Efficient Multi-label Classification with Hypergraph Regularization

Gang Chen, Jianwen Zhang, Fei Wang, Changshui Zhang
State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology(TNList)
Department of Automation, Tsinghua University, Beijing 100084, China
{g-c05, jw-zhang06, feiwang03}@mails.thu.edu.cn, zcs@mail.thu.edu.cn

Yuli Gao
Hewlett-Packard Laboratories
1501 Page Mill Road, MS 1203
Palo Alto, CA 94304 USA
yuli.gao@hp.com

Abstract

Many computer vision applications, such as image classification and video indexing, are usually multi-label classification problems in which an instance can be assigned to more than one category. In this paper, we present a novel multi-label classification approach with hypergraph regularization that addresses the correlations among different categories. First, a hypergraph is constructed to capture the correlations among different categories, in which each vertex represents one training instance and each hyperedge for one category contains all the instances belonging to the same category. Then, an improved SVM like learning system incorporating the hypergraph regularization, called Rank-HLapSVM, is proposed to handle the multi-label classification problems. We find that the corresponding optimization problem can be efficiently solved by the dual coordinate descent method. Many promising experimental results on the real datasets including ImageCLEF and MediaMill demonstrate the effectiveness and efficiency of the proposed algorithm.

1. Introduction

In computer vision, many applications such as image classification [6] and video indexing [22], are usually multi-label classification problems. Multi-label classification refers to the classification problems where an instance can be associated with more than one category. It is different from multi-class classification, in which an instance can only assigned a single category. Consider an example of multi-label classification in image classification, and an image can be annotated as 'Road', 'Car', 'Tree', 'Human' and 'Building' (See Fig 1), where these different terms represent different semantic concepts. Besides in image classification and video indexing, such type of problems also arise in many other practical applications, such as text categorization [30] and protein function prediction[12].

Figure 1. An example image associated with a set of categories including 'Road', 'Car', 'Tree', 'Human' and 'Building'.

The most simple method for multi-label classification is to divide it into a set of independent binary classification problems, one for each category. The final labels for each instance can be determined by aggregating the classification results from all the binary classifiers. Obviously, under this framework many state-of-the-art binary classifying techniques can be easily adopted to handle the multi-label classification problem [8, 15, 29]. However, just as pointed out by [19, 31], this approach does not take into account the underlying mutual correlations among different categories, which usually do exist and could even have significant influences to the prediction performance. For example, in Fig. 1, the two categories 'Car' and 'Road' often emerge in the same image, that means 'Car' and 'Road' have a strong pos-

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

itive relationship, *i.e*. the two categories are not independent of each other. If we ignore the correlations among categories and directly apply the above method, the classification performance might be poor.

To address the challenge how to model the category correlations, some novel multi-label classification algorithms have been proposed, a kind of which is label ranking [21, 12, 10, 11]. These approaches take ranking-based strategies that learn a ranking function of category labels from the labeled instances and apply it to obtain a real-valued score to each instance-category pair, then classify each instance by choosing all the categories with the scores above the given threshold. Although label ranking approaches provide a novel way to handle the multi-label learning problem, they generally do not explicitly exploit the correlations among data categories. In addition, there are other studies toward multi-label classification modeling the correlations among categories [19, 25, 13, 31, 16, 17, 9, 30, 23].

Recently, researches on learning with structured outputs where the prediction variables are interdependent in complex ways have drawn considerable interests. Multi-label classification is a kind of typical learning problems with structured outputs and can be solved by these learning models [20], whose main idea is to learn a discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ over input-output joint feature pairs. Although such a joint feature map effectively exploits the internal structure information, it results in quite high computational expenses on learning and inference [2, 24].

In this paper, we provide a novel supervised multi-label classification approach with hypergraph regularization that address the correlations among different categories. The two main contributions of our paper are

- Incorporate the hypergraph regularization into RankSVM and offer a general label ranking algorithm for multi-label classification, *i.e*. Rank-HLapSVM.
- Apply the dual coordinate descent method to efficiently solve the corresponding optimization problem, that is much faster than Frank and Wolfe's method [12].

Concretely, we first construct a hypergraph where each vertex represents a training instance, and each edge called hyperedge for one category includes all the instances belonging to the same category. Thus the correlations among categories can be effectively captured via hyperedge links. Intuitively, the similar instances tend to have the similar labels. Based on this assumption, an improved SVM like approach incorporating the hypergraph Laplacian regularizer is proposed to give a label ranking algorithm for the multi-label problems. We find that the corresponding optimization problem can be efficiently solved by the dual coordinate descent method. Finally, a predictor of the size of label set is suggested to determine the label number for one instance.

The rest of the paper is organized as follows. Section 2 gives some background knowledge about hypergraph. In Section 3, we elaborate our novel supervised multi-label classification algorithm. The data and experiment results are presented in Section 4, followed by our conclusions in Section 5.

2. Background

In this section, we introduce some background knowledge about hypergraph.

A hypergraph is a generalization of a graph in which edges, called hyperedges, may connect any number of vertices [5]. Formally, a hypergraph G is a pair (V, E) where V is a set of vertices and E is a set of hyperedges. The degree of a hyperedge e associated with weight $w(e)$, denoted as $\delta(e)$, is the number of vertices in e. In particular, for the traditional graphs or "2-graphs", $\delta(e) = 2$ .The degree $d(v)$ of a vertex v is $d(v) = \sum_{v \in e, e \in E} w(e)$. The vextex-edge incidence matrix $H \in \mathbb{R}^{|V| \times |E|}$ is defined as: $h(v, e) = 1$ if $v \in e$ and 0 otherwise. So we have

$$d(v) = \sum_{e \in E} w(e) h(v, e) \tag{1}$$

$$\delta(e) = \sum_{v \in V} h(v, e) \tag{2}$$

Let D_e and D_v be the diagonal matrices consisting of $\delta(e)$ and $d(v)$, respectively. Denote W as the diagonal matrix of edge weights $w(e)$.

The graph Laplacian is the discrete analog of the Laplace-Beltrami operator on compact Riemannian manifolds [3]. It has been widely used in unsupervised (e.g. spectral clustering [27])and semi-supervised learning (e.g. [28, 32]) problems. Next, we will briefly describe one of the commonly used algorithms constructing the hypergraph Laplacian, which is called the clique expansion algorithm [1, 23].

The clique expansion algorithm constructs a traditional 2-graph $G_c = (V_c, E_c)$ from the original hypergraph $G = (V, E)$ and regards the Laplacian of G_c as that of G. Suppose $V_c = V$ and $E_c = \{(u, v) | u, v \in e, e \in E\}$. The edge weight $w_c(u, v)$ of G_c is defined by

$$w_c(u, v) = \sum_{u, v \in e, e \in E} w(e) \tag{3}$$

The above definition means that the similarity matrix of G_c can be expressed by

$$W_c = HWH^T \tag{4}$$

Let D_c be the diagonal matrix where $D_c(u, u) = \sum_v w_c(u, v)$. Then the combinatorial Laplacian of G_c is given by

$$L_c = D_c - W_c = D_c - HWH^T \tag{5}$$

and the normalized Laplacian is given by

$$\begin{aligned}\mathcal{L}_c &= I - D_c^{-1/2} W_c D_c^{-1/2} \\ &= I - D_c^{-1/2} H W H^T D_c^{-1/2} \end{aligned} \quad (6)$$

From Eq. (5) and (6), we have

$$\mathcal{L}_c = D_c^{-1/2} L_c D_c^{-1/2} \quad (7)$$

3. Multi-Label Classification with Hypergraph Regularization

We first give some notations that will be used throughout the paper. In a typical multi-label scenario, there are n training samples $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)$. We assume that each instance $\boldsymbol{x}_i$ is drawn from a domain $\mathcal{X} \subseteq \mathbb{R}^m$ and its label y_i is a subset of the output label set $\mathcal{Y} = \{1, \cdots, k\}$. For example, if x_i belongs to category $1, 3, 4$, $y_i = \{1, 3, 4\}$. Set $X = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)^T$.

Our basic strategy is to first take multi-label classification as a label ranking problem, then predict the label number of each instance and finally obtain the final labels of each instance. Label ranking is the task of inferring a total order over a predefined set of labels for each given instance [11]. Generally, for each category, we define a linear function $f_i(x) = \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i \ \ (i = 1, \cdots, k)$, where $\langle \cdot, \cdot \rangle$ is the inner product. One often deal with the bias term b_i by appending each instance with an additional dimension

$$x^T \leftarrow [\boldsymbol{x}^T, 1], \ \ \boldsymbol{w}_i^T \leftarrow [\boldsymbol{w}_i^T, b_i] \quad (8)$$

then the linear function becomes $f_i(\boldsymbol{x}) = \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle$. The goal of label ranking is to order $\{f_i(\boldsymbol{x}), i = 1, \cdots, k\}$ for each instance $\boldsymbol{x}$ according to some predefined empirical loss function and complexity measures. Elisseeff and Weston [12] applied the large margin idea to multi-label classification and presented a SVM like ranking system, called Rank-SVM, as follows

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2 + C \sum_{i=1}^{n} \frac{1}{|y_i||\overline{y}_i|} \sum_{(p,q)\in y_i \times \overline{y}_i} \xi_{ipq} \\ \text{s.t.} \quad & \langle \boldsymbol{w}_p - \boldsymbol{w}_q, x_i \rangle \geq 1 - \xi_{ipq}, (p,q) \in y_i \times \overline{y}_i \\ & \xi_{ipq} \geq 0 \end{aligned} \quad (9)$$

where C is the nonnegative penalty coefficient that reflects the trade-off between the empirical loss and model complexity, $\overline{y}_i$ is the complementary set of y_i in $\mathcal{Y}$, $|y_i|$ is the cardinality of the set y_i, i.e. the number of elements of the set y_i, and $\xi_{ipq} (i = 1, \cdots, n; (p,q) \in y_i \times \overline{y}_i)$ are slack variables. The margin term $\sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2$ controls the model complexity and improves the model generalization performance. Although this approach performs better than Binary-SVM in many cases, it still does not model the category correlations clearly. Next, we will introduce how to construct a hypergraph to exploit the category correlations and how to incorporate the hypergraph regularization into Eq. (9).

3.1. Basic Framework

To model the correlations among different categories effectively, a hypergraph is built where each vertex corresponds to one training instance and each hyperedge for one category includes all the training instances relevant to the same category. Here, we apply the clique expansion algorithm to construct the similarity matrix of the hypergraph. It means that the similarity of two instances is proportional to the sum of the weights of their common categories, which captures the higher order relations among different categories. This kind of hypergraph structure was used in the feature extraction by spectral learning [23]. However, we consider how to apply the relation information encoded in the hypergraph to directly design the multi-label classification model. Intuitively, two instances tend to have large overlap in their assigned categories if they share high similarity in the hypergraph. Formally, this smoothness assumption can be expressed using the hypergraph Laplacian regularizer, trace$(\widehat{F}^T L \widehat{F})$, so we introduce it into Eq. (9) and have

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2 + \frac{1}{2}\lambda \text{trace}(\widehat{F}^T L \widehat{F}) + \\ & C \sum_{i=1}^{n} \frac{1}{|y_i||\overline{y}_i|} \sum_{(p,q)\in y_i \times \overline{y}_i} \xi_{ipq} \\ \text{s.t.} \quad & \langle \boldsymbol{w}_p - \boldsymbol{w}_q, x_i \rangle \geq 1 - \xi_{ipq}, (p,q) \in y_i \times \overline{y}_i \\ & \xi_{ipq} \geq 0 \end{aligned} \quad (10)$$

where $\widehat{F} = (F(\boldsymbol{x}_1), \cdots, F(\boldsymbol{x}_n))^T \in \mathbb{R}^{n \times k}$, $F(\boldsymbol{x}_i) = (f_1(\boldsymbol{x}_i), \cdots, f_k(\boldsymbol{x}_i))^T$ predicts the label matrix for training data, L is the $n \times n$ hypergraph Laplacian and λ is the nonnegative constant that controls the model complexity in the intrinsic geometry of input distribution. Our general framework for multi-label classification, called Rank-HLapSVM, has a close relation with the manifold regularization algorithm [4], and more discussions can be found in Section 3.5.

3.2. The Dual Problem

Eq. (10) is a linearly constrained quadratic convex optimization problem. First, we introduce a dual set of variables, one for each constraint, i.e $\alpha_{ipq} \geq 0$ for $\langle \boldsymbol{w}_p - \boldsymbol{w}_q, \boldsymbol{x}_i \rangle - 1 + \xi_{ipq} \geq 0$ and η_{ipq} for $\xi_{ipq} \geq 0$. Then the

Lagrangian of Eq. (10) can be given by

$$\begin{aligned}\mathrm{Lag}(\boldsymbol{x},\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\eta}) &= \frac{1}{2}\sum_{i=1}^{k}\|\boldsymbol{w}_i\|^2+\frac{1}{2}\lambda\mathrm{trace}(\widehat{F}^T L\widehat{F})\\ &+C\sum_{i=1}^{n}\frac{1}{|y_i||\overline{y}_i|}\sum_{(p,q)\in y_i\times\overline{y}_i}\xi_{ipq}\\ &-\sum_{i=1}^{n}\sum_{(p,q)\in y_i\times\overline{y}_i}\alpha_{ipq}(\langle\boldsymbol{x}_p-\boldsymbol{w}_q,\boldsymbol{x}_i\rangle-1\\ &+\xi_{ipq})-\sum_{i=1}^{n}\sum_{(p,q)\in y_i\times\overline{y}_i}\eta_{ipq}\xi_{ipq} \qquad (11)\end{aligned}$$

Here, we choose the combinatorial Laplacian that yields $\widehat{F}^T L\widehat{F}=W_k X^T LXW_k^T$ with $W_k=(\boldsymbol{w}_1,\cdots,\boldsymbol{w}_k)^T$

To find the minimum over the primal variables we require

$$\begin{aligned}\frac{\partial\mathrm{Lag}}{\partial\boldsymbol{w}_p} &= \boldsymbol{w}_p+\lambda X^T LX\boldsymbol{w}_p-\sum_{i=1}^{n}\sum_{(j,q)\in y_i\times\overline{y}_i}t^p_{ijq}\alpha_{ijq}\boldsymbol{x}_i\\ &= 0 \qquad (12)\end{aligned}$$

where

$$t^p_{ijq}=\begin{cases}1 & j=p\\ -1 & q=p\\ 0 & \text{if } j\neq p \text{ and } q\neq p\end{cases} \qquad (13)$$

Similarly, let

$$\frac{\partial\mathrm{Lag}}{\partial\xi_{ipg}}=\frac{C}{|y_i||\overline{y}_i|}-\alpha_{ipq}-\eta_{ipq}=0 \qquad (14)$$

To abbreviate the notations, we define

$$\beta_{pi}=\sum_{(j,q)\in y_i\times\overline{y}_i}t^p_{ijq}\alpha_{ijq} \qquad (15)$$

that yields

$$\boldsymbol{w}_p=(I+\lambda X^T LX)^{-1}\sum_{i=1}^{n}\beta_{pi}\boldsymbol{x}_i \qquad (16)$$

where I is the $(m+1)\times(m+1)$ unit matrix.

So the dual problem of Eq. (10) can be expressed by

$$\begin{aligned}\min g(\boldsymbol{\alpha}) &= \frac{1}{2}\sum_{p=1}^{k}\sum_{h,i=1}^{n}\beta_{ph}\beta_{pi}\boldsymbol{x}_h^T(I+\lambda X^T LX)^{-1}\boldsymbol{x}_i\\ &-\sum_{i=1}^{n}\sum_{(p,q)\in y_i\times\overline{y}_i}\alpha_{ipq}\\ \text{s.t.}\quad & 0\le\alpha_{ipq}\le\frac{C}{|y_i||\overline{y}_i|} \qquad (17)\end{aligned}$$

Note the box constraints are derived from Eq. (14) by using the fact that $\eta_{ipq}\ge 0$.

Once the variables α_{ipq} are solved, w_p can be calculated by Eq. (16). Compared with the primal optimization problem, the dual decreases in the number of variables by $k(m+1)$ and includes more simple box constraints. In the following section, we will describe an efficient optimization algorithm to solve the dual problem.

3.3. Efficient Optimization Algorithm using Dual Coordinate Descent

In the practical multi-label problems, the amount of instances is usually large (maybe a few thousand even tens of thousands), which makes it computationally expensive to solve Eq. (17) by off the shelf QP solvers. Hsieh *et al*. [14] recently proposed a dual coordinate descent method for linear SVM that is evidently superior to other state of the art solvers. Actually, Eq. (17) can be efficiently solved by the coordinate descent method. Next, we will describe the coordinate descent method for Eq. (17).

Coordinate descent is a popular optimization technique which updates one variable ar a time by minimizing a single variable subproblem. If the subproblem can be efficiently solved, then it can be a competitive optimization method. For the dual problem (17), the coordinate descent method picks one variable α_{ipq} at a time and solves the following single variable subproblem keeping all other variables fixed

$$\begin{aligned}\min_d\quad & g(\boldsymbol{\alpha}+d\boldsymbol{e}_{ipq})\\ \text{s.t.}\quad & 0\le\alpha_{ipq}+d\le\frac{C}{|y_i||\overline{y}_i|} \qquad (18)\end{aligned}$$

where $\boldsymbol{e}_{ipq}=(0,\cdots,0,1,0,\cdots,0)^T$. The object function $g(\boldsymbol{\alpha}+d\boldsymbol{e}_{ipq})$ of Eq. (18) is a simple quadratic function of d

$$g(\boldsymbol{\alpha}+d\boldsymbol{e}_{ipq})=A_{ii}d^2+\nabla_{ipq}g(\boldsymbol{\alpha})d+\text{constant} \qquad (19)$$

where $A_{ii}=\boldsymbol{x}_i^T(I+\lambda X^T LX)^{-1}\boldsymbol{x}_i$ and

$$\begin{aligned}\nabla_{ipq}g(\boldsymbol{\alpha}) &= \frac{\partial g(\boldsymbol{\alpha})}{\partial\alpha_{ipq}}\\ &= \sum_{j=1}^{n}(\beta_{pj}-\beta_{qj})\langle\boldsymbol{x}_j,(I+\lambda X^T LX)^{-1}\boldsymbol{x}_i\rangle-1\\ &= (\boldsymbol{w}_p-\boldsymbol{w}_q)^T\boldsymbol{x}_i-1 \qquad (20)\end{aligned}$$

It can be easily seen that Eq. (18) has an optimum at $d=0$ if and only if

$$\nabla^P_{ipq}g(\boldsymbol{\alpha})=0 \qquad (21)$$

where $\nabla^P g(\boldsymbol{\alpha})$ is the projected gradient

$$\nabla^P_{ipq} g(\boldsymbol{\alpha}) = \begin{cases} \nabla_{ipq} g(\boldsymbol{\alpha}) & \text{if } 0 < \alpha_{ipq} < \frac{C}{|y_i||\overline{y}_i|} \\ \min(0, \nabla_{ipq} g(\boldsymbol{\alpha})) & \text{if } \alpha_{ipq} = 0 \\ \max(0, \nabla_{ipq} g(\boldsymbol{\alpha})) & \text{if } \alpha_{ipq} = \frac{C}{|y_i||\overline{y}_i|} \end{cases} \tag{22}$$

If Eq. (21) holds, we do not need to update α_{ipq} and directly move to next variable. Otherwise, the optimal solution of Eq. (18) is

$$\alpha^*_{ipq} = \min(\max(\alpha_{ipq} - \frac{\nabla_{ipq} g(\boldsymbol{\alpha})}{2A_{ii}}, 0), \frac{C}{|y_i||\overline{y}_i|}) \tag{23}$$

This means the subproblem can be solved analytically that ensures the efficiency of the coordinate descent method. Here, we need to calculate A_{ii} and $\nabla_{ipq} g(\boldsymbol{\alpha})$. First, A_{ii} can be precomputed and stored in the memory. Second, to evaluate $\nabla_{ipq} g(\boldsymbol{\alpha})$ using Eq. (20), we need to maintain $\boldsymbol{w}$ by

$$\boldsymbol{w}_p \leftarrow \boldsymbol{w}_p + (\alpha^*_{ipq} - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i \tag{24}$$

$$\boldsymbol{w}_q \leftarrow \boldsymbol{w}_q - (\alpha^*_{ipq} - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i \tag{25}$$

where $(I+\lambda X^T L X)^{-1}\boldsymbol{x}_i$ can also precomputed and stored in the memory.

The dual coordinate descent method for linear Rank-HLapSVM is listed in Algorithm 1. Calculating $\nabla_{ipq} g(\boldsymbol{\alpha})$ takes $O(\overline{m})$ operations, where $\overline{m}$ is the average number of nonzero elements per instance. Updating $\boldsymbol{w}_p$ and $\boldsymbol{w}_q$ needs $O(\overline{m})$ operations. Thus the cost per iteration (updating $\boldsymbol{\alpha}$ one time) is $O(n_\alpha \overline{m})$. The main memory requirement is on storing $\boldsymbol{x}_i$, A_{ii} and $(I + \lambda X^T L X)^{-1}\boldsymbol{x}_i (i = 1, \cdots, n)$. Like [14], we can easily prove the following convergence theorem using techniques in [18]

Theorem 1 *$\boldsymbol{\alpha}$ generated by Algorithm 1 globally converge to an optimal solution $\boldsymbol{\alpha}^*$. The convergence rate is at least linear: there are $0 < \mu < 1$ and an iteration t_0 such that*

$$g(\boldsymbol{\alpha}^{t+1}) - g(\boldsymbol{\alpha}^*) \leq \mu(g(\boldsymbol{\alpha}^t) - g(\boldsymbol{\alpha}^*)), \forall t > t_0 \tag{26}$$

Due to the space limitation, we omit the proof. The linear convergence result is remarkable, that means Algorithm 1 can achieve an ϵ-accurate solution $\boldsymbol{\alpha}$ ($g(\boldsymbol{\alpha}) \leq g(\boldsymbol{\alpha}^*) + \epsilon$) in $O(\log(1/\epsilon))$ iterations.

In order to speed up our algorithm, we also employ two heuristic strategies [14] for Algorithm 1. The first is to randomly permute the subproblems at each outer iteration. The second is to apply the shrinking technique to reduce the size of the optimization problem without considering some bounded variables.

In addition, let $A \in \mathbb{R}^{n \times n}$ and $A_{ij} = \langle \boldsymbol{x}_i, (I + \lambda X^T L X)^{-1} \boldsymbol{x}_j \rangle$, then we have

$$\begin{aligned} A &= X(I + \lambda X^T L X)^{-1} X^T \\ &= X X^T (I + \lambda L X X^T)^{-1} \end{aligned} \tag{27}$$

Algorithm 1 A dual coordinate descent method for linear Rank-HLapSVM

Start with $\boldsymbol{\alpha} = \mathbf{0} \in \mathbb{R}^{n_\alpha}$ ($n_\alpha = \sum_{i=1}^n |y_i||\overline{y}_i|$), and the corresponding $\boldsymbol{w}_i = \mathbf{0}$ $(i = 1, \cdots, k)$

while 1 **do**

  for $i = 1, \cdots, n$ and $(j, q) \in y_i \times \overline{y}_i$ **do**

    1. $G = (\boldsymbol{w}_p - \boldsymbol{w}_q)^T \boldsymbol{x}_i - 1$

    2. $PG = \begin{cases} G & \text{if } 0 < \alpha_{ipq} < \frac{C}{|y_i||\overline{y}_i|} \\ \min(0, G) & \text{if } \alpha_{ipq} = 0 \\ \max(0, G) & \text{if } \alpha_{ipq} = \frac{C}{|y_i||\overline{y}_i|} \end{cases}$

    3. If $|PG| \neq 0$,

$$\alpha^*_{ipq} \leftarrow \min(\max(\alpha_{ipq} - \frac{G}{2A_{ii}}, 0), \frac{C}{|y_i||\overline{y}_i|})$$

$$\boldsymbol{w}_p \leftarrow \boldsymbol{w}_p + (\alpha^*_{ipq} - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i$$

$$\boldsymbol{w}_q \leftarrow \boldsymbol{w}_q - (\alpha^*_{ipq} - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i$$

  end for

  if $\|\boldsymbol{\alpha}^* - \boldsymbol{\alpha}\| < \epsilon$ **then**

    Break

  end if

  $\boldsymbol{\alpha} = \boldsymbol{\alpha}^*$

end while

where XX^T is the inner product matrix and $\{XX^T\}_{ij} = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$. Therefore, only replacing the inner products $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$ by appropriate kernels $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, our algorithm is easily extended to the nonlinear version. However, in our experiments we only use the linear algorithm.

3.4. Predicting the Size of Label Set

So far we have only provided a label ranking algorithm. To identify the final labels of data, we need to design an appropriate threshold for each instance to determine the size of its corresponding label set. Here, we adopt the strategy proposed by Elisseeff and Weston [12], which takes threshold designing as a supervised learning problem. More concretely, for each instance $\boldsymbol{x}$, define a threshold function $h(\boldsymbol{x})$ and the size of label set $s(\boldsymbol{x}) = |\{j | f_j(\boldsymbol{x}) > h(\boldsymbol{x}), j = 1, \cdots, k\}|$. Our goal is to obtain $h(\boldsymbol{x})$ through a supervised learning method. For the training data $\boldsymbol{x}_i$, its label ranking value $f_1(\boldsymbol{x}_i), \cdots, f_k(\boldsymbol{x}_i)$ can be given by the foregoing ranking algorithm, and its corresponding threshold $h(\boldsymbol{x}_i)$ is simply defined by

$$h(\boldsymbol{x}_i) = \frac{1}{2}(\min\{f_j(\boldsymbol{x}_i), j \in y_i\} + \max\{f_j(\boldsymbol{x}_i), j \in \overline{y}_i\})$$

Once the training data $(\boldsymbol{x}_1, h(\boldsymbol{x}_1)), \cdots, (\boldsymbol{x}_u, h(\boldsymbol{x}_u))$ are generated, we can use off the shelf learning methods to learn $h(\boldsymbol{x})$. In this paper, linear Support Vector Regression [26] have been adopted to solve $h(\boldsymbol{x})$.

Actually, all the label ranking based algorithms toward multi-label learning can apply this postprocessing approach to predict the size of label set.

3.5. Connections to Manifold Regularization

Belkin *et al.* [4] extended the traditional regularization algorithms with different empirical cost functions and complexity measures in an appropriately chosen Reproducing Kernel Hilbert Space (RKHS), *e.g.* SVM and Regularized Least Squares (RLS), and suggested a general semi-supervised learning framework for binary classification problems by incorporating manifold regularization. More concretely, given a set of labeled examples $(\boldsymbol{x}_i, y_i)$ $(i = 1, \cdots, l)$ and a set of unlabeled examples $\boldsymbol{x}_j$ $(j = l+1, \cdots, l+u)$, the object model f^* can be obtained by solving the following optimization problem

$$f^* = \arg\min_{f \in \mathcal{H}_K} \frac{1}{l}\sum_{i=1}^{l} V(\boldsymbol{x}_i, y_i, f) + \gamma_A \|f\|_K^2 + \gamma_I \int_{\mathcal{M}} (\nabla_{\mathcal{M}} f, \nabla_{\mathcal{M}} f) \quad (28)$$

where $V(\boldsymbol{x}_i, y_i, f)$ is some empirical loss function, such as squared loss $(y_i - f(\boldsymbol{x}_i))^2$ for RLS or the hinge loss $\max\{0, 1 - y_i f(\boldsymbol{x}_i)\}$ for SVM. Penalizing the RKHS norm $\|f\|_K^2$ imposes smoothness conditions on possible solutions. The manifold regularizer $\int_{\mathcal{M}} (\nabla_{\mathcal{M}} f, \nabla_{\mathcal{M}} f)$ reflects the intrinsic structure of the marginal distribution $P(\boldsymbol{x})$, *i.e.* the conditional probability distribution $P(y|\boldsymbol{x})$ should vary smoothly along the geodesics in the intrinsic geometry of $P(\boldsymbol{x})$. Note the term $\int_{\mathcal{M}} (\nabla_{\mathcal{M}} f, \nabla_{\mathcal{M}} f)$ may be approximated on the basis of labeled and unlabeled data using the graph Laplacian.

As a matter of fact, Eq. (10) can be converted into the following formulation

$$\min C \sum_{i=1}^{n} \frac{1}{|y_i||\overline{y}_i|} \sum_{(p,q) \in y_i \times \overline{y}_i} \max\{0, 1 - f_p(\boldsymbol{x}_i) + f_q(\boldsymbol{x}_i)\} + \frac{1}{2}\sum_{i=1}^{m} \|\boldsymbol{w}_i\|^2 + \frac{1}{2}\lambda \text{trace}(\widehat{F}^T L \widehat{F})$$

where $\max\{0, 1 - f_p(\boldsymbol{x}_i) + f_q(\boldsymbol{x}_i)\}$ is the empirical loss function, $\sum_{i=1}^{m} \|\boldsymbol{w}_i\|^2$ is the regularizer in the RKHS and $\text{trace}(\widehat{F}^T L \widehat{F})$ is the manifold regularizer that imposes the manifold smoothness. Therefore, to some extent, our algorithm can be viewed as a generalization of the original manifold regularization from binary classification to multi-label classification. However, an evident distinction is that our algorithm is supervised not semi-supervised since the hypergraph only contains the labeled examples.

4. Experiments

We performed experiments on two real world multi-label classification problems arising in image classification and video indexing. Comparisons are made with Binary-SVM and Rank-SVM [12].

4.1. Methods and Experimental Setup

Here, the three models used for multi-label classification are listed below

- Binary-SVM. In this model, first, for each category, train a linear SVM classifier independently. Then, the labels for each test instance can be obtained by aggregating the classification results from all the binary classifiers. Here, we use LIBSVM [7] to train the linear SVM classifiers.

- Rank-SVM [12]. In this model, first, as Eq. (9), implement Algorithm 1 ($\lambda = 0$) to train a linear label ranking system. Second, apply the prediction method for the size of label set in Section 3.4 to design the threshold model. Finally, for each test instance, combine the label ranking and threshold models, thus infer its labels.

- Rank-HLapSVM. This is our suggested algorithm. First, as Eq. (10), implement Algorithm 1 to achieve a linear label ranking system. Second, apply the method in Section 3.4 to design the threshold model. Finally, for each test instance, combine the label ranking and threshold models, thus infer its labels.

In Rank-HLapSVM, we use Eq. (5) to construct the hypergraph Laplacian L, where the weight $w(e)$ of the hyperedge is calculated by

$$w(e) = \exp(-\nu \overline{d}_e) \quad (29)$$

where ν is a nonnegative constant, and $\overline{d}_e$ is the average intra-class distance (Note each hyperedge corresponds to one category)

$$\overline{d}_e = \frac{\sum_{u,v \in e} \|\boldsymbol{x}_u - \boldsymbol{x}_v\|^2}{\delta(e)(\delta(e) - 1)} \quad (30)$$

The smaller the average intra-class distance, the larger the corresponding hyperedge weight.

In the above three models, it is necessary to identify the best value of model parameters such as C, λ and ν on the training data. Here, the grid search method with 5-fold cross validation is used to determine the best parameter values. For example, there is a penalty coefficient C in the linear SVM. In order to find a good parameter C, select different values $C = 2^{-6}, 2^{-5}, 2^{-4}, \cdots, 2^{0}, 2^{1}, 2^{2}, \cdots, 2^{13}$. For

Methods	F1 *Macro*	F1 *Micro*
Binary-SVM	0.7128	0.7294
Rank-SVM	0.7236	0.7457
Rank-HLapSVM	**0.7453**	**0.7629**

Table 1. Performance comparisons of three models on the ImageCLEF dataset

Methods	F1 *Macro*	F1 *Micro*
Binary-SVM	0.1865	0.2381
Rank-SVM	0.2232	0.2699
Rank-HLapSVM	**0.2382**	**0.2836**

Table 2. Performance comparisons of three models on the MediaMill dataset

each value of C, do 5-fold cross validation on the training data and compute the corresponding performance measure. Finally, select the one with the best performance as the value of C.

In addition, all the experiments are performed on a PC with Intel Core 2 Quad Q6600 2.40G CPU and 4G RAM.

4.2. Evaluation Metrics

We choose two measures, F_1 *Macro* and F_1 *Micro*, as the evaluation metrics for multi-label learning. F_1 *Macro* is the arithmetic average of F_1 scores over all the categories, and F_1 *Micro* can be seen as the weighted average of $F1$ scores over all the categories that emphasizes the performance on those categories with more positive instances (see [29] for details). The F_1 measure of the sth category is defined by

$$F_1(s) = \frac{2p_s r_s}{p_s + r_s} \tag{31}$$

where p_s and r_s are the precision and recall of the sth category, respectively.

4.3. The ImageCLEF Dataset

ImageCLEF [1] is a cross-language image retrieval track. We randomly pick 3500 documents from ImageCLEF collection, and choose the top 60 most popular categories. On average, each document is assigned to 3.9 categories. The 2000 documents is randomly selected for training and the left 1500 for test.

Table 1 shows the experimental results on the test set of the ImageCLEF data. For Binary-SVM and Rank-SVM, the penalty coefficient $C = 8$. For Rank-HLapSVM, the parameters $\nu = 2$, $C = 8$ and $\lambda = 4$. From the above experiments, we find that Rank-HLapSVM performs better that Rank-SVM, and Rank-SVM does better than Binary-SVM.

4.4. The MediaMill Dataset

The MediaMill dataset [2] recently released by Snoke *et al*. [22] is a challenging dataset for generic video indexing which are extracted from the TRECVID 2005/2006 benchmark. The dataset includes 101 semantic concepts such as 'People', 'Meeting' , 'Studio' and 'Military', and each video instance is represented as a 120-dimensional feature vector. Here, we randomly select a subset of the MediaMill dataset containing 3000 instances and 42 semantic concepts with more than 30 positive assignments. On average, each instance is assigned to 4.3 semantic concepts. The subset is equally split into a training set and a test set.

Table 2 lists the experimental results of three models on the test set of the MediaMill data. For Binary-SVM and Rank-SVM, the penalty coefficient $C = 1024$. For Rank-HLapSVM, the parameters $\nu = 8$, $C = 1024$ and $\lambda = 32$. It can be seen that based on F_1 *Macro* and F_1 *Micro*, Rank-HLapSVM is evidently superior to Rank-SVM, and Rank-SVM is evidently superior to Binary-SVM. This results also indicate that there indeed exists close correlations among different topics of MediaMill data and Rank-HLapSVM effectively exploits these correlations so as to improve the prediction performance.

Actually, from Eq. (10), if $\lambda = 0$, Rank-HLapSVM reduces to Rank-SVM. As long as there is the appropriate λ, Rank-HLapSVM can always perform better than Rank-SVM. Therefore, Rank-HLapSVM is a substaintial improvement of Rank-SVM.

4.5. Efficiency

Elisseeff and Weston [12] proposed to apply Frank and Wolfe's method, *i.e*. the conditional gradient method, to solve Rank-SVM. Its basic idea is to transform the quadratic optimization problem (9) into many simple linear programming and linear search problems and the corresponding time cost of each iteration is $O(n^2k)$. As depicted in Section 3.3, the time cost per iteration of the dual coordinate descent method is $O(n_\alpha \overline{m})$. Therefore, the time cost of our algorithm is approximately proportional to the amount of the instances while that of Frank and Wolfe's method is proportional to the square of the amount of the instances. Besides, our algorithm has a remarkable linear convergence rate while Frank and Wolfe's method does not. Hereby, in the practical problems, especially when the number of instances are much larger than the number of categories, the dual coordinate method should be superior to Frank and Wolfe's method.

Table 3 gives the average execution time of Rank-SVM using Frank and Wolfe's method and the dual coordinate descent method on the above two datasets respectively. It

[1] http://ir.shef.ac.uk/imageclef/

[2] http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/multilabel.html

Dataset	Frank and Wolfe's Method	DCD
ImageCLEF	7834.2	**155.2**
MediaMill	4519.8	**87.3**

Table 3. The average execution time $(/s)$ of Rank-SVM using Frank and Wolfe's method and the dual coordinate descent method (DCD) on the two datasets respectively

can be found that the dual coordinate descent method is much faster than Frank and Wolfe's method in those real data. This sufficiently verifies the efficiency of the dual coordinate descent method.

5. Conclusions and Future Work

In this paper, we have proposed a novel label ranking algorithm, Rank-HLapSVM, for multi-label classification. The hypergraph is constructed to capture the higher order relations among categories. We incorporate the hypergraph Laplacian regularizer into Rank-SVM and offer a more effective label ranking framework. The dual coordinate descent method is introduced to efficiently solve the corresponding quadratic optimization problem. The experimental results on the real data show that Rank-HLapSVM indeed performs better than Binary-SVM and Rank-SVM.

However, our current algorithm cannot handled very large scale data, since the hypergraph Laplacian L need to be directly calculated. In order to facilitate the multi-label classification tasks for large scale data, we will further develop some fast approximate algorithms of calculating L and $(I + \lambda X^T L X)^{-1}$ that have low time and memory cost.

In addition, it is easy to obtain the nonlinear version of Rank-HLapSVM by the kernel method. In the future, we will also test our algorithm and its kernel version on more real data.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under the grant No. 60835002 and Tsinghua-HP Multimedia Research Center.

References

[1] S. Agarwal, K. Branson, and S. Belongie. Higher order learning with graphs. In *Proc. of ICML*, 2006. 2

[2] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. In *Proc. of ICML*, 2003. 2

[3] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifold. *Machine Learning*, 56:209–239, 2003. 2

[4] M. Belkin, P. Niyogi, and V. Sindhwani. On manifold regularization. In *Proc. of AISTATS*, 2005. 3, 6

[5] C. Berge. *Graphs and Hypergraphs*. Elsevier, 1973. 2

[6] M. R. Boutella, X. Luoh, J.and Shen, and C. M. Browna. Learning multi-label scene classification. *Pattern Recognition*, 37:1757–1771, 2004. 1

[7] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm. 6

[8] E. Chang, K. Goh, G. Sychay, and G. Wu. Content-based soft annotation for multimodal image retrieval using bayes point machines. *IEEE Tran. on Circuits and Systems for Video Tech. Special Issue on Conceptual and Dynamical Aspects of Multimedia Content Description*, 13(1), 2003. 1

[9] G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *Proc. of the 8th SIAM Conference on Data Mining*, 2008. 2

[10] K. Crammer and Y. Singer. A new family of online algorithms for category ranking. In *Proc. of SIGIR*, 2002. 2

[11] O. Dekel, C. D. Manning, and Y. Singer. Log-linear models for label ranking. In *Proc. of NIPS*, 2003. 2, 3

[12] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Proc. of NIPS*, 2001. 1, 2, 3, 5, 6, 7

[13] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *Proc. of NIPS*, 2005. 2

[14] C.-J. Hsieh, K.-W. Chang, C. jen Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *Proc. of ICML*, 2008. 4, 5

[15] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proc. of ECML*, 1998. 1

[16] F. Kang, R. Jin, and R. Sukthankar. Correlated label propagation with application to multi-label learning. In *Proc. of CVPR*, 2006. 2

[17] Y. Liu, R. Jin, and L. Yang. Semi-supervised multi-label learning by constrained non-negative matrix factorization. In *Proc. of AAAI*, 2006. 2

[18] Z.-Q. Luo and P. Tseng. On the convergence of coordinate descent method for convex differentiable minimization. *J. Optim. Theory Appl.*, 72:7–35, 1992. 5

[19] A. McCallum. Multi-label text classification with a mixture model trained by em. In *Proc. of AAAI Workshop on Text Learning*, 1999. 1, 2

[20] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. On maximum margin hierarchical multi-label classification. In *Proc. of NIPS Workshop on Learning With Structured Outputs*, 2004. 2

[21] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2-3), 2000. 2

[22] C. G. Snoek, M. Worring, J. C. van Gemert, J.-M. Geusebroek, and A. W. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proc. of SIGMM*, 2006. 1, 7

[23] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008. 2, 3

[24] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Proc. of NIPS*, 2003. 2

[25] N. Ueda and K. Saito. Parametric metric models for multi-labelled text. In *Proc. of NIPS*, 2002. 2

[26] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995. 5

[27] U. von Luxburg. A tutorial on spectral clustering. Technical report, Max Planck Institute for Biological Cybernetics, 2006. 2

[28] F. Wang and C. Zhang. Label propagation through linear neighborhoods. In *Proc. of ICML*, 2006. 2

[29] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2), 1999. 1, 7

[30] K. Yu, S. Yu, and V. Tresp. Multi-label informed latent semantic indexing. In *Proc. of SIGIR*, 2005. 1, 2

[31] S. Zhu, X. Ji, W. Xu, and Y. Gong. Multi-labelled classification using maximum entropy method. In *Proc. of SIGIR*, 2005. 1, 2

[32] X. Zhu. Semi-supervised learning literature survey. Techinial Report TR 1530, University of Wisconsin-Madision, 2006. 2