	67.55	58.17
	Low	3178	92.54	83.42
dlc4	High	2362	64.01	62.53
	Low	3169	91.86	77.75

From Table 8, we observe that the high count tags usually capture the *salient property* of a URL. For example, the URL www.cplusplus.com, which is a famous site for C++ programming language reference and documentation, has been annotated with the tag "c" for 771 times; and the URL ocw.mit.edu/courses, which is a famous site called MIT OpenCourseWare, has been annotated with the tag "education" for 1024 times. We believe that the association between a URL and their high count tags is intuitive, obvious, and significant.

In contrast, the association between a URL and their low count tags is not salient and sometimes hard to understand. For example, the URL www.shoutcast.com, which aims to provide free internet radio stations, has been annotated with "art" for 2 times. The reason might be that some people consider music as a kind of art. Its high count tags "music" and "radio", as shown in the last column in Table 9, indeed capture more salient properties of this site than the "art" tag. The URL www.python.org/doc, which is the official documentation website of the Python programming language, has been annotated with "c" for 2 times. A plausible reason is that Python provides an application programming interface (API) for the C language. Its high count tags "python" and "reference" obviously capture more salient properties of this URL than the "c" tag.

The examples in Tables 8 and 9 demonstrate why the low count tags are more difficult to recognize than the



(a) Hamming Loss (b) Ranking Loss (c) Subset 0/1 Loss

(d) One Error (e) Average Precision

Fig. 1. Average Relative Improvement of GLE over RA*k*EL in Terms of Five Different Evaluation Metrics with Respect to Different Parameters *k* and M.

TABLE 8
Some Example URLs with the Four Example Tags as the High Count Tags in the Delicious Dataset

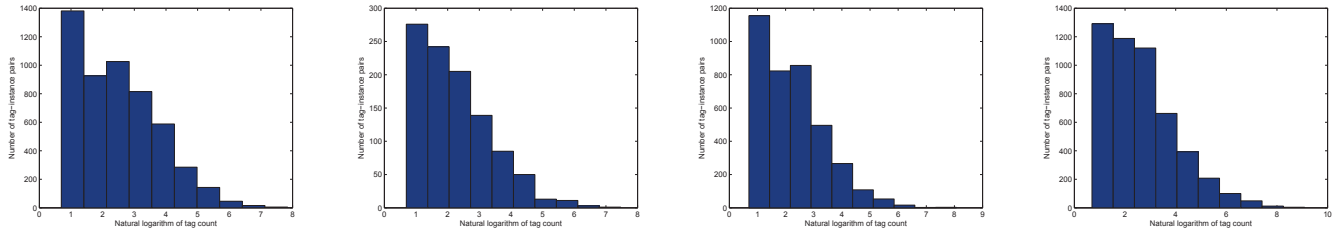
Target Tag	URL	Tag Count	Description	Other High Count Tags
art	www.drawspace.com	1649	Provides large library of free art lessons	draw, tutori
	www.threadless.com	1383	An online shirt shop that prints designs by users	t shirt, shop
	www.julianbeever.net/pave.htm	1068	Julian Beever's pavement drawings	illusion, pavem
c	www.cplusplus.com	852	The C++ resources network	programming, reference
	aelinik.free.fr/c	771	Teach Yourself C in 24 Hours	tutori, book
	www.codeblocks.org	535	Open Source, Cross-platform Free C++ IDE	ide, opensource
education	ocw.mit.edu/courses	1024	MIT opencourseware	free, course
	oreillyschool.com	330	O'Reilly School of Technology	programming, oreilly
	itunes.berkeley.edu	726	UC Berkeley on iTune	podcasting, itun
web2.0	www.go2web20.net	8900	A web 2.0 directory	statistic, reference
	digg.com	3525	A social bookmark website	social, link
	www.librarything.com	2293	A social network for sharing books	book, library

high count tags, as the results shown in Table 7. Similar observation can be drawn from the *cal500* and *majorminer* datasets. We think that the tag counts do reflect the confidence of the tags to the URLs. Low counts imply that users are less confident about such tag. Therefore, misclassifying low count tags should be assigned a less penalty.

Figure 2 shows the distribution of the tag counts of the four selected tags in the *delicious* data. The horizontal axis indicates the natural logarithm of the tag counts; and the vertical axis indicates the number of instance-tag pairs that fall into each interval. Generally, there are more low count tags than high count tags. The tag count distribution more or less follows the power law (we have a similar observation on the *cal500* and *majorminer* datasets). When using a normal multi-label classification algorithm to predict the tags, the noisy, low count tags can dominate the positive distribution and cause problems for the learners. To solve the problem, we propose using the tag count information to train a cost-sensitive multi-label classifier that minimizes the training error associated with tag counts. More specifically, the training process should give a higher importance weight (i.e., a higher misclassification cost) on correctly classifying the reliable high count tags and a lower importance weight (i.e., a lower misclassification cost) on the low count tags.

TABLE 9
Some Example URLs with the Four Example Tags as the Low Count Tags in the Delicious Dataset

Target Tag	URL	Tag Count	Description	Other High Count Tags
art	www.shoutcast.com	2	Free internet radio stations	music, radio
	www.goodyblog.com	2	A blog about babies, kids, parents, and families	blog, parent
	www.tvguide.com	2	TV Guide's official page for TV news, live event,...,etc.	television, entertain
c	www.python.org/doc	2	Python Documentation Index	python, reference
	www.cons.org/cmucl	2	A free Common Lisp implementation	lisp, programming
	www.arsmathematica.net	2	A blog about math	blog, math
education	www.solidworks.com	2	3D Mechanical Design and 3D CAD Software	softwar, cad
	www.ebookee.com	2	The Free eBooks Download Library	ebook, free
	www.robotvillage.com	2	An online store for renting robots	robot, store
web2.0	dustindiaz.com/udasss	2	UDASSS Official Documentation	ajax, javascript
	rollerweblogger.org/project	2	An open source Java blog software	blog, java
	www.pstut.com	2	Easy to follow photoshop tutorials	tutori, design



(a) art (b) c (c) education (d) web2.0

Fig. 2. Histogram for the tag counts of the four selected tags in the *delicious* data.

5.2 Evaluation Metrics for Cost-Sensitive Social Tag Prediction

The evaluation metrics used in cost-insensitive multi-label classification, such as Hamming loss, ranking loss, subset 0/1 loss, one loss, and average precision, do not consider the costs (i.e., tag counts). In this paper, we also evaluate the prediction performance from the perspective of social tag annotation and retrieval that considers the tag counts. The social tag annotation task is viewed as a multi-label classification problem since a fixed number of tags are given. The performance can be evaluated in terms of the percentage of tags that are verified correctly, or instance-based F-measure (i.e., the correct tags should receive higher scores). In the social tag retrieval task, given a specific tag as a query, the objective is to retrieve the instances (URLs or audio clips) that correspond to the tag. This can be achieved by using the multi-label classifier to determine whether, based on the prediction score for the query label, each instance is relevant to the tag. The instances are then ranked according to the relevance scores, and those with the highest scores are returned to the user. The performance can be evaluated in terms of label-based F-measure.

The cost-sensitive precision (CP) and the cost-sensitive recall (CR) are defined as follows:

$$CP = \frac{\text{Weighted sum of TP}}{\text{Weighted sum of TP+Weighted sum of FP}}, \tag{13}$$

$$CR = \frac{\text{Weighted sum of TP}}{\text{Weighted sum of TP+Weighted sum of FN}}, \tag{14}$$

where TP, FP, and FN denote the true positive, the false positive, and the false negative, respectively. The weight of each positive instance is assigned as the count of the associated tag. However, assigning a weight to each negative instance is not as straightforward because people do not use negative tags like "non-rock" and "no drum." Therefore, we assign a uniform cost to negative instances and balance the cost between positive and negative classes, i.e., the total cost of the positive instances is the same as that of the negative instances. As a result, the expected CP of a random guess baseline method will be 0.5. Then, the *cost-sensitive F-measure* based on CP and CR is calculated as follows:

$$\frac{2 \times \text{CR} \times \text{CP}}{\text{CR} + \text{CP}}. \tag{15}$$

We use the instance-based cost-sensitive F-measure and the label-based cost-sensitive F-measure to evaluate the social tag annotation task and the social tag retrieval task, respectively.

5.3 Cost-Sensitive Experiments

As mentioned in Section 4, six datasets (including *cal500*, *majorminer*, *dlc1*, *dlc2*, *dlc3*, and *dlc4*) come from the social tagging domain. These datasets, which contain the tag count information, are used for cost-sensitive social

tagging experiments. The experimental setup is the same as that in the Section 4. However, we only compare GLE with RA*k*EL, BR, and MLKNN, since these three methods perform better than CC, IBLR, and BPMLL on the experiments in Section 4. We replace the base classifier in BR by *cost sensitive* binary classifier (CSBR) as in the previous work [9]. The cost-sensitive binary classifier is implemented using LIBSVM.

The experimental results of cost-sensitive social tag annotation and retrieval are summarized in Table 10. In most cases, GLE outperforms RA*k*EL, CSBR, and MLKNN. In only one case, GLE performs slightly worse than RA*k*EL in cost-sensitive annotation; however, the difference is not significant. The average rankings of GLE on six datasets using two different metrics are 1.2 and 1.0, respectively.

6 CONCLUSION

In this paper, we have proposed a Generalized *k*-Labelsets Ensemble, which is based on the concept of label powerset method, for multi-label classification. We have proposed a novel objective function to learn the expansion coefficients of the base classifiers and found an analytic solution to learn the coefficients efficiently. GLE can be used for both multi-label classification and cost-sensitive multi-label classification. Automatic social tagging is one important application of multi-label classification. We have analyzed the effect of tag counts in automatic social tagging and shown that the tag count is important information, which indicates whether the annotated instance is a confident positive instance for that tag. By treating the tag counts as the misclassification costs, we can model the social tagging problem as a cost-sensitive multi-label classification problem. Experimental results on both multi-label classification and cost-sensitive social tagging show that GLE has better performance against other methods.

REFERENCES

[1] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Springer, 2010.

[2] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier, "On label dependence in multi-label classification," in *Proc. Int. Workshop Learning from Multi-Label Data*, 2010.

[3] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Trans. on Knowledge and Data Engineering*, vol. 23, pp. 1079–1089, 2011.

[4] T. Dietterich, "Ensemble methods in machine learning," in *Proc. Multiple Classifier Systems*, 2000.

[5] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Computer and System Sciences*, vol. 55, no. 1, 1997.

[6] L. Breiman, "Arcing the edge," Department of Statistics, University of California, Tech. Rep., 1997.

[7] R. Caruana, A. Munson, and A. Niculescu-Mizil, "Getting the most out of ensemble selection," in *Proc. Int. Conf. Data Mining*, 2006.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. Springer New York Inc., 2001.

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

[10] M.-C. Yuen, L.-J. Chen, and I. King, "A survey of human computation systems," in *Proc. Int. Conf. Computational Science and Engineering*, 2009.

[11] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green, "Automatic generation of social tags for music recommendation," in *Proc. Advances in Neural Information Processing Systems*, 2007.

[12] M. I. Mandel and D. P. W. Ellis, "Multiple-instance learning for music information retrieval," in *Proc. Int. Conf. Music Information Retrieval*, 2007.

[13] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective and Efficient Multilabel Classification in Domains with Large Number of Labels," in *Proc. ECML/PKDD 2008 Workshop Mining Multidimensional Data*, 2008.

[14] C. Sanden and J. Z. Zhang, "An empirical study of multi-label classifiers for music tag annotation," in *Proc. Int. Soc. Music Information Retrieval Conf.*, 2011.

[15] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang, "Correlative multi-label video annotation," in *Proc. ACM Int. Conf. Multimedia*, 2007.

[16] X. Li, C. G. M. Snoek, and M. Worring, "Learning social tag relevance by neighbor voting," *IEEE Trans. on Multimedia*, vol. 11, pp. 1310–1322, 2009.

[17] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Trans. Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338 – 1351, 2006.

[18] ——, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recogn.*, vol. 40, pp. 2038–2048, 2007.

[19] W. Cheng and E. Hüllermeier, "Combining instance-based learning and logistic regression for multilabel classification," *Mach. Learn.*, vol. 76, no. 2-3, pp. 211–225, 2009.

[20] J. Wang, Y. Zhao, X. Wu, and X.-S. Hua, "A transductive multi-label learning approach for video concept detection," *Pattern Recogn.*, vol. 44, no. 10-11, pp. 2274–2286, 2011.

[21] Y. Liu, R. Jin, and L. Yang, "Semi-supervised multi-label learning by constrained non-negative matrix factorization," in *Proc. National Conf. Artificial Intelligence*, 2006.

[22] L. Sun, S. Ji, and J. Ye, "Hypergraph spectral learning for multi-label classification," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining*, 2008.

[23] L. Rokach and E. Itach, "An Ensemble Method for Multi-Label Classification using an Approximation Algorithm for the Set Covering Problem," in *Proc. Int. Workshop Learning from Multi-Label Data*, 2010.

[24] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proc. ACM Int. Conf. Knowledge discovery and data mining*, 2010.

[25] M.-L. Zhang, "Lift: Multi-label learning with label-specific features," in *Proc. Int. Joint Conf. Artificial Intelligence*, 2011.

[26] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 5–32, 2001.

[27] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, 1999.

[28] H.-Y. Lo, S.-D. Lin, and H.-M. Wang, "Generalized *k*-labelset ensemble for multi-label classification," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 2012.

[29] S. Agarwal, K. Branson, and S. Belongie, "Higher order learning with graphs," in *Proc. Int. Conf. Machine learning*, 2006.

[30] D. Zhou, J. Huang, and B. Scholkopf, "Learning with hypergraphs: Clustering, classification, and embedding," in *Proc. Advances in Neural Information Processing Systems*, 2006.

[31] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[32] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, "Mulan: A java library for multi-label learning," *J. Mach. Learn. Res.*, vol. 12, pp. 2411–2414, 2011.

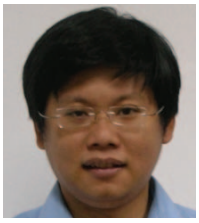
[33] M. I. Mandel and D. P. W. Ellis, "A web-based game for collecting music metadata," *J. New Music Res.*, vol. 37, no. 2, pp. 151–165, 2008.

[34] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 16, pp. 467–476, 2008.

TABLE 10
Experimental Results in Terms of Two Cost-Sensitive Evaluation Metrics. The Average Rank is the Average of the Ranks Across All Datasets. •/◦ indicates whether GLE is statistically superior/inferior to the compared algorithm (the pairwise t-test at the 5% significance level).

	GLE	RA*k*EL	CSBR	MLKNN
Cost-Sensitive F-Measure for Annotation				
cal500	0.6544 (1)	0.6436 (2) •	0.4916 (4) •	0.5889 (3) •
majorminer	0.4938 (1)	0.4885 (2) •	0.2607 (4) •	0.2964 (3) •
dlc1	0.2048 (2)	0.2052 (1)	0.0780 (4) •	0.1537 (3) •
dlc2	0.1498 (1)	0.1433 (2) •	0.0588 (4) •	0.1213 (3) •
dlc3	0.1555 (1)	0.1502 (2) •	0.0621 (4) •	0.1174 (3) •
dlc4	0.1875 (1)	0.1835 (2) •	0.0529 (4) •	0.1449 (3) •
Average Rank	1.2	1.8	4.0	3.0
Cost-Sensitive F-Measure for Retrieval				
cal500	0.4699 (1)	0.2929 (4) •	0.3341 (2) •	0.3053 (3) •
majorminer	0.3157 (1)	0.3066 (2) •	0.1147 (4) •	0.1427 (3) •
dlc1	0.2141 (1)	0.2094 (2) •	0.1721 (4)	0.1874 (3) •
dlc2	0.2801 (1)	0.2678 (2) •	0.1887 (3) •	0.1725 (4) •
dlc3	0.2515 (1)	0.2416 (2)	0.1864 (3)	0.1464 (4) •
dlc4	0.2572 (1)	0.2479 (2) •	0.1329 (4) •	0.1466 (3) •
Average Rank	1.0	2.3	3.3	3.3

[35] R. Miotto and G. R. G. Lanckriet, "A generative context model for semantic music annotation and retrieval," *IEEE Trans. Audio, Speech and Language Processing*, vol. 20, no. 4, pp. 1096–1108, 2012.
[36] M.-H. Hsu and H.-H. Chen, "Tag normalization and prediction for effective social media retrieval," in *Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence and Intelligent Agent Technology*, 2008.
[37] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Deroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recogn.*, vol. 45, no. 9, pp. 3084–3104, 2012.
[38] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proc. European Conf. Machine Learning and Knowledge Discovery in Databases*, 2009.



Shou-De Lin received the B.S. degree in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, the M.S.E.E. degree from the University of Michigan, Ann Arbor, and the M.S. degree in computational linguistics and Ph.D. degree in computer science from the University of Southern California, Los Angeles.

In 2007, he joined the CSIE Department of National Taiwan University as an Assistant Professor. He leads the Machine Discovery and Social Network Mining Lab in NTU. His research includes the areas of knowledge discovery and data mining, social network analysis, natural language processing, and machine learning.

Dr. Lin received international recognition, which includes the Best Paper Award in the IEEE Web Intelligent Conference 2003, Google Research Award in 2007, and Microsoft Research Award in 2008. He led or co-led the NTU team to win the first place of the ACMKDD Cup in 2008 and 2010, and the third place in 2009.

Hung-Yi Lo received the B.S. degree in electronic engineering and the M.S. degree in computer science and information engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 2002 and 2004, respectively. He received the Ph.D. degree in computer science and information engineering from National Taiwan University, Taipei, Taiwan, in 2013.

In 2013, he joined the Department of Information Technology and Communication, Shih Chien University, as an Assistant Professor. His research interests include data mining, machine learning, and music information retrieval. He participated in the NTU team that won the first place of the ACM KDD Cup in 2008 and 2010, and the third place in 2009.

Hsin-Min Wang (S'92-M'95-SM'04) received the B.S. and Ph.D. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1989 and 1995, respectively.

In October 1995, he joined the Institute of Information Science, Academia Sinica, Taipei, as a Post-Doctoral Fellow. He was promoted to Assistant Research Fellow, Associate Research Fellow, and then Research Fellow, in 1996, 2002, and 2010, respectively. He was an Adjunct Associate Professor with the National Taipei University of Technology, Taipei, and National Chengchi University, Taipei. His major research interests include speech processing, natural language processing, multimedia information retrieval, and pattern recognition.

Dr. Wang was a recipient of the Chinese Institute of Engineers Technical Paper Award in 1995. He was a board member, chair of academic council, and secretary-general of the Association for Computational Linguistics and Chinese Language Processing (ACLCLP). He currently serves as the vice president of ACLCLP, an editorial board member of *International Journal of Computational Linguistics and Chinese Language Processing*, and a managing editor of *Journal of Information Science and Engineering*. He is a life member of ACLCLP and the Institute of Information and Computing Machinery, and is a member of the International Speech Communication Association.