Transductive Multilabel Learning via Label Set Propagation

Xiangnan Kong, Michael K. Ng, and Zhi-Hua Zhou, *Fellow*, *IEEE*

Abstract—The problem of multilabel classification has attracted great interest in the last decade, where each instance can be assigned with a set of multiple class labels simultaneously. It has a wide variety of real-world applications, e.g., automatic image annotations and gene function analysis. Current research on multilabel classification focuses on supervised settings which assume existence of large amounts of labeled training data. However, in many applications, the labeling of multilabeled data is extremely expensive and time consuming, while there are often abundant unlabeled data available. In this paper, we study the problem of transductive multilabel learning and propose a novel solution, called TRAsductive Multilabel Classification (TRAM), to effectively assign a set of multiple labels to each instance. Different from supervised multilabel learning methods, we estimate the label sets of the unlabeled instances effectively by utilizing the information from both labeled and unlabeled data. We first formulate the transductive multilabel learning as an optimization problem of estimating label concept compositions. Then, we derive a closed-form solution to this optimization problem and propose an effective algorithm to assign label sets to the unlabeled instances. Empirical studies on several real-world multilabel learning tasks demonstrate that our TRAM method can effectively boost the performance of multilabel classification by using both labeled and unlabeled data.

Index Terms—Data mining, machine learning, multilabel learning, transductive learning, semi-supervised learning, unlabeled data

1 Introduction

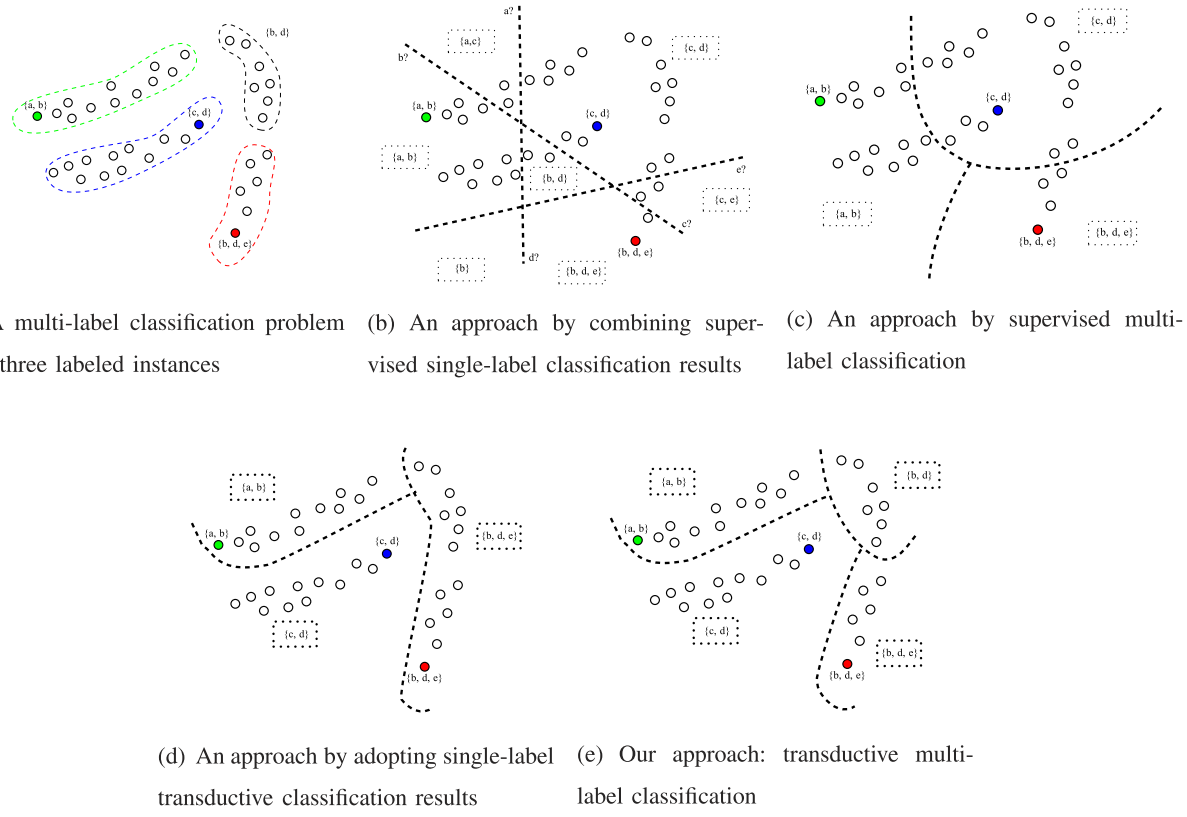
Conventional classification approaches assume that each instance is associated with only *one* class label within a number of candidate classes. However, many real-world applications often involve the scenario where each instance can be assigned with a set of *multiple* labels. For example, in image annotation, one image can be tagged with a set of multiple words, such as *urban*, *building*, and *road*, indicating the contents of the image [6], [27]. In bioinformatics, one gene sequence can be associated with a set of multiple functions, such as *metabolism* and *protein synthesis* indicating the functions of the gene sequence within a cell's life circle [10]. In text categorization, one news article can cover multiple aspects of an event, thus being assigned with a set of multiple topics, such as *economics* and *politics* [24], [28]. An effective classification model for these real-world data should be able to adopt the multiple labels of each training example and predict a label set, instead of one single label, for each testing example. Motivated by these challenges, the problem of multilabel learning has received considerable attention in the last decade.

In the literature, multilabel learning has been extensively studied [30]. Conventional approaches focus on supervised settings, which require a sufficiently large amount of labeled examples in order to train an accurate model. However, in many real world applications, the labeling process is extremely expensive and time consuming, especially with multilabel data. Creating a large training data set, where each example is labeled with a set of multiple labels within the candidate classes, is usually infeasible in practice. For example, in image annotation, human experts have to go through the entire list of all candidate words in order to decide the set of all possible tags for an image. It requires time, efforts, and excessive resources to manually tag each image with all its labels, and hence only a limited amount of labeled images can be obtained in practice. Moreover, there are often copious amounts of unlabeled images available from various sources. Thus, it is much desired that the large amount of unlabeled data can be effectively utilized together with the limited amount of labeled data to improve the multilabel classification performances. Transductive learning [32] is a type of approaches to exploit unlabeled data in classification processes. Transductive learning assumes all the testing data are available, and the goal is to achieve better performances on these testing data by exploiting the unlabeled testing data in the classification process. It has been shown useful in many single-label classification tasks [17], [32].

Formally, the transductive multilabel classification problem corresponds to predicting the label sets of a group of unlabeled instances simultaneously by utilizing the information from both labeled and unlabeled data. Transductive learning is particularly challenging in multilabel settings. The reason is that, in the single-label case, conventional transductive learning methods can be applied to propagate class labels among the unlabeled data and predict each unlabeled instance with the class label which has the highest confidence. But in multilabel cases, each instance contains multiple label concepts and the transductive classification task corresponds to finding a label set for each unlabeled instance within the space of label sets, i.e., the *power set* of all

- *X. Kong and Z.-H. Zhou are with the National Key Laboratory for Novel Software Technology, Nanjing University, Mailbox 603, 163 Xianlin Avenue, Nanjing 210023, China. E-mail: {kongxn, zhouzh}@lamda.nju.edu.cn.*
- *M.K. Ng is with the Center for Mathematical Imaging and Vision and the Department of Mathematics, Hong Kong Baptist University, Hong Kong, China. E-mail: mng@math.hkbu.edu.hk.*

Manuscript received 30 June 2010; revised 5 Apr. 2011; accepted 9 June 2011; published online 23 June 2011.
Recommended for acceptance by C. Jermaine.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2010-06-0360.
Digital Object Identifier no. 10.1109/TKDE.2011.141.



(a) A multi-label classification problem with three labeled instances

(b) An approach by combining supervised single-label classification results

(c) An approach by supervised multi-label classification

(d) An approach by adopting single-label transductive classification results

(e) Our approach: transductive multi-label classification

Fig. 1. An illustrative example for transductive multilabel classification problem.

labels. The number of possible label sets is exponential to the number of candidate labels, which is extremely large even with a small number of candidate labels.

If we consider the transductive learning and multilabel classification as a whole, the major research challenges on transductive multilabel classification can be summarized as follows:

1. *Lack of labeled data*. One fundamental problem in transductive multilabel classification lies in the labeling cost of the training data. Conventional multilabel classification approaches focuses on supervised settings [30]. The training of classification models strictly follows the assumption that there exists a large amount of labeled data. However, many real-world multilabel classification problems usually suffer from a lack of training data due to the labeling costs. Thus, it is ineffective to only use the limited training data and directly adopt existing multilabel classification approaches. For example, in Fig. 1, we show an illustrative example on multilabel classification. In Fig. 1a, we have three labeled instances with a large number of unlabeled instances. Figs. 1b and 1c show that supervised classification methods, either based upon combining single-label methods or multilabel approaches, can only make use of the information from labeled instances to make predictions on the unlabeled data, where the predictions are not quite effective when the number of labeled data is small. To cope with this issue, it is deemed that the information within the unlabeled data should be exploited to facilitate multilabel classification.
2. *Multiple labels*. Another problem in transductive multilabel classification lies in the multiple labels of each instance. Conventional transductive learning approaches focus on single-labeled classification problems [7], [38], [42]. The classification strategy strictly follows the assumption that each instance has only one label. However in multilabel classification problem, each instance can be associated with a *set* of labels within the power set of all labels. Directly adopting conventional single-label transductive approaches may not be effective for multilabel classification. For example, in Fig. 1d, we directly adopt a single-label transductive classification approach by treating each type of label set as a "class" (i.e., we directly convent a multilabel classification problem to a single-label classification problem with three classes). Since we only have a limited number of labeled instances, not every ground-truth label set has a representative instance being labeled in the training set, e.g., the label set {b, d}. Thus, the trivial application of single-label transductive classification method will not be able to predict new label sets like {b, d} in the unlabeled data.

In this paper, we study the problem of transductive multilabel classification and propose a novel solution, called TRAsductive Multilabel Classification (TRAM), to effectively assign multiple labels to each instance using both labeled and unlabeled data. Different from supervised multilabel classification methods, we estimate the label sets of the unlabeled instances effectively by utilizing the information from both labeled and unlabeled data. We first

formulate the transductive multilabel classification as an optimization problem of estimating label concept compositions. Then, we derive a closed-form solution to this optimization problem and propose an effective algorithm to assign label sets to the unlabeled instances. Empirical studies on several real-world multilabel classification tasks demonstrate that our TRAM method can effectively boost the performance of multilabel classification by using both labeled and unlabeled data.

The rest of this paper is organized as follows: Section 2 gives a brief summary of related work on multilabel classification and transductive learning. In Section 3, we formulate transductive multilabel classification as an optimization problem, and then derive a closed-form solution. Section 4 introduces label set prediction methods. Evaluation metrics used in multilabel classification are then briefly introduced and experiments of TRAM on real-world multilabel classification tasks are reported in Section 6. Finally, we give some concluding remarks in Section 7.

2 Related Work

2.1 Multilabel Classification

Multilabel classification deals with the problem where each example can belong to multiple different classes simultaneously. Traditional two class and multiclass problems can both be cast as special cases of multilabel classification problem. Thus, multilabel problems are inevitably more difficult and complicated to solve than traditional single-label problems (i.e., two class or multiclass problems). Until now, multilabel classification problem has been studied by a lot of researchers and many algorithms have been developed to solve different real-world application tasks, such as text categorization [8], [13], [20], [24], [28], [31], bioinformatics [10], [34], scene classification [4], image or video annotation [27].

Some multilabel learning algorithms are derived from traditional learning techniques. One famous approach proposed by Schapire and Singer, BOOSTEXTER [28], is extended from the popular ensemble learning method ADABOOST [11]. In the training phase, BOOSTEXTER maintains a set of weights over both training examples and their labels, which will be incrementally enlarged if examples or labels are hard to be predicted correctly. Elisseeff and Weston [10] presented a kernel method RANK-SVM for multilabel classification, by minimizing a loss function named *ranking loss*. Experimental results on the Yeast gene functional classification problem demonstrate its effectiveness. Zhang and Zhou [35] extended the lazy learning algorithm, kNN, to a multilabel version, Ml-KNN. It employs label prior probabilities gained from each example's k nearest neighbors and use Maximum a Posteriori (MAP) principle to determine labels. Extension of other traditional learning techniques have also been studied, such as probabilistic generative models [24], [31], decision trees [8], neural networks [34], maximal margin methods [15], [20], maximum entropy methods [14], [41], and ensemble methods [12].

Unlike the previous works that only consider the correlations among different categories, Liu et al. [22] present a semi-supervised multilabel classification method to exploit unlabeled data as well as category correlations. This approach is based on constrained non-negative matrix factorization. Generally, in comparison with supervised methods, semi-supervised methods can efficiently make use of the information provided by unlabeled instances. Zhou et al. [39], [40] proposed the MIML framework which deals with multilabel examples each is represented as a set of instances. Sun et al. [29] employed hypergraph spectral learning to solve multilabel classification problems.

2.2 Transductive Learning

The use of unlabeled data has been increasingly popular these years in machine learning society. As in many practical learning problems, we usually need to handle situations when a small size of labeled data with a large amount of unlabeled data is available. The unlabeled data are usually much easier to obtain but quite expensive to identify their labels. Roughly speaking, there are three main paradigms of approaches to utilize unlabeled data [38], that is, semi-supervised learning, transductive learning and active learning. Semi-supervised learning approaches attempt to automatically exploit unlabeled data usually assuming the testing data are different from the unlabeled data; transductive learning approaches attempt to automatically exploit unlabeled data where the testing data are exactly the unlabeled data; active learning approaches query an *oracle* for the labels of specific instances in the input space, in order to get better models while minimizing the number of required queries.

In this paper, we focus on transductive learning. Transductive learning was proposed by Vapnik [32] in the 1990s where all unlabeled points belong to the testing set. Many transductive learning approaches have been proposed. One famous approach is Transductive SVMs, introduced by [32] and applied to text classification by [17]. They exploit the structure in both training and testing data for better positioning the maximum margin hyperplane. Another type of approaches are graph-based methods, which define a graph with the nodes representing both labeled and unlabeled instances, and edges reflect the similarity of instances (e.g., [1], [37], [42]). Graph-based approaches usually assume label smoothness over the graph. One example is to exploit the structure of the entire data set in search for mincuts [3] or for min average cuts [18] on the graph.

3 Problem Formulation

3.1 Transductive Multilabel Classification

Before presenting the transductive multilabel classification model, we first introduce the notations that will be used throughout this paper. Let $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ denote the entire data set, which consists of n instances ($\boldsymbol{x}_i \in \mathbb{R}^d$). The data set includes both labeled and unlabeled instances. Without loss of generality, we assume the first $n_l (n_l \ll n)$ instances within $\mathcal{D}$ are labeled by $\{Y_1, \ldots, Y_{n_l}\}$, where $Y_i \subseteq \mathcal{C}$ denotes the set of multiple labels assigned to $\boldsymbol{x}_i$. Here, $\mathcal{C} = \{l_1, \ldots, l_m\}$ is the set of all possible label concepts. For convenience, we also denote $\mathcal{L} = \{1, \ldots, n_l\}$ as the index set for the labeled instances and $\mathcal{U} = \{n_l + 1, \ldots, n\}$ for the unlabeled instances ($n = n_l + n_u$). The multilabel classification task corresponds to finding an optimal label set Y_i for

each unlabeled instance $\boldsymbol{x}_i$ in the space of label sets $\mathcal{P}(\mathcal{C})$, i.e., the power set of $\mathcal{C}$.

As reviewed in Section 1, previous approaches in multilabel classification are focused on supervised settings. In this paper, we address the multilabel classification problem under the transductive setting. Our goal is to find a simple and efficient way to improve the performance of multilabel classification by exploiting both labeled and unlabeled data.

The key issue of transductive multilabel classification is how to predict *a set* of multiple labels for each unlabeled instance based on a limited number of labeled examples and a large number of unlabeled examples, which is a nontrivial task due to the following problems:

P1. How to properly estimate the composition of label concepts within the label set of an unlabeled instance based upon information from both labeled instances and all the other unlabeled instances? Intuitively, all the unlabeled instances should be estimated simultaneously and similar instances should contain similar label concepts in their label set. The question is how to jointly and effectively estimate the composition of label concepts on each instance within the unlabeled data set.

P2. How to predict the label set for each unlabeled instance based on the estimated label concept composition with only a limited number of training examples? Some types of the label sets may not even have any representative labeled data in the training set. The question is how to predict new label sets based upon only limited examples of label sets in the training data set.

In the following sections, we will introduce the optimization framework for transductive multilabel classification. Then, we will derive our closed-form solution to the optimization problem and propose an effective algorithm to predict multiple labels for each unlabeled instance.

3.2 Basic Idea

We address Problem (P1) discussed as in Section 3.1 by defining transductive multilabel classification as an optimization problem of estimating the label composition for each unlabeled instance. Our target is to first effectively estimate the label concept composition for each unlabeled instance and then make the multilabel predictions based upon the estimated concept compositions. Here, we define the *label concept composition* for a multilabel instance as follows: suppose, we have a multilabel instance $\boldsymbol{x}_i$, and its label set Y_i contains a set of multiple label concepts. For example, if we have a text document with 20 percent of the paragraphs writing about the label concept "politics" (l_1), 50 percent of the paragraphs writing about "economics" (l_2), and the rest about "culture" (l_3). Now we can say the label set for $\boldsymbol{x}_i$ is $\{l_1, l_2, l_3\}$ and the label concept composition is $(l_1 : 0.2,\ l_2 : 0.5,\ l_3 : 0.3,\ l_4 : 0, \ldots, l_m : 0)$. Here, the label concept composition means that in the text document, only 20 percent of the paragraphs were writing about concept l_1. Of course this is just an extreme example, since in most cases there is no clear "fraction" of the instance belonging to different labels. Indeed, the label concept composition expresses the typicality of the belongingness of the example to the labels, or the probability for the example to have different labels.

Formally, we denote the concept composition for instance $\boldsymbol{x}_i$ as $\boldsymbol{\alpha}_i = (\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{im})^\top$, where α_{ij} represents the fraction of label concept l_j in instance $\boldsymbol{x}_i$. Here, we assume $\alpha_{ij} \geq 0$ and $\boldsymbol{\alpha}_i^\top \mathbf{1} = 1$ $(\forall i)$. For convenience of representation, we denote $\boldsymbol{\alpha}_{(j)} = (\alpha_{1j}, \ldots, \alpha_{nj})^\top$ and illustrate our notations as follow:

$$\begin{matrix} \boldsymbol{\alpha}_{(1)} & \cdots & \boldsymbol{\alpha}_{(j)} & \cdots & \boldsymbol{\alpha}_{(m)} \\ \downarrow & & \downarrow & & \downarrow \end{matrix}$$
$$\begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1j} & \cdots & \alpha_{1m} \\ \vdots & & \vdots & & \vdots \\ \alpha_{i1} & \cdots & \alpha_{ij} & \cdots & \alpha_{im} \\ \vdots & & \vdots & & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nj} & \cdots & \alpha_{nm} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}_1^\top \\ \vdots \\ \boldsymbol{\alpha}_i^\top \\ \vdots \\ \boldsymbol{\alpha}_n^\top \end{bmatrix} \begin{matrix} \\ \\ \leftarrow \boldsymbol{x}_1 \\ \\ \leftarrow \boldsymbol{x}_i \\ \\ \leftarrow \boldsymbol{x}_n. \end{matrix}$$

In multilabel classification problems, we only know the label set of each training instance. There is no concept composition information available explicitly. We can only assume that, in a labeled training instance, all label concepts in its label set have equal weights or importance for concept composition, i.e., the ground-truth concept composition $\bar{\boldsymbol{\alpha}}_i = (\bar{\alpha}_{i1}, \ldots, \bar{\alpha}_{im})^\top$ for a labeled instance $\boldsymbol{x}_i$ is defined as follow:

$$\bar{\alpha}_{ij} = \begin{cases} \dfrac{1}{|Y_i|}, & \text{if } l_j \in Y_i, \\ 0, & \text{otherwise}, \end{cases} \qquad (i \in \mathcal{L}).$$

And our target is to estimate the concept compositions of all the unlabeled instances based upon both labeled and unlabeled data.

We assume that the optimal estimation of concept compositions should have the following property: *smoothness*, i.e., similar instances should have similar concept compositions within their label sets. If an unlabeled instance $\boldsymbol{x}_i$ is similar to a labeled instance $\boldsymbol{x}_j$, the $\boldsymbol{\alpha}_i$ should be similar to $\boldsymbol{\alpha}_j = \bar{\boldsymbol{\alpha}}_j$. Moreover, if two unlabeled instances are similar to each other, their concept compositions should also be similar. Thus, it is deemed that we need the estimate the concept compositions for all the unlabeled instances jointly/simultaneously in order to find optimal solutions on all the unlabeled data.

3.3 Optimization

In order to characterize the relation between similar instances, we build a weighted neighborhood graph $G = (V, E)$ on both labeled and unlabeled instances. Each vertex corresponds to an instance $\boldsymbol{x}_i$, an edge is put between $\boldsymbol{x}_i$ and $\boldsymbol{x}_z$, iff $\boldsymbol{x}_i$ is among the k nearest neighbors of $\boldsymbol{x}_z$ or $\boldsymbol{x}_z$ is among the k nearest neighbors of $\boldsymbol{x}_i$.

In order to reduce computational cost of kNN search among the labeled and unlabeled instances, we use kd-tree to efficiently search for approximate k nearest neighbors for each instance. Since kd-trees suffer seriously from the curse of dimensionality which will degenerate to linear search in high dimensions [33], in our work a multilabel dimensionality

reduction approach (MDDM [36]) is used before using kd-tree to construct kNN graphs, which finds a linear subspace from the original features to maximize the dependence between the label information and the subspace.

After the kNN search, we define a sparse $n \times n$ matrix W indicating the similarities among neighboring instances

$$W_{iz} = \begin{cases} \frac{1}{Z_i} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_z\|^2}{2\sigma^2}\right), & \text{if} z \in \mathcal{N}_i, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where $\mathcal{N}_i$ is the index set of ith instance's k nearest neighbors. Typically, $\|\cdot\|$ refers to the euclidean distance. And parameter σ is empirically estimated as the average distance between instances. $Z_i = \sum_{z \in \mathcal{N}_i} \exp(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_z\|^2}{2\sigma^2})$, thus $\sum_z W_{iz} = 1$ for instances.

Thus, based on the *smoothness* assumption in the previous section, we propose the following general optimization framework to estimate the optimal alpha values for unlabeled instances

$$\min_{\boldsymbol{\alpha}_{n_l+1}, \ldots, \boldsymbol{\alpha}_n} \sum_{i \in \mathcal{U}} \sum_{j=1}^{m} \left(\alpha_{ij} - \sum_{z \in \mathcal{N}_i} W_{iz} \alpha_{zj}\right)^2$$

$$s.t. \begin{array}{l} \alpha_{ij} \geq 0, \quad \sum_{j=1}^{m} \alpha_{ij} = 1 \\ \alpha_{ij} = \bar{\alpha}_{ij} \ (\forall i \in \mathcal{L}). \end{array} \quad (2)$$

Here, the $\bar{\alpha}_{ij}$ is defined as

$$\bar{\alpha}_{ij} = \begin{cases} \frac{1}{|Y_i|}, & \text{if } l_j \in Y_i, \\ 0, & \text{otherwise,} \end{cases} \quad (i \in \mathcal{L}).$$

The optimization objective is to minimize the weighted differences among the concept compositions of similar/neighboring instances. As for the labeled instances, the concept compositions are "known," and hence we put constraints $\alpha_{ij} = \bar{\alpha}_{ij}$ in the optimization. In an optimal solution to the above problem, it guarantees that the estimated concept compositions of any pair of instances, that are closely connected in the weighted neighborhood graph G, will be similar. Intuitively, the estimation process corresponds to the propagation of concept compositions among instances along the graph G.

To simplify the optimization, we have

$$\sum_{i \in \mathcal{U}} \sum_{j=1}^{m} \left(\alpha_{ij} - \sum_{z \in \mathcal{N}_i} W_{iz} \alpha_{zj}\right)^2 = \sum_{j=1}^{m} \left\| D_u \left(\boldsymbol{\alpha}_{(j)} - W \boldsymbol{\alpha}_{(j)}\right) \right\|^2,$$

where $D_u = \left(\begin{smallmatrix} 0 & 0 \\ 0 & I_u \end{smallmatrix}\right)_{(n \times n)}$, and the vector $\boldsymbol{\alpha}_{(j)} = (\alpha_{1j}, \ldots, \alpha_{nj})^\top = [\begin{smallmatrix} \boldsymbol{\alpha}_{\mathcal{L}j} \\ \boldsymbol{\alpha}_{\mathcal{U}j} \end{smallmatrix}]$. Then, the optimization problem in (2) can be simplified into matrix form as

$$\min_{\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(m)}} \sum_{j=1}^{m} \left\| D_u (I - W) \boldsymbol{\alpha}_{(j)} \right\|^2 \ s.t. \begin{cases} \boldsymbol{\alpha}_{(j)} \geq \mathbf{0}, \sum_{j=1}^{m} \boldsymbol{\alpha}_{(j)} = \mathbf{1} \\ \boldsymbol{\alpha}_{\mathcal{L}j} = \bar{\boldsymbol{\alpha}}_{\mathcal{L}j}. \end{cases} \quad (3)$$

3.4 A Closed-Form Solution

We note that the objective function and the constraints in (3) are convex. Therefore, a global minimizer exists [25]. Let $A = I - W$ in (3). We partition the matrix A and $\boldsymbol{\alpha}_{(j)}$ vectors into blocks according to the labeled and unlabeled data,

$$A = \begin{bmatrix} A_{\mathcal{LL}} & A_{\mathcal{LU}} \\ A_{\mathcal{UL}} & A_{\mathcal{UU}} \end{bmatrix} \quad \text{and} \quad \boldsymbol{\alpha}_{(j)} = \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{L}j} \\ \boldsymbol{\alpha}_{\mathcal{U}j} \end{bmatrix}, (j = 1, \ldots, m).$$

By ignoring the constraints $\boldsymbol{\alpha}_{(j)} \geq \mathbf{0}$, the Lagrange function for (3) becomes

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \frac{1}{2} \sum_{j=1}^{m} \left\| D_u A \boldsymbol{\alpha}_{(j)} \right\|^2 - \boldsymbol{\beta}^\top \left(\sum_{j=1}^{m} \boldsymbol{\alpha}_{\mathcal{U}j} - \mathbf{1}\right) - \sum_{j=1}^{m} \gamma_j{}^\top (\boldsymbol{\alpha}_{\mathcal{L}j} - \bar{\boldsymbol{\alpha}}_{\mathcal{L}j}),$$

where $\boldsymbol{\beta} \geq \mathbf{0}$ and $\boldsymbol{\gamma}_j \geq \mathbf{0}$. The optimal condition for $\boldsymbol{\alpha}_{(j)}$ is

$$\frac{\partial L}{\partial \boldsymbol{\alpha}_{(j)}} = A^\top D_u^\top D_u A \boldsymbol{\alpha}_{(j)} - \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\beta} \end{bmatrix} - \begin{bmatrix} \gamma_j \\ \mathbf{0} \end{bmatrix} = 0. \quad (4)$$

By summing over the optimal conditions in (4) for all $\alpha_{(j)}$ $(j = 1, \ldots, m)$, we have

$$\sum_{j=1}^{m} (A^\top D_u^\top D_u A \boldsymbol{\alpha}_{(j)}) = \begin{bmatrix} \sum_{j=1}^{m} \gamma_j \\ m\boldsymbol{\beta} \end{bmatrix}.$$

Then, using the constraints $\sum_{j=1}^{m} \boldsymbol{\alpha}_{(j)} = \mathbf{1}$, we have

$$A^\top D_u^\top D_u A \mathbf{1} = \begin{bmatrix} \sum_{j=1}^{m} \gamma_j \\ m\boldsymbol{\beta} \end{bmatrix}.$$

Notice that the $A\mathbf{1} = (I - W)\mathbf{1} = \mathbf{1} - W\mathbf{1} = \mathbf{0}$. So, the following equations can be derived, $\beta = \mathbf{0}$, $\sum_{j=1}^{m} \gamma_j = 0$ and then we substitute them into (4),

$$\begin{bmatrix} A_{\mathcal{UL}}^\top A_{\mathcal{UL}} & A_{\mathcal{UL}}^\top A_{\mathcal{UU}} \\ A_{\mathcal{UU}}^\top A_{\mathcal{UL}} & A_{\mathcal{UU}}^\top A_{\mathcal{UU}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{L}j} \\ \boldsymbol{\alpha}_{\mathcal{U}j} \end{bmatrix} = \begin{bmatrix} \gamma_j \\ \mathbf{0} \end{bmatrix}.$$

Therefore, we get

$$A_{\mathcal{UU}}^\top (A_{\mathcal{UL}} \boldsymbol{\alpha}_{\mathcal{L}}^j + A_{\mathcal{UU}} \boldsymbol{\alpha}_{\mathcal{U}}^j) = \mathbf{0}. \quad (5)$$

Here, $A_{\mathcal{UU}}^\top$ is guaranteed to be nonsingular for a connected graph [2]. By substituting the constraints $\boldsymbol{\alpha}_{\mathcal{L}j} = \bar{\boldsymbol{\alpha}}_{\mathcal{L}j}$ into (5), the optimal alpha values of unlabeled instances for class j, i.e., $\alpha_{\mathcal{U}j}$, can be calculated by the following linear equation:

$$A_{\mathcal{UU}} \boldsymbol{\alpha}_{\mathcal{U}j} = -A_{\mathcal{UL}} \bar{\boldsymbol{\alpha}}_{\mathcal{L}j}, \quad (6)$$

which is a sparse, symmetric linear system. The number of equations equals to n_u and the number of nonzero entries is less than $(k+1) \times n_u$. Here, the solution $\boldsymbol{\alpha}_{\mathcal{U}j}$ is guaranteed to exist and be unique with values guaranteed to lie between 0 and 1. The proofs can be found in [25], we put them in the Appendix section to make the paper self-contained.

After the optimal alpha values are solved in (6), we will show how to use the optimal alpha values to predict a set of labels for each unlabeled instance in the following section.

4 Label Set Prediction

In this section, we address Problem (P2) as discussed in Section 3.1 to predict a set of labels for each unlabeled instance based on the optimal alpha values. We propose a supervised version of label set prediction method, and a transductive version of label set prediction method. The differences between these two versions are as follows: 1) In the supervised version, we only make use of the labeled instances to learn a *threshold* function and directly predict a label set based upon the estimated alpha values. 2) In the transductive version, we make use of both labeled and unlabeled instances to estimate the *cardinality* of the label set for each unlabeled instance. After the label set cardinality is estimated, we sort all the labels based on instance's concept composition (i.e., the estimated alpha values), and predict the label set with the top ranked labels with the estimated label set cardinality.

4.1 Supervised Label Set Prediction via Linear Regression

In this section, we propose a supervised label set predicting mechanism based on the optimal alpha values on unlabeled instances. More precisely, a label set predicting function $f(\alpha(\boldsymbol{x}))$ is modeled by a linear function $f(\alpha(\boldsymbol{x})) = P\alpha(\boldsymbol{x})$, where $\alpha(\boldsymbol{x}) = (\alpha_1(\boldsymbol{x}), \ldots, \alpha_m(\boldsymbol{x}))$ is the m-dimensional vector of the optimal alpha values for unlabeled instance $\boldsymbol{x}$, and P is a $m \times m$ linear transformation matrix. The procedure used to learn the optimal linear transformation matrix P is described as follows:

We perform the leave-one-out process using (6) on the training set to calculate the estimated optimal alpha values on each training instance, denoted by $\hat{\alpha}_{ij}$s. By combining $\hat{\alpha}_{ij}, (i \in \mathcal{L})$ into a vector, the estimated alpha outputs on every training instance can be solved by the following equation:

$$\hat{\boldsymbol{\alpha}}_{\mathcal{L}j} = (I - A_{\mathcal{L}\mathcal{L}})\boldsymbol{\alpha}_{\mathcal{L}j} = W_{\mathcal{L}\mathcal{L}}\boldsymbol{\alpha}_{\mathcal{L}j} \qquad (j = 1, \ldots, m). \tag{7}$$

Suppose, the output vector for instance i is $\hat{\boldsymbol{\alpha}}_i = (\hat{\alpha}_{i1}, \hat{\alpha}_{i2}, \ldots, \hat{\alpha}_{im})^\top$ $(i \in \mathcal{L})$. The ground-truth labels for instance i are known, i.e., $Y_i \subseteq \mathcal{C}$. Here, for convenience of prediction, we denote the vector of ground-truth labels as $\tilde{\boldsymbol{y}}_i \in \{-1, 1\}^m$. Then, transformation matrix P can be calculated by minimizing the following sum-of-squares error function with a regular term,

$$P = \arg\min_P \sum_{i \in \mathcal{L}} \|\tilde{\boldsymbol{y}}_i - P\hat{\boldsymbol{\alpha}}_i\|_2^2 + \lambda \sum\nolimits_j \|P_{j\cdot}\|_2^2,$$

where $P_{j\cdot}$ denotes the jth row of matrix P. Then, the solution is

$$P = \tilde{\boldsymbol{y}}_{\mathcal{L}}\hat{\boldsymbol{\alpha}}^\top(\hat{\boldsymbol{\alpha}}\hat{\boldsymbol{\alpha}}^\top + \lambda I)^{-1}. \tag{8}$$

Here, λ is used to avoid the singularity of the linear system in (8). In practice, we set λ as a very small number (it is set to be 1×10^{-7} in the experiment). Then, with the linear transforms matrix P, we can predict label vector for unlabeled instances from their optimal alpha values by

$$\boldsymbol{y}_i = sign(P\boldsymbol{\alpha}_i) \quad (\forall i \in \mathcal{U}).$$

Where $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{im})^\top$. Then, the outputted label set for the ith instance is $Y_i = \{l_j : y_{ij} = 1\}$.

4.2 Transductive Label Set Prediction

In this section, we propose a transductive label set predicting method based on the optimal alpha values. Different from the supervised method in the previous section, the transductive label set prediction method can utilize information from both labeled and unlabeled data.

As we have already found the optimal alpha values for any unlabeled instance $\boldsymbol{x}_i$. A sorted list of all potential labels for $\boldsymbol{x}_i$ can be find by ranking all candidate labels using their alpha values in descending order. The larger the alpha value is the more likely $\boldsymbol{x}_i$ will have the corresponding label. For example, suppose there are three class labels l_1, l_2, l_3, and the optimal alpha values $\boldsymbol{x}_i$ are $(\alpha_{i1} = 0.25, \alpha_{i2} = 0.4, \alpha_{i3} = 0.35)$. The sorted list for instance $\boldsymbol{x}_i$ is (l_2, l_3, l_1). Now the only problem is how to decide how many labels should be predicted into the label set of $\boldsymbol{x}_i$ using both labeled and unlabeled data. As long as the number of labels on instance $\boldsymbol{x}_i$ is decided, say θ_i, we can predict the top θ_i labels on the sorted list as the label set of instance $\boldsymbol{x}_i$.

Let θ_i denote the number of labels in the label set for instance $\boldsymbol{x}_i$. The θ_i values on the labeled instances are fixed according to the ground truth of their label sets, i.e., $\theta_i = |Y_i|$ $(i \in \mathcal{L})$. For unlabeled data, the number of labels (θ_i) should be a non-negative integer, here we can relax the $\theta_i \in \mathbb{R}$ and $\theta_i \geq 0$ $(i \in \mathcal{U})$. Then, by using similar *smoothness* assumption in Section 3.2, we assume similar instances should have similar number of labels.

Then, the optimal θ_i values can be solved by the following optimization problem:

$$\min_{\theta_1, \ldots, \theta_n} \sum_{i \in \mathcal{U}} \left(\theta_i - \sum_{z \in \mathcal{N}_i} W_{iz}\theta_z \right)^2$$

$$s.t.\ \theta_i = |Y_i| \ (\forall i \in \mathcal{L}). \tag{9}$$

Similar to the optimization problem in Section 3.4, optimal solutions of the (9) can be found by solving the following linear equation:

$$A_{\mathcal{U}\mathcal{U}}\boldsymbol{\theta}_{\mathcal{U}} = -A_{\mathcal{U}\mathcal{L}}\boldsymbol{\theta}_{\mathcal{L}}, \tag{10}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)^\top = [\boldsymbol{\theta}_{\mathcal{L}}; \boldsymbol{\theta}_{\mathcal{U}}]$. We can now use the optimal solutions (θ_i^*) on each unlabeled data to predict its label set. The number of labels for unlabeled instance $\boldsymbol{x}_i$ is predicted as the closest integer to θ_i^*.

The TRAM method is briefly summarized in Fig. 2. Note the default label set prediction method in TRAM is the transductive version described in Section 4.2. The TRAM method using supervised version of label set prediction in Section 4.1 is denoted as TRAM_S.

5 Computational Complexity

In this section, we briefly analyze the computational complexity of TRAM as follows: beyond the computational cost of MDDM dimensionality reduction $(O(m \cdot n))$ in the training step and the neighborhood graph searched by kd-tree $(O(n \log n))$ in the testing step, the alpha solutions and the label learning procedure of TRAM involve the following costs: in the worst case, the least squares solution of the linear systems in (6) requires $O(n_u^3 + n_l \cdot n_u))$ operations when all data points are connected in a full graph

$(Y_U, \boldsymbol{\alpha}_{\mathcal{U}})$ = TRAM(X, $Y_{\mathcal{L}}$)

Input:

$X : (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$ encoding features of the whole data set

$Y_{\mathcal{L}}$: $(Y_1, \cdots, Y_l)$ encoding labels of training set

Process:

1 Construct kNN graph among instances.

2 Initialize the similarities on each edge as $W_{iz} = \exp(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_z||^2}{2\sigma^2})$ and normalize to $\sum_z W_{iz} = 1$;

3 Determine the $\boldsymbol{\alpha}_{\mathcal{U}}^j$ values for all unlabeled data by solving the linear system in Eq.6;

\# Supervised version:

4 Compute the label set prediction matrix P by solving Eq.8;

5 Predict label set for each unlabeled instance by $\boldsymbol{y}_i = sign(P\boldsymbol{\alpha}_i) \quad (\forall i \in \mathcal{U})$.

\# Transductive version:

4 Compute sorted label list on each unlabeled instance using optimal alpha values in Step 3;

5 Determine the optimal number of labels on each instance by solving the linear equation in Eq. 10.

Output:

$Y_{\mathcal{U}}$: the predicted labels for unlabeled instances.

$\boldsymbol{\alpha}_{\mathcal{U}}$: the alpha value outputs for unlabeled instances.

Fig. 2. The TRAM algorithm.

(i.e., $k = n$). However, this cost can be significantly reduced using a k-nearest neighbor graph ($k \ll n$) which leads directly to a sparse matrix ($A_{\mathcal{UU}}$). Thus, the linear systems are large, sparse, and symmetric, many good solvers can be employed, e.g., direct methods (e.g., LU factorizations), or iterative solvers [16]. In practice, "the cost of computing the sparse LU factorization depends in a complicated way on the size of $A_{\mathcal{UU}}$, the number of nonzero elements, its sparsity pattern, but is often dramatically smaller than the cost of a dense LU factorization. In many cases the cost grows approximately linearly with n_u, when n_u is large. This means that when $A_{\mathcal{UU}}$ is sparse, we can solve $A_{\mathcal{UU}}\boldsymbol{\alpha}_{\mathcal{U}j} = \boldsymbol{b}$ very efficiently, often with an order approximately n_u" [5].

For simplicity, we have used QR factorization designed for sparse matrix in MATLAB to compute the R factor very cheaply, which avoids the expensive computation of an explicit Q, details are described in [23]. Then, for label learning procedure of TRAM, the computation of $\hat{\alpha}_L^j$ and transforms matrix P costs $O(m \cdot n_l)$ and $O(n_l \cdot m + m^3)$, respectively.

The computational complexity of RANK-SVM [10] is currently of the order $O(m \cdot {n_l}^2)$ in each iteration for training. Ml-KNN [35] as a lazy learning algorithm requires $(O(n_l^2 + n_l \cdot m))$ for training, and $O(n_l \cdot n_u + n_u \cdot m)$ for testing. BOOSTEXTER [28] requires $O(n_l \cdot m)$ for each iteration round in training with additional cost for the training of base learners. CNMF [22] as a transductive learning method requires $O(n^2)$ for similarity calculation between samples and $O(m \cdot n_u)$ in each iteration for testing.

6 EXPERIMENTS

In this section, we show the performance of TRAM on several real-world multilabel classification tasks. Table 1 summarizes the characteristics of the data sets used. For comparison, we also compare with several general-purpose multilabel classification algorithms, including CNMF [22], BOOSTEXTER [28], RANK-SVM [10], and Ml-KNN [35], which are applicable to various multilabel problems, and represent the state-of-the-art techniques in multilabel classification:

1. TRAM. The proposed algorithm TRAM, i.e., a transductive multilabel classification algorithm via label set propagation (implementation in MATLAB). For label set prediction step, the default setting is using transductive version of label set prediction. TRAM with supervised version of label set prediction is also compared, denoted by TRAM_S.
2. CNMF. The CNMF [22] is a semi-supervised multilabel classification algorithm by constrained nonnegative matrix factorization. The key assumption behind CNMF is that two instances tend to have large overlap in their assigned class memberships if they share high similarity in their input patterns. By minimizing the difference between inputs similarity with class label overlaps, CNMF can determine the labels of unlabeled data.
3. BOOSTEXTER. The BOOSTEXTER [28] (implementation in C) is a Boosting style multilabel ranking

TABLE 1
Summary of Experimental Data Sets

Task Studied	Data Set	# Instances	# Attributes	# Labels
Automatic Image Annotation	annotation	4,800	500	43
Gene Functional Analysis	yeast	2,417	103	14
Web Page Categorization	yahoo (11 subsets)	5,000	(462 $\sim$ 1,047)	(21 $\sim$ 40)
Text Categorization	RCV1-v2	6,000	662	54
Natural Scene Classification	scene	2,407	294	6

system, which has been shown with excellent performance in previous studies, especially on text categorization tasks.

4. RANK-SVM. The RANK-SVM [10] (implementation in MATLAB) is an SVM style multilabel classification algorithm which minimizes ranking loss directly and has also exhibited excellent performance in previous studies.
5. ML-KNN. The ML-KNN [35] (implementation in MATLAB) is a kNN style multilabel classification algorithm which often outperforms other existing multilabel algorithms.

Parameters are used in their default settings unless otherwise specified. For BOOSTEXTER,[1] the number of boosting rounds is set to 500 because on all data sets studied in this paper, the performance of BOOSTEXTER will not significantly change after the specified boosting rounds; For RANK-SVM the best parameters reported in the literature [10] are used; For CNMF, the best parameters in [22] are used.

Our TRAM implementation is in MATLAB and the size of neighbors k is 10. Moreover, the influence of TRAM's parameters will be discussed in Section 6.7.

6.1 Evaluation Metrics

Multilabel classification systems require much more complicated evaluation criteria than traditional single-label systems. In this section, we briefly summarize the criteria used for performance evaluation from various perspectives. Since our approach not only produces a ranked list of class labels, but also produces a predicted label set, in this paper, we employ two sets of evaluation metrics to evaluate the performance of label ranking as well as the label set prediction. Adopting the same notations as used in Section 3, for a test set $\mathcal{D}_{\mathcal{U}} = \{(\boldsymbol{x}_{l+1}, Y_{l+1}), \ldots, (\boldsymbol{x}_n, Y_n)\}$, the following multilabel evaluation criteria are used in this paper, which have been used in [10], [28], [34], and [35].

Label Set Prediction Performances. The first set of evaluation criteria are concerning algorithm's performance on label set prediction for each instance. It is based on multilabel classifier's label set prediction function $h : \mathbb{R}^d \rightarrow \mathcal{P}(\mathcal{C})$, assume $h(\boldsymbol{x}_i)$ be the set of labels predicted by a multilabel classifier for instance $\boldsymbol{x}_i$.

1. http://www.cs.princeton.edu/~schapire/boostexter.html.

1. *MicroF1*. Evaluates both microaverage of Precision and micro average of Recall with equal importance.

$$MicroF1 = \frac{2 \times \sum_{i \in \mathcal{U}} |h(\boldsymbol{x}) \cap Y_i|}{\sum_{i \in \mathcal{U}} |h(\boldsymbol{x})| + \sum_{i \in \mathcal{U}} |Y_i|}.$$

The bigger the value, the better the performance. This criterion has been used in [19], and [22].

2. *Hamming loss*. Evaluates how many times an instance-label pair is misclassified.

$$HammingLoss(h, \mathcal{D}_{\mathcal{U}}) = \frac{1}{|\mathcal{D}_{\mathcal{U}}|} \sum_{i \in \mathcal{U}} \frac{1}{m} |h(\boldsymbol{x}_i) \Delta Y_i|,$$

where Δ stands for the symmetric difference of two sets. The smaller the value, the better the performance.

Label Ranking Performances. The second group of evaluation criteria are concerning algorithm's label ranking performance for each instance, they are based on the real-valued output function $f : \mathbb{R}^d \times \mathcal{C} \rightarrow \mathbb{R}$ of each algorithm. For TRAM method, the optimal alpha values are used as the real-valued outputs.

3. *Ranking loss*. Evaluates the average fraction of label pairs that are not correctly ordered.

$$RankLoss(f, \mathcal{D}_{\mathcal{U}}) = \frac{1}{|\mathcal{D}_{\mathcal{U}}|} \sum_{i \in \mathcal{U}} \frac{1}{|Y_i||\overline{Y_i}|} |\{(y_1, y_2) \in Y_i \times \overline{Y_i} | f(\boldsymbol{x}_i, y_1) \leq f(\boldsymbol{x}_i, y_2)\}|,$$

where the $\overline{Y_i}$ denotes the complementary set of Y_i in $\mathcal{C}$. The performance is perfect when $RankLoss(f) = 0$. The smaller the value, the better the performance.

4. *Average Precision*. Evaluates the average fraction of labels ranked above a particular label $y \in Y_i$ which actually is in Y_i

$$AvePrec(f, \mathcal{D}_{\mathcal{U}}) = \frac{1}{|\mathcal{D}_{\mathcal{U}}|} \sum_{i \in \mathcal{U}} \frac{1}{|Y_i|} \sum_{y \in Y_i} \frac{|\{y' \in Y_i | r_f(\boldsymbol{x}_i, y') \leq r_f(\boldsymbol{x}_i, y)\}|}{r_f(\boldsymbol{x}_i, y)}.$$

The bigger the value, the better the performance.

Note that all the criteria evaluate the performance of multilabel classification systems from different aspects. Usually, few algorithms could outperform another algorithm

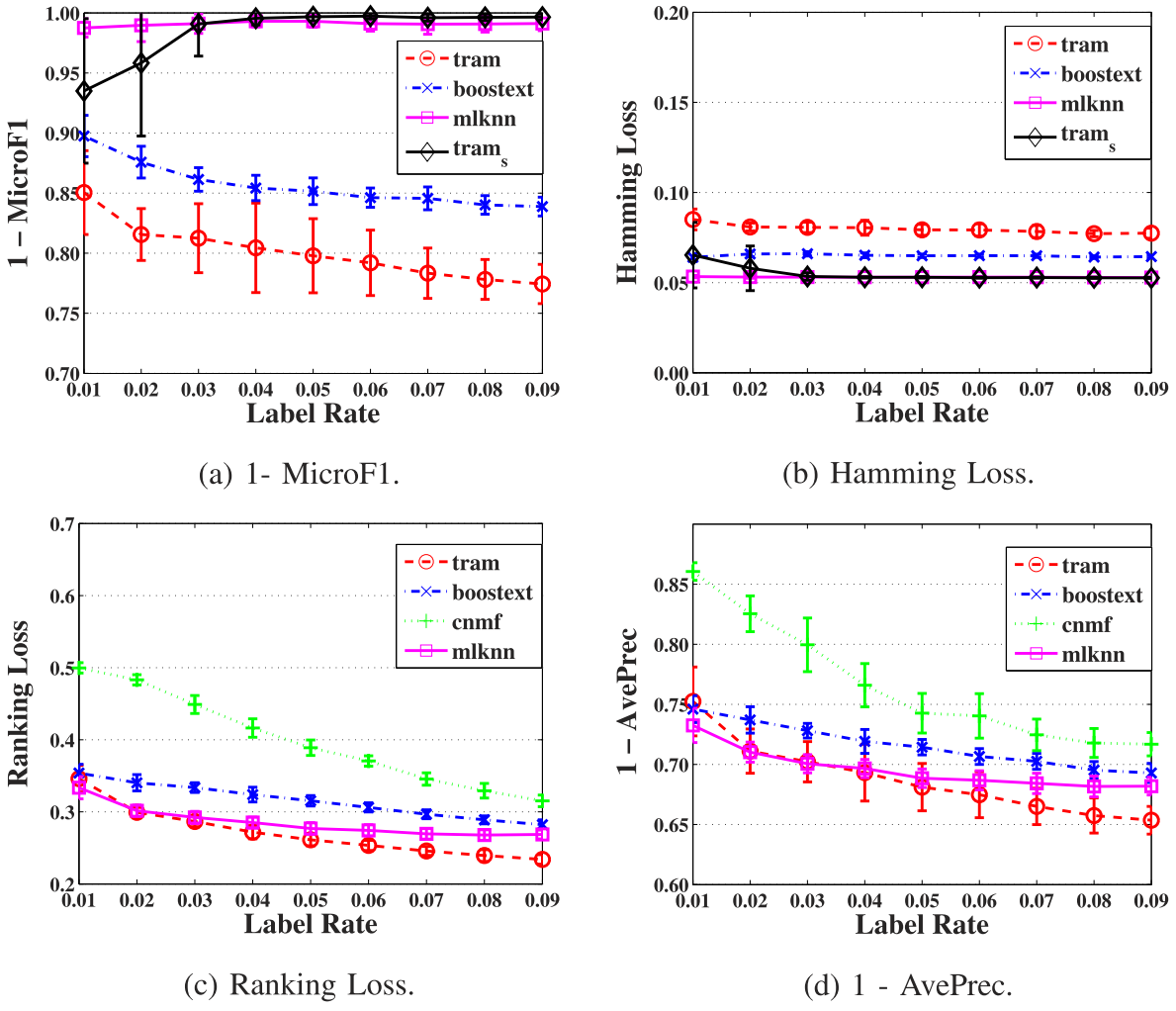


Fig. 3. Results on automatic image annotation task under different label rates. The lower the value, the better the performance. Along with the curves, we also plot the mean ± std on each point for different random data set partitions.

on all those criteria. In order to make our evaluation criteria more comprehensive, we will use the value of $1 - AvePrec$ and $1 - MicroF1$ to replace the original *Average Precision* and *MicroF1*. Thus, under all evaluation criteria, smaller values are always indicating better performances.

6.2 Application to Automatic Image Annotation

We test the automatic image annotation task on Corel data set used in [9]. The original data set contains 5,000 images each was segmented into several regions and tagged with several words. The regions of similar features are clustered into 500 clusters, known as blobs [9]. Then, each image is represented by a binary vector of these 500 blobs. The average annotated words for each image is 3.5. We remove the words that occur less than 100 times, and obtain 4,800 images and 43 annotation words.

This data set is partitioned randomly into labeled/unlabeled data sets according to certain ratios. In detail, we randomly draw from 1 to 9 percent of the data as labeled training examples and randomly selection 50 percent of the data from the remaining as unlabeled examples. For instance, assuming the data set contains 4,800 examples and the label rate is 1 percent, we randomly draw 48 examples as labeled training examples; and 2,400 examples from the remaining data set as unlabeled testing examples. Thirty runs of experiments are conducted under every label rate; in each run, algorithms are evaluated on random data set partitions. We also compared against the RANK-SVM algorithm [10], but on the Image Annotation data set alone, the algorithm did not get good results.

The results of multilabel classification on image annotation task are shown in Fig. 3.[2] In label set prediction performances, TRAM with transductive version of label set prediction gets much better performances on MicroF1 than other algorithms including the supervised version of TRAM on label set prediction (i.e., TRAM_S). It is not strange that the classic multilabel classification methods such as ML-KNN could not work well in this setting since they were designed for supervised scenarios where there are lots of labeled training examples. When the number of labeled data is extremely small, the supervised version of TRAM becomes unstable in MicroF1 performance, since TRAM_S only use labeled data to train the label set prediction function, and the supervised information in labeled data can be weak in these cases. Although TRAM_S gets better performance in Hamming Loss than TRAM, this may be explained by the fact that Hamming Loss treats two types of

2. Evaluation results of *Hamming Loss* and *MicroF1* are not available for CNMF.

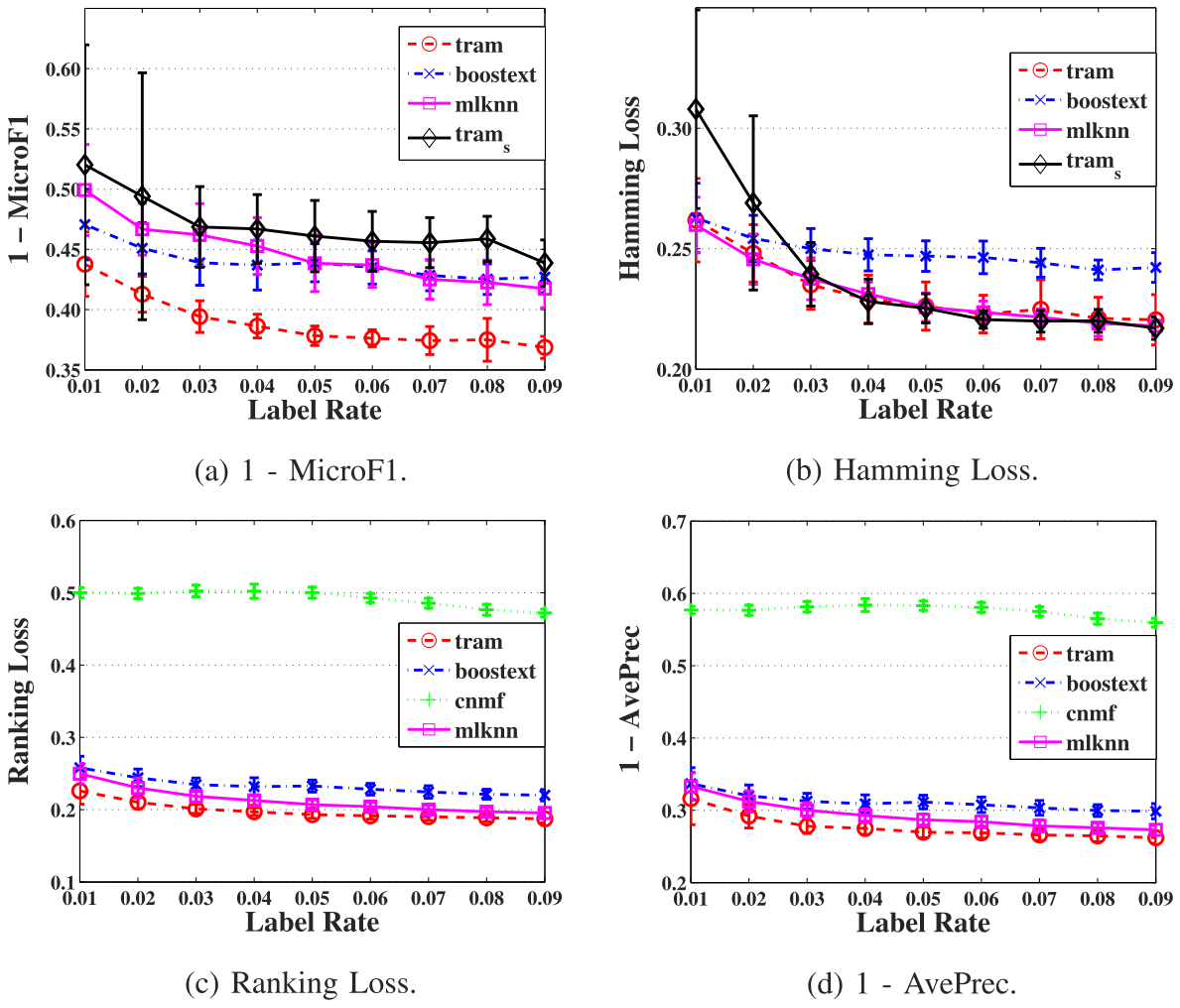


Fig. 4. Results on yeast gene function analysis task with different label rates. The lower the value, the better the performance. Along with the curves, we also plot the mean $\pm$ std on each point for different random data set partitions.

misclassification errors (false alarm and missing prediction) equally, which is quite similar to the sum-of-squares error function in TRAM$_S$'s label set prediction step. In image annotation task each image usually has a small number of labels compared with the large number of classes. In other words, the label distribution on each class is quite imbalanced. Classification methods like TRAM$_S$ with better Hamming Loss and bad MicroF1 are biased to avoid predicting any label for each instance. TRAM$_S$ obtains bad Micro-Recall performance and good Micro-Precision performance. Since MicroF1 is treating both Micro-Precision and Micro-Recall equally, MicroF1 can better evaluated the label set prediction performances in this case.

On evaluation criteria concerning label ranking, i.e., ranking loss and average precision, TRAM's performances are better than other methods. TRAM can make use of both labeled and unlabeled data to get an optimal set of alpha values on each unlabeled instance, which may significantly help to improve the ranking performance especially when there are not sufficient but reasonable size of training data.

6.3 Application to Yeast Gene Functional Analysis

The task of the yeast gene functional analysis has been studied as a multilabel classification problem in many works (e.g., [10] and [26]). Following [10], we aim at predicting the functional classes in the gene of yeast *Saccharomyces cerevisiae*. These functional classes are structured into four levels of hierarchies.[3] As in [10], only top level hierarchy is considered. The whole data set has 2,417 instances of genes and 14 possible class labels. Each of the gene is represented by a 103D vector and the average number of class labels is 4.24 ± 1.57 for each instance.

The data set is partitioned randomly into labeled/unlabeled data sets according to certain ratios, the same setup as in the automatic image annotation task. Thirty runs of experiments are conducted under every label rate; in each run, algorithms are evaluated on random data set partitions and the average performance is recorded.

The results of multilabel classification on Yeast Gene Functional Analysis are shown in Fig. 4. For label set prediction performances, TRAM gets better performances than the other methods on MicroF1, while getting comparable performances with other methods on Hamming Loss. For label ranking performances, TRAM outperforms the other methods on all evaluation criteria and all label rates.

3. Details in http://mips.gsf.de/proj/yeast/catalogues/funcat/.

TABLE 2
Data Subsets Used in the Automatic Web Page Categorization Task

	Number of	Vocabulary	Training Set		Test Set	
Data Subset	Labels	Size	*MDoc%*	*#AveLabel*	*MDoc%*	*#AveLabel*
Arts&Humanities	26	462	44.50%	1.627	43.63%	1.642
Business&Economy	30	438	42.20%	1.590	41.93%	1.586
Computers&Internet	33	681	29.60%	1.487	31.27%	1.522
Education	33	550	33.50%	1.465	33.73%	1.458
Entertainment	21	640	29.30%	1.426	28.20%	1.417
Health	32	612	48.05%	1.667	47.20%	1.659
Recreation&Sports	22	606	30.20%	1.414	31.20%	1.429
Reference	33	793	13.75%	1.159	14.60%	1.177
Science	40	743	34.85%	1.489	30.57%	1.425
Social&Science	39	1,047	20.95%	1.274	22.83%	1.290
Society&Culture	27	636	41.90%	1.705	39.97%	1.684

"$MDoc\%$" denotes the percentage of web pages belonging multiple categories, and "#AveLabel" represents the average number of labels for each web page.

6.4 Application to Automatic Web Page Categorization

The web page categorization task has been studied in [20], [31], [35]. In this experiment, our task is to classify web pages in a collection of eleven data subsets.[4] The web pages were collected from the "yahoo.com" domain, represented by the form of "*Bag-of-Words*," i.e., each dimension of the feature vector represents the number of times a word appearing in the web page. Each data subset corresponds to a top-level category (e.g., "Entertainment," "Education," etc.), which contains 2,000 web pages in the training set and 3,000 web pages in the test set. Each web page is assigned to several second-level categories and may belongs to multiple categories simultaneously.

The web page data subsets are briefly summarized in Table 2. Details of these data subsets can also be found in [35]. Comparing with the data sets used in previous tasks, the number of instances and size of vocabulary size in these 11 data subsets are much larger. Furthermore, a larger percentage of instances (about 30 to 40 percent) are assigned to multiple labels. Thus, the data subsets used in automatic web page categorization tasks are more difficult to learn from.

The same experiment settings are used to randomly partition the data subset into labeled/unlabeled sets according to different label rates. To make a more meaningful comparison among 11 data subsets, we used the geometrical means of the evaluation values across the 11 data subsets instead of simply using the average values. Such that, only the algorithms that have good performances over all 11 data subsets can have good performance values after the geometrical means.

The results of multilabel classification on automatic web page categorization task are shown in Fig. 5. For label set prediction performances, TRAM has better MicroF1 results after the geometrical mean over 11 data subsets on this task, in other words, TRAM achieves better performances on average over 11 data subsets. On web page categorization task, the average number of labels on each webpage is much smaller than the number of classes. Thus, TRAM's performance on Hamming Loss is not as good as TRAM_S, but the difference is not quite significant. For label ranking performances, TRAM gets better or comparable performances than other methods after the geometrical mean on 11 data subsets.

6.5 Application to Text Categorization

In this Section, we perform text categorization using RCV1-v2 data set [21]. The original data set has 804,414 documents, and 47,236 features. We use a benchmark subset, rcv1v2 (topics;subset),[5] which contains 6,000 documents. We removed the words that occur less than 200 times and topics with less than 50 positive examples, thus obtain 662 words and 54 topics. Note that the number of examples in this subset (6,000) is much larger than in the previous tasks in this paper. Here, the dimensionality (662) is also very high.

The results of multilabel classification on automatic text categorization task are reported in Fig. 6. The performance of TRAM and BOOSTEXTER get best performances on label set prediction and label ranking. BOOSTEXTER is originally designed and one of the state-of-the-art multilabel classification methods on text data. Although on some label rates, BOOSTEXTER gets better performances than TRAM, but

4. Data set available at http://www.kecl.ntt.co.jp/as/members/ueda/yahoo.tar.gz.

5. Data set available at http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html.

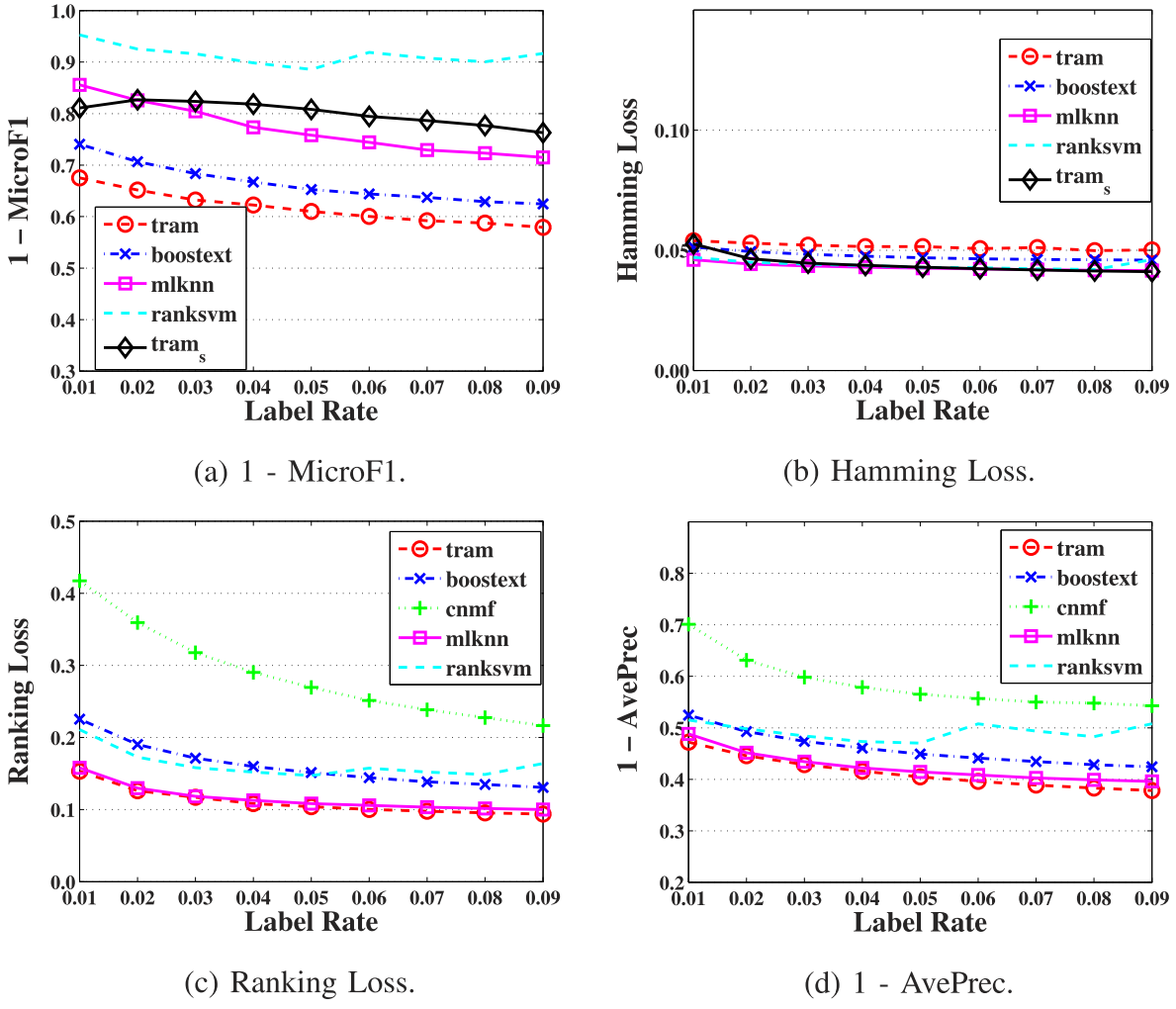


Fig. 5. Results on automatic web page categorization task with different label rates. Note that the values in each figure are reported as the geometrical means across the 11 data subsets.

TRAM is still getting better performances than the other comparing methods on MicroF1, Ranking Loss, and Average Precision.

6.6 Application to Natural Scene Classification

The last multilabel task studied in this paper is natural scene classification. The data set is relatively small, and consists of 2,400 natural scene images belonging to different classes, which is also used in [4]. Following [4], we convert each color image to the CIE Luv space, where the euclidean distances closely correspond to the color differences perceived by human. Then, the image is divided into 7×7 blocks using grids of equal width, and in each block the first and second moments of each color band are calculated, which is equal to resizing the image to a low resolution and calculating simple texture features. Thus, each image is represented as a feature vector with $7 \times 7 \times 3 \times 2 = 294$-dimensions. The percentage of images that have multiple labels is over 22 percent. The same setting as in the previous experiments are used to randomly partition the data set into labeled/unlabeled sets according to different label rates.

The results of multilabel classification on natural scene classification task are reported in Fig. 7. TRAM is among the most accurate methods on both label set prediction and label ranking. Since this data set is relatively small, the number of labeled data set is smaller than all the other tasks. The TRAM's performances are still stable as the labeled instances decrease to small label rates.

6.7 The Influence of Parameters

As observed in previous sections, when TRAM is used with the same parameters in all the multilabel tasks, it can all achieve satisfactory classification performances as accurate as the others. In this section, we analyze the influence of parameters in TRAM.

The first exploration is about the number of nearest neighbors during the instance graph construction. The experiment is based on automatic image annotation task. We randomly partition the data set into labeled and unlabeled data with 5 percent label rate. The experiment result of TRAM is reported in Table 3, when the number of nearest neighbor during the graph construction varies from 8 to 12. The value following "$\pm$" gives the standard deviation and the best result on each metric is shown in bold face. With respect to above configurations, Table 3 shows that the number of nearest neighbors used in graph construction step does not significantly affect TRAMs performance. Therefore, all the results of TRAM shown in this paper are obtained with the parameter k set to be the moderate value of 10.

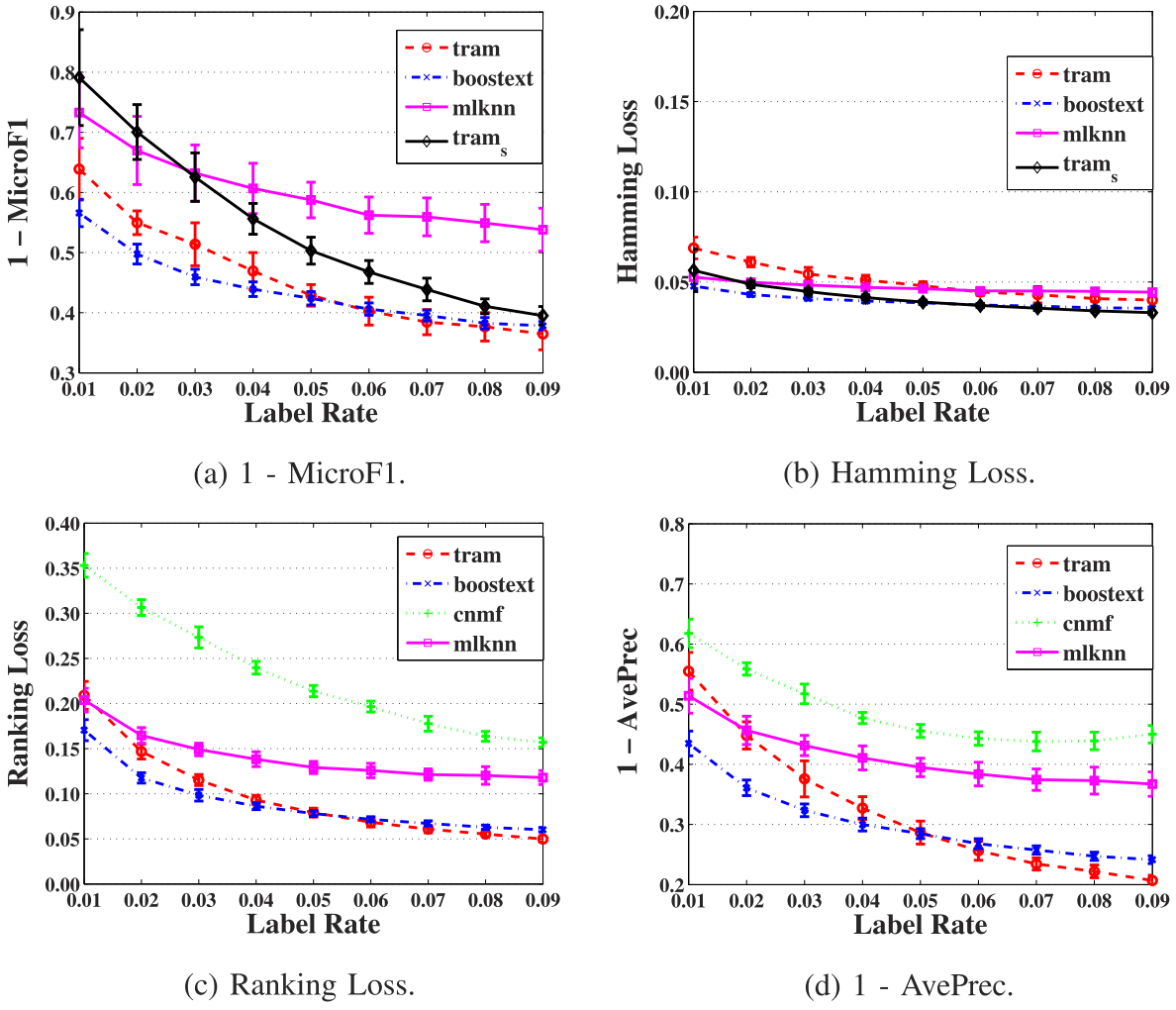


Fig. 6. Results on text categorization task under different label rates. The lower the value, the better the performance. Along with the curves, we also plot the mean $\pm$ std on each point for different random data set partitions.

Besides the number of nearest neighbor, another parameter is about the number of dimensions in the subspace used by MDDM. Note that due to the curse of the dimensionality, the similarities directly calculated based on distances between instances in the input space may be unreliable, especially when these similarities are the key parameters for the TRAM model. A simple, but often very effective, way of dealing with high-dimensional data is to reduce the number of dimensions, by finding a subspace from the input features that is most relevant to label information. Therefore, we need to utilize MDDM before the graph construction among instances. In order to verify this assumption, the results under different percentage of dimensions in the preprocess stage are reported in Fig. 8. The experiment is based on automatic image annotation task, and results on other tasks are similar to the case in this task.

Fig. 8 shows that on automatic image annotation task, the *MicroF1* and *Ranking Loss* of TRAM are significantly improved by introducing the dimensionality reduction (MDDM) before constructing the instance graph. TRAM's best performance are more likely to appear at the relatively low percentage of dimensions. Nonetheless, the number of dimensions does not have to be prespecified, which can automatically be determined by setting MDDM's threshold parameter *thr* as preserving 99.99 percent of the eigenvalues.

7 Conclusion

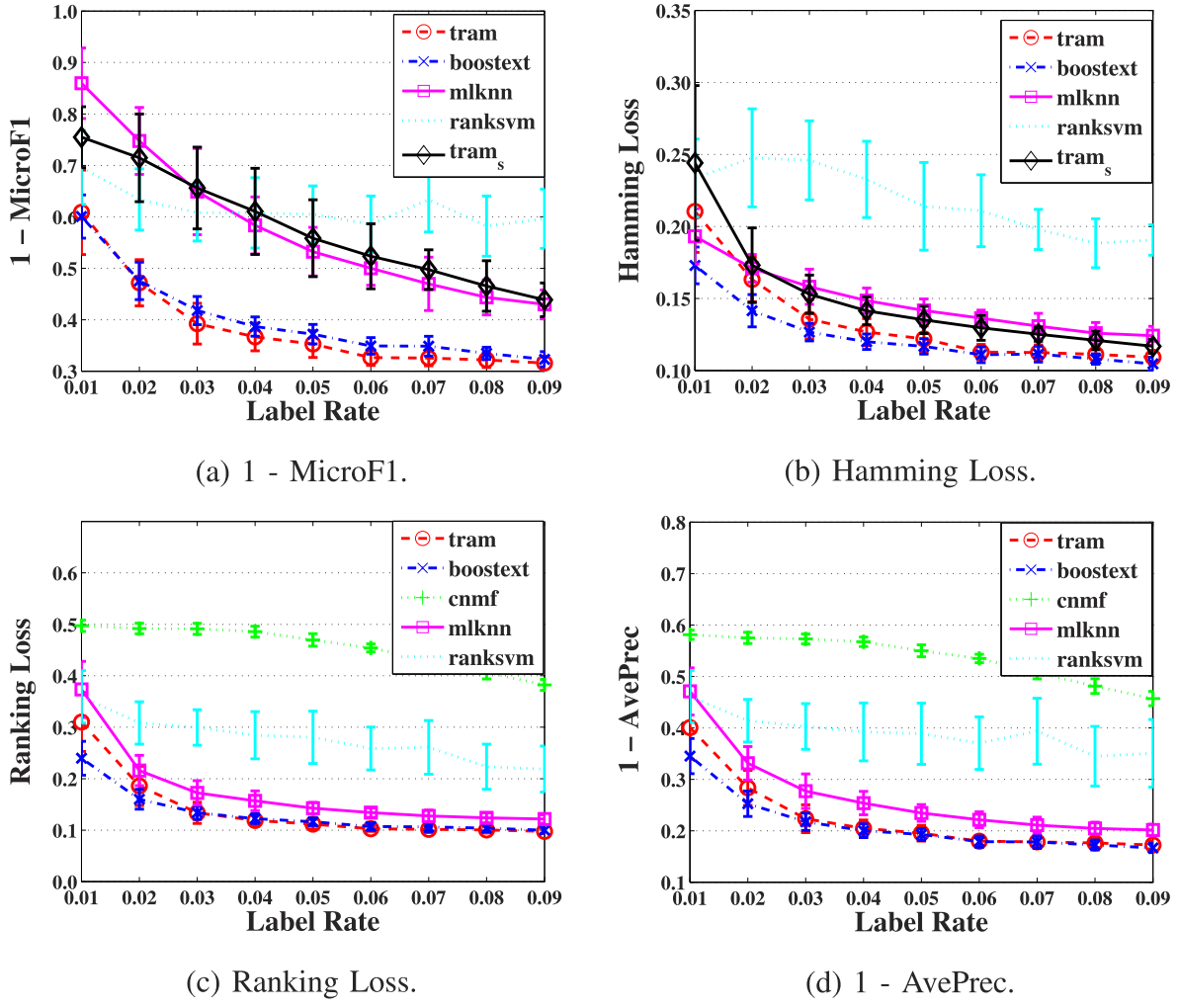
In this paper, we propose TRAM, a transductive multilabel classification method by label set propagation. At first, we formulate the task as an optimization problem which is able to exploit unlabeled data to obtain an effective model for assigning appropriate multiple labels to instances. Then, we develop an efficient algorithm which has a closed-form solution for this optimization problem. Empirical studies on a broad range of real-world tasks demonstrate that our TRAM method can effectively boost the performance of multilabel classification by using unlabeled data in addition to labeled data.

Appendix

Here, we study the properties of the linear system solutions for (5) and (6). For convenience of study, we combine the (5) with the constrains for labeled data as

$$A_{\mathcal{U}\mathcal{U}}\boldsymbol{\alpha}_{\mathcal{U}j} + A_{\mathcal{U}\mathcal{L}}\boldsymbol{\alpha}_{\mathcal{L}j} = \mathbf{0}, \tag{11}$$

$$\boldsymbol{\alpha}_{\mathcal{L}j} = \bar{\alpha}_{\mathcal{L}j}, \tag{12}$$



(a) 1 - MicroF1. (b) Hamming Loss.

(c) Ranking Loss. (d) 1 - AvePrec.

Fig. 7. Results on natural scene classification task with different label rates. The lower the value, the better the performance. Along with the curves, we also plot the mean ± std on each point for different random data set partitions.

TABLE 3
Results (Mean ± Std) of TRAM with Different Number of Nearest Neighbors Considered in the Instance Graph Construction Step on Automatic Image Annotation Task ("↓ " Indicates "the Smaller the Better," and "↑ " Indicates "the Larger the Better")

Evaluation	Number of Nearest Neighbors Considered				
Criterion	k=8	k=9	k=10	k=11	k=12
MicroF1 ↑	0.2075±0.0203	**0.2077±0.0215**	0.2066±0.0256	0.2049±0.0219	0.2031±0.0286
Hamming Loss ($\times 10^{-1}$)↓	**0.7860±0.0200**	**0.7860±0.0210**	0.787±0.025	0.788±0.021	0.791±0.028
Ranking Loss ↓	**0.2590±0.0080**	**0.2590±0.0080**	0.2601±0.0058	0.2604±0.0079	0.2605±0.0061
Average Precision ↑	**0.3240±0.0138**	0.3239±0.0146	0.3216±0.0206	0.3217±0.0141	0.3184±0.0225

which is equivalent to

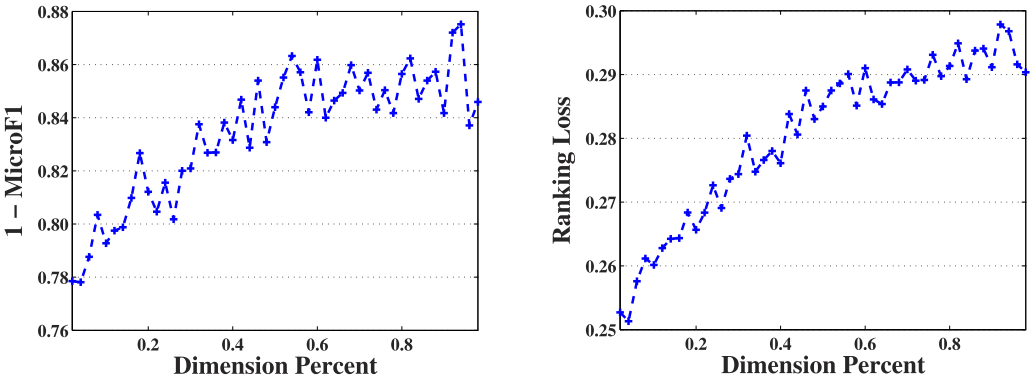
$$\tilde{A}\boldsymbol{\alpha}_{(j)} = \boldsymbol{b}_{(j)}, \quad j = 1, \ldots, m, \tag{13}$$

where

$$\tilde{A} = \begin{bmatrix} A_{\mathcal{U}\mathcal{U}} & A_{\mathcal{U}\mathcal{L}} \\ 0 & I \end{bmatrix} \text{ and } \boldsymbol{b}_{(j)} = \begin{bmatrix} 0 \\ \bar{\boldsymbol{\alpha}}_{\mathcal{L}j} \end{bmatrix}$$

Then, we show that the solution of $\tilde{A}\boldsymbol{\alpha}_{(j)} = \boldsymbol{b}_{(j)}$ automatically satisfies the bilateral constrains $\mathbf{0} \leq \boldsymbol{\alpha}_{(j)} \leq \mathbf{1}$.

Instance i and z are connected by an edge if and only if they are a neighbor of each other, and W_{iz} and W_{zi} are both positive. Let $\alpha = (\alpha_i)$ be a discrete function defined on $\mathcal{U} \bigcup \mathcal{L}$, then the (*strong*) *discrete maximum principle* says that α can only attain its maximum in $\mathcal{L}$, unless α is constant in $\mathcal{U} \bigcup \mathcal{L}$. It is similar for the minimum principle.



(a) 1 - MicroF1. (b) Ranking Loss.

Fig. 8. Performances of TRAM with different percentages of dimensions in MDDM step on automatic image annotation task.

If there are more than one connected components in $\mathcal{U}$, we can apply the principle to each component independently. We also assume that each point in $\mathcal{L}$ is a neighbor of some instance in $\mathcal{U}$.

Theorem 1. *The solution to $\tilde{A}\boldsymbol{\alpha} = \boldsymbol{b}$ satisfies the discrete maximum principle.*

Proof. Suppose that the maximum of α can be attained at an interior point $i_0 \in \mathcal{U}$. Then, the i_0th equation of (13) is $(\tilde{A}\boldsymbol{\alpha})_{i_0} = 0$ since $b_{i_0} = 0$. Notice that the i_0th row of $\tilde{A}$ is the same as the i_0th row of $A = I - W$. Therefore,

$$(\tilde{A}\boldsymbol{\alpha})_{i_0} = \alpha_{i_0} - \sum_{z \in \mathcal{N}_{i_0}} W_{i_0 z}\alpha_z = 0,$$

or

$$\alpha_{i_0} = \sum_{z \in \mathcal{N}_{i_0}} W_{i_0 z}\alpha_z.$$

Note that $W_{i_0 z} > 0$ for $z \in \mathcal{N}_{i_0}$ and $\sum_{z \in \mathcal{N}_{i_0}} W_{i_0 z} = 1$, which means the maximum value α_{i_0} equals a weighted average of $\{\alpha_z : z \in \mathcal{N}_{i_0}\}$, thus for all $z \in \mathcal{N}_{i_0}$, α_z is also the maximum. Similarly, since the domain $\mathcal{U}$ is connected, we can conclude that the values of α in $\mathcal{U}$ and the neighbor of $\mathcal{U}$ which covers $\mathcal{L}$ are all maximum. This shows that if α has an interior maximum, then α is constant in $\mathcal{U} \bigcup \mathcal{L}$. □

Corollary 1. *The solution to $\tilde{A}\boldsymbol{\alpha} = \boldsymbol{b}$ satisfies the the bilateral constraints $\mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{1}$, if $\{\alpha_i = 0 : i \in \mathcal{L}\}$ and $\{\alpha_i = 1 : i \in \mathcal{L}\}$ are nonempty sets.*

Proof. According to maximum principle, $\alpha_z \leq \max_{i \in \mathcal{L}} \alpha_i = 1$ for all $z \in \mathcal{U}$. Similarly, we have $\alpha \geq \min_{i \in \mathcal{L}} \alpha_i = 0$. Therefore, $0 \leq \alpha_z \leq 1$ for all $z \in \mathcal{U}$. □

ACKNOWLEDGMENTS

The authors wish to thank the editor and anonymous reviewers for their helpful comments and suggestions, and Yu-Feng Li, Jieping Ye, Yang Yu, De-Chuan Zhan, and Yin Zhang for reading a draft of the paper. This work was supported by the National Fundamental Research Program of China (2010CB327900), the National Science Foundation of China (61073097), the Jiangsu Science Foundation (BK2011566), the Jiangsu 333 High-Level Talent Cultivation Program, the Hong Kong Baptist University Faculty Research Grants and the Hong Kong Research Grant Council (201508).

REFERENCES

[1] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold Regularization: A Geometric Framework for Learning from Examples," *J. Machine Learning Research*, vol. 7, pp. 2399-2434, 2006.
[2] N. Biggs, *Algebraic Graph Theory*. Cambridge Univ. Press, 1974.
[3] A. Blum and S. Chawla, "Learning from Labeled and Unlabeled Data Using Graph Mincuts," *Proc. 18th Int'l Conf. Machine Learning*, pp. 19-26, 2001.
[4] M.R. Boutell, J. Luo, X. Shen, and C.M. Brown, "Learning Multi-Label Scene Classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757-1771, 2004.
[5] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge Univ. Press, 2004.
[6] G. Carneiro, A. Chan, P. Moreno, and N. Vasconcelos, "Supervised Learning of Semantic Classes for Image Annotation and Retrieval," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 394-410, Mar. 2007.
[7] *Semi-Supervised Learning*. O. Chapelle, B. Schölkopf, and A. Zien, eds., MIT Press, 2006.
[8] F.D. Comité, R. Gilleron, and M. Tommasi, "Learning Multi-Label Altenating Decision Tree From Texts and Data," *Proc. Third Int'l Conf. Machine Learning and Data Mining in Pattern Recognition*, pp. 35-49, 2003.
[9] P. Duygulu, K. Barnard, N. De Freitas, and D. Forsyth, "Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary," *Proc. Seventh European Conf. Computer Vision*, pp. 97-112, 2002.
[10] A. Elisseeff and J. Weston, "A Kernel Method for Multi-Labelled Classification," *Advances in Neural Information Processing Systems 14*, T.G. Dietterich, S. Becker and Z. Ghahramani, eds., pp. 681-687, MIT Press, 2002.
[11] 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, pp. 119-139, 1997.
[12] I. Vlahavas and G. Tsoumakas, "Random K-Labelsets: An Ensemble Method for Multi-Label Classification," *Proc. 18th European Conf. Machine Learning*, pp. 406-417, 2007.
[13] S. Gao, W. Wu, C.H. Lee, and T.-S. Chua, "A MFoM Learning Approach to Robust Multiclass Multi-Label Text Categorization," *Proc. 21th Int'l Conf. Machine Learning*, pp. 329-336, 2004.
[14] N. Ghamrawi and A. McCallum, "Collective Multi-Label Classification," *Proc. 14th Int'l Conf. Information and Knowledge Management*, pp. 195-200, 2005.
[15] S. Godbole and S. Sarawagi, "Discriminative Methods for Multi-Labeled Classification," *Proc. Eight Pacific-Asia Conf. Knowledge Discovery and Data Mining*, pp. 22-30, 2004.
[16] W. Hackbusch, "Iterative Solution of Large Sparse Systems of Equations," *Math. of Computation*, vol. 64, no. 212, pp. 1759-1761, 1995.

[17] T. Joachims, "Transductive Inference for Text Classification Using Support Vector Machines," *Proc. 16th Int'l Conf. Machine Learning,* pp. 200-209, 1999.
[18] T. Joachims, "Transductive Learning via Spectral Graph Partitioning," *Proc. 20th Int'l Conf. Machine Learning,* pp. 290-297, 2003.
[19] F. Kang, R. Jin, and R. Sukthankar, "Correlated Label Propagation with Application to Multi-Label Learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1719-1726, 2006.
[20] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda, "Maximal Margin Labeling for Multi-Topic Text Categorization," *Advances in Neural Information Processing Systems 17,* L.K. Saul, Y. Weiss, and L. Bottou, eds., pp. 649-656, MIT Press, 2005.
[21] D.D. Lewis, Y. Yang, T. Rose, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research," *J. Machine Learning Research,* vol. 5, pp. 361-397, 2004.
[22] Y. Liu, R. Jin, and L. Yang, "Semi-Supervised Multi-Label Learning by Constrained Non-Negative Matrix Factorization," *Proc. 21st Nat'l Conf. Artificial Intelligence,* pp. 421-426, 2006.
[23] P. Matstoms, "Sparse QR Factorization in MATLAB," *ACM Trans. Math. Software,* vol. 20, no. 1, pp. 136-159, 1994.
[24] A. McCallum, "Multi-Label Text Classification with a Mixture Model Trained by EM," *Proc. Working Notes Am. Assoc. Artificial Intelligence Workshop Text Learning (AAAI '99),* 1999.
[25] M. Ng, G. Qiu, and A. Yip, "A Study of Interactive Multiple Class Image Segmentation Problems," Technical Report 07-51, UCLA CAM, 2007.
[26] P. Pavlidis, J. Weston, J. Cai, and W.N. Grundy, "Combining Microarray Expression Data and Phylogenetic Profiles to Learn Functional Categories Using Support Vector Machines," *Proc. Fifth Int'l Conf. Computational Biology,* pp. 242-248, 2001.
[27] G.J. Qi, X.S. Hua, Y. Rui, J. Tang, T. Mei, and H.J. Zhang, "Correlative Multi-Label Video Annotation," *Proc. 15th Int'l Conf. Multimedia,* pp. 17-26, 2007.
[28] R.E. Schapire and Y. Singer, "BoosTexter: A Boosting-Based System for Text Categorization," *Machine Learning,* vol. 39, nos. 2/3, pp. 135-168, 2000.
[29] L. Sun, S.-W. Ji, and J.-P. Ye, "Hypergraph Spectral Learning for Multi-Label Classification," *Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 668-676, 2008.
[30] G. Tsoumakas and I. Katakis, "Multi-Label Classification: An Overview," *Int'l J. Data Warehousing and Mining,* vol. 3, no. 3, pp. 1-13, 2007.
[31] N. Ueda and K. Saito, "Parametric Mixture Models for Multi-Labeled Text," *Advances in Neural Information Processing Systems 15,* S. Becker, S. Thrun, and K. Obermayer, eds., pp. 721-728, MIT Press, 2003.
[32] V.N. Vapnik, *Statistical Learning Theory.* Wiley, 1998.
[33] R. Weber, H.-J. Schek, and S. Blott, "A Quantitative Analysis and Performance Study for Similaritysearch Methods in High-Dimensional Space," *Proc. 24th Int'l Conf. Very Large Data Bases,* pp. 194-205, 1998.
[34] M.-L. Zhang and Z.-H. Zhou, "Multi-Label Neural Networks with Applications to Functional Genomics and Text Categorization," *IEEE Trans. Knowledge and Data Eng.,* vol. 18, no. 10, pp. 1479-1493, Oct. 2006.
[35] M.-L. Zhang and Z.-H. Zhou, "ML-kNN: A Lazy Learning Approach to Multi-Label Learning," *Pattern Recognition,* vol. 40, no. 7, pp. 2038-2048, 2007.
[36] Y. Zhang and Z.-H. Zhou, "Multilabel Dimensionality Reduction via Dependence Maximization," *ACM Trans. Knowledge Discovery from Data,* vol. 4, no. 3, article 14, 2010.
[37] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Scholkopf, "Learning with Local and Global Consistency," *Advances in Neural Information Processing Systems 15,* S. Becker, S. Thrun, and K. Obermayer, eds., pp. 321-328, MIT Press, 2003.
[38] Z.-H. Zhou and M. Li, "Semi-Supervised Learning by Disagreement," *Knowledge and Information Systems,* vol. 24, no. 3, pp. 415-439, 2010.
[39] Z.-H. Zhou and M.-L. Zhang, "Multi-Instance Multi-Label Learning with Application to Scene Classification," *Advances in Neural Information Processing Systems 18,* Y. Weiss, B. Schölkopf, and J. Platt, eds., pp. 1609-1616, MIT Press, 2006.
[40] Z.-H. Zhou and M.-L. Zhang, and S.-J. Huang, and Y.-F. Li, "Multi-Instance Multi-Label Learning," *Artificial Intelligence,* vol. 176, no. 1, pp. 2291-2320, 2012.
[41] S. Zhu, X. Ji, W. Xu, and Y. Gong, "Multi-Labelled Classification Using Maximum Entropy Method," *Proc. 28th Int'l Conf. Research and Development in Information Retrieval,* pp. 274-281, 2005.
[42] X. Zhu, "Semi-Supervised Learning Literature Survey," Technical Report 1530, Department of Computer Sciences, Univ. of Wisconsin at Madison, 2006.

Xiangnan Kong received the bachelor and master's degrees in computer science from Nanjing University, China, in 2006 and 2009, respectively. He is pursuing a PhD degree in the Department of Computer Science at the University of Illinois at Chicago. His main research areas are data mining and machine learning, especially multilabel learning and graph mining.

Michael K. Ng received the BSc and MPhil degrees from the University of Hong Kong in 1990 and 1992, respectively, and the PhD degree from Chinese University of Hong Kong in 1995. He is a professor in the Department of Mathematics at the Hong Kong Baptist University. From 1995-1997, he was a research fellow of Computer Sciences Laboratory at Australian National University, and from 1997-2005, as an assistant/associate professor at the University of Hong Kong before joining Hong Kong Baptist University. As an applied mathematician, his research interests include bioinformatics, data mining, operations research, and scientific computing. He has published and edited five books, published more than 200 journal papers. Currently, he serves on several editorial boards of international journals.

Zhi-Hua Zhou (S'00-M'01-SM'06-F'13) received the BSc, MSc, and PhD degrees in computer science from Nanjing University, China, in 1996, 1998, and 2000, respectively, all with the highest honors. He joined the Department of Computer Science and Technology at Nanjing University as an assistant professor in 2001, and currently, he is a professor and director of the LAMDA group. He serves/ed as an associate editor-in-chief of the *Chinese Science Bulletin,* and an associate editor of the *IEEE Transactions on Knowledge and Data Engineering* and the *ACM Transaction on Intelligent Systems and Technology,* and on the editorial boards of various other journals. He is the founder and Steering Committee chair of ACML and Steering Committee member of PAKDD and PRICAI. He serves/ed as general chair/co-chair of ACML '12 and ADMA '12, program chair/co-chair for PAKDD '07, PRICAI '08, ACML '09, and SDM '13, and workshop chair of KDD '12, program vice chair or area chair of various conferences, and chaired various domestic conferences in China. He is the chair of the Machine Learning Technical Committee of the Chinese Association of Artificial Intelligence, chair of the Artificial Intelligence and Pattern Recognition Technical Committee of the China Computer Federation, vice chair of the Data Mining Technical Committee of IEEE Computational Intelligence Society and the chair of the IEEE Computer Society Nanjing Chapter. His research interests include artificial intelligence, machine learning, data mining, pattern recognition, and multimedia information retrieval. In these areas, he has published more than 90 papers in leading International journals or conference proceedings, and holds 12 patents. He has won various awards/honors including the National Science and Technology Award for Young Scholars of China, the Fok Ying Tung Young Professorship First-Grade Award, the Microsoft Young Professorship Award, and nine International Journals/Conferences Paper Awards or Competition Awards. He is a fellow of the IAPR, the IEEE, and the IET/IEE.