Correlated Label Propagation with Application to Multi-label Learning

Feng Kang[1] kangfeng@msu.edu
Rong Jin[1] rongjin@cse.msu.edu
Rahul Sukthankar[2,3] rahuls@cs.cmu.edu
[1]Michigan State University [2]Intel Research Pittsburgh [3]Robotics Institute, Carnegie Mellon

Abstract

Many computer vision applications, such as scene analysis and medical image interpretation, are ill-suited for traditional classification where each image can only be associated with a single class. This has stimulated recent work in multi-label learning where a given image can be tagged with multiple class labels. A serious problem with existing approaches is that they are unable to exploit correlations between class labels. This paper presents a novel framework for multi-label learning termed Correlated Label Propagation *(CLP) that explicitly models interactions between labels in an efficient manner. As in standard label propagation, labels attached to training data points are propagated to test data points; however, unlike standard algorithms that treat each label independently, CLP simultaneously co-propagates multiple labels. Existing work eschews such an approach since naive algorithms for label co-propagation are intractable. We present an algorithm based on properties of submodular functions that efficiently finds an optimal solution. Our experiments demonstrate that CLP leads to significant gains in precision/recall against standard techniques on two real-world computer vision tasks involving several hundred labels.*

1 Introduction

Multi-label learning refers to problems where an instance can be assigned to multiple classes. This differs from multi-class learning where every instance can be assigned to only *one* class even though the number of classes is more than two. The essential difference between multi-class learning and multi-label learning is that classes in multi-class learning are assumed to be mutually exclusive while classes in multi-label learning are often correlated. Consider the problem of automatically annotating images with textual words, in which each annotation can be treated as a separate class label. Since many words are semantically related, class labels are correlated. Often, this correlation among classes can be helpful for predicting class labels of test examples. For instance, the words "ocean" and "sky" are both strongly related to the blue color feature. Therefore it may be difficult to distinguish these two words based on the color features alone. However, if we are confident that an image should be annotated with "grass", then it is more likely that a region of blue in the same image should be annotated as "sky" rather than "ocean".

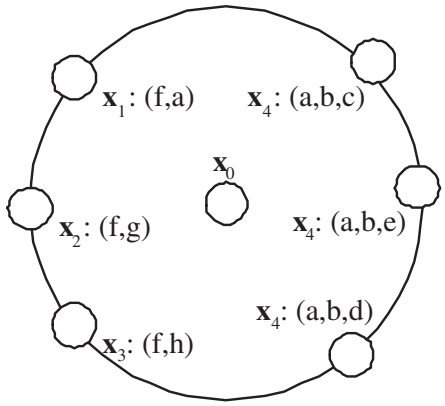

Figure 1. An illustrative example of correlated label propagation

We present a novel framework, **Correlated Label Propagation**, for multi-label learning that explicitly exploits high-order correlation between labels. Unlike most existing approaches that usually only consider the propagation of a single class label between training examples and test examples, the proposed framework takes into account the simultaneous propagation of multiple labels. To illustrate the basic idea of our approach, consider the example in Figure 1. Points $\mathbf{x}_1, \ldots, \mathbf{x}_6$ are training examples and are assigned to six different classes. Point $\mathbf{x}_0$ is the test point. For the convenience of discussion, let us assume that the six training examples share the same similarity to the test example $\mathbf{x}_0$. If we only consider the first-order propagation, which amounts to the counting of label frequency in the neighborhood of $\mathbf{x}_0$, we will find that "a" is the most likely class label, and that both "b" and "f" are the next

0-7695-2646-2/06 $20.00 (c) 2006 IEEE

equally-likely class labels. However, if we allow two labels to be propagated simultaneously from the training examples to the testing one, we will rank "b" ahead of "f" since "b" co-occurs more frequently with "a" than "f".

In addition to the general framework, this paper also presents an efficient algorithm for correlated label propagation. The rest of the paper is arranged as follows. Section 2 briefly reviews the related work. Section 3 describes the proposed framework for multi-label learning. Section 4 presents the efficient algorithm for solving the related optimization problem. Section 5 presents experiments employing the proposed algorithm.

2 Related work

We first review the related work on multi-label learning, followed by a discussion of label propagation.

2.1 Multi-label Learning

The most commonly-used approach for multi-label learning is to divide it into a number of binary classification problems [3, 11, 24]. In particular, for each class c, training examples are organized into two groups: the group of "positive" examples that are labeled by c, and the group of "negative" examples that are not labeled by c. A binary classifier is learned for c based on these two groups of examples. One disadvantage with such approaches is that they treat each class label independently, and are therefore unable to exploit any correlation among class labels. Another disadvantage is that these approaches do not easily scale to a large number of classes since a binary classifier has to be built for every class. Finally, most binary classification approaches toward multi-label learning suffer severely from the unbalanced data problem [22], particularly when the number of classes is large. This is because, when the number of classes is large, the number of "negative" examples is overwhelmingly larger than the number of "positive" examples for any individual class. As a result, it is likely that the binary classifiers will output the negative labels for all instances.

Another group of approaches toward multi-label learning is label ranking [5, 7, 18]. These approaches learn a ranking function of class labels from the labeled examples and apply it to order the class labels for the given test examples. Compared to binary classification approaches, the label ranking approaches are generally superior at dealing with large numbers of classes because only a single ranking function is learned to compare the relevance of individual class labels with respect to the test examples. The label ranking approaches also avoid the problem of unbalanced data since no binary decision has to be made regarding class labels. However, similar to the binary classification approaches, the label ranking approaches are unable to exploit the label correlation information.

There has been little previous work in exploiting the label correlation within the context of multi-label learning. [21] proposes a generative model for multi-label learning that explicitly incorporates the pairwise correlation between any two class labels. [8] introduces a Bayesian model to assign labels through underlying latent representations. [28] suggests a maximum entropy model for exploring the label correlation for multi-label learning. In [25], a linear model is assumed between the input features and the output class labels, and a regression model is used to find the appropriate linear combination weights. The label correlation is explored by imposing a common prior for the combination weights on different classes. Despite these efforts in exploiting label correlation information, most of the research is limited to pairwise correlation of class labels. Finally, there are several papers on multi-label learning that assume a particular structure among the class labels. [15] assumes that class labels can be divided into a small number of disjoint clusters and [2, 17] assume a hierarchical structure among the class labels. We believe that, given the complicated relationships between class labels, such assumptions are likely to be violated in many real-world applications. In contrast, the framework proposed in this paper can take into account the label correlation of *any* order. We also show that the proposed framework can be solved efficiently based on the concept of submodular functions.

2.2 Label Propagation

Label propagation approaches have recently become popular in machine learning. The main idea is to propagate the labels from training examples to test examples through their similarities. The label information propagated from different training examples are then accumulated and used as the basis for scoring the class labels of test examples. A number of machine learning methods have been developed for label propagation, including kernel-based K Nearest Neighbor (kNN) [14], Gaussian processes [23, 4], harmonic functions [29], and Green functions [27]. This work distinguishes itself from the previous work in that multiple labels can be propagated simultaneously from the training examples to the test examples, which is the key to exploiting the correlation among multiple labels.

3 A Correlated Label Propagation Framework for Multi-label Learning

This section first presents a brief overview of the setup of multi-label learning based on label ranking. We then describe the proposed framework for multi-label learning, fol-

lowed by the efficient greedy algorithm using the concept of submodular functions.

3.1 The Label Ranking Formulation of Multi-label Learning

Let $\mathcal{D} = \{(\mathbf{x}_1, \mathcal{S}_1), \ldots, (\mathbf{x}_n, \mathcal{S}_n)\}$ denote the set of labeled examples, where n is the number of training examples and m is the number of classes. Each $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})$ is an input vector of d dimension. Each set $\mathcal{S}_i$ contains the class labels that are assigned to the i-th training example. For the convenience of presentation, we will employ a binary vector to represent a set of class labels. In particular, for a class label set $\mathcal{S}$, its vector representation $\mathbf{t}(\mathcal{S}) = (t_{i,1}, \ldots, t_{i,m})$ has its j-th element set to 1 only when $j \in \mathcal{S}$ and zero otherwise. Given a test point $\mathbf{x}_t$, our goal is to determine a confidence vector $\mathbf{z}_t = \{z_{t,1}, \ldots, z_{t,m}\}$ such that each component $z_{t,i}$ indicates the confidence of assigning $\mathbf{x}_t$ to the i-th class.

3.2 Correlated Label Propagation for Multi-label Learning

To motivate the proposed framework, we first describe the kernel-based kNN approach, which is one of the most popular methods for label propagation and can be found in many applications of computer vision [1, 19, 20].

Suppose the similarity of any two data points is measured by a kernel function $K(\cdot, \cdot) : \mathbf{R}^d \times \mathbf{R}^d \rightarrow \mathbf{R}$. Consider the case of single-step propagation. The score of assigning the j-th class to the test example $\mathbf{x}_t$, i.e., $z_{t,j}$, could be estimated by

$$z_{t,j} = \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(j \in \mathcal{S}_i), \tag{1}$$

where $I(j \in \mathcal{S})$ is an indicator function that outputs 1 when the j-th class belongs to set $\mathcal{S}$ and is zero otherwise. There are two problems with the expression in Equation 1:

- *Overestimated Confidence Score*. Equation 1 assumes that a training example $\mathbf{x}_i$ will propagate *all* of its class labels to the test example $\mathbf{x}_t$ according to the similarity $K(\mathbf{x}_t, \mathbf{x}_i)$. This is not necessarily true since maybe only *some* of the class labels of $\mathbf{x}_i$ will be propagated to $\mathbf{x}_t$ even though $\mathbf{x}_i$ is similar to $\mathbf{x}_t$.
- *Independent Label Propagation*. As indicated in Equation 1, each class label is propagated from training examples to the test example independently of the other class labels. In particular, the computation of the confidence score $z_{t,j}$ for the j-th label is independent from the confidence scores assigned to other class labels.

To resolve the problem of overestimated confidence score, we replace the equality constraint in Equation 1 with the following inequality constraint:

$$z_{t,j} \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(j \in \mathcal{S}_i). \tag{2}$$

The above inequality indicates that the confidence score propagated from training examples to the test example is upper bounded by the sum of the pairwise similarity $K(\cdot, \cdot)$. Note that no explicit value of the confidence score $z_{t,j}$ is specified in the above expression.

To incorporate the label correlation information into label propagation, we consider the propagation of multiple labels. Let the binary vector $\mathcal{S}$ be the set of labels that are propagated from the training examples $\mathcal{D}$ to the test example $\mathbf{x}_t$. We denote by $s_t(\mathcal{S})$ the confidence score of assigning *any subset* of $\mathcal{S}$ to $\mathbf{x}_t$. Similar to Equation 2, we introduce the following constraint on the confidence score $s_t(\mathcal{S})$, i.e.,

$$s_t(\mathcal{S}) \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(\mathcal{S} \cap \mathcal{S}_i \neq \phi) \tag{3}$$

where $I(\mathcal{S} \cap \mathcal{S}_i \neq \phi)$ is used to ensure that only the training examples whose class labels overlap with the set $\mathcal{S}$ are included in computing the confidence score. To link $z_{t,j}$, i.e., the confidence score of assigning individual classes, to $s_t(\mathcal{S})$, i.e., the confidence score of assigning multiple classes, we assume the following inequality,

$$\sum_{j=1}^{m} z_{t,j} I(j \in \mathcal{S}) \leq s_t(\mathcal{S}). \tag{4}$$

The above inequality implies that for a single data point, the confidence of assigning any subset of class label set $\mathcal{S}$ to $\mathbf{x}_t$ should be no less than the confidence of assigning the class label separately in $\mathcal{S}$ to $\mathbf{x}_t$. Combining Equation 4 with Equation 1, we obtain

$$\sum_{j=1}^{m} z_{t,j} I(j \in \mathcal{S}) \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(\mathcal{S} \cap \mathcal{S}_i \neq \phi).$$

The above expression can be simplified if we present it in the vector form of the class labels, i.e.,

$$\mathbf{z}_t^T \mathbf{t}(\mathcal{S}) \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(\mathbf{t}(\mathcal{S})^T \mathbf{t}(\mathcal{S}_i)) \tag{5}$$

Hence, given m different class labels and multi-labeled examples $\mathcal{D}$, the confidence $\mathbf{z}$ of assigning individual classes to the test example $\mathbf{x}_t$ is subject to the following constraints:

$$\begin{aligned} &\forall \mathbf{t} \in \{0, 1\}^m,\ \mathbf{z}_t^T \mathbf{t} \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(\mathbf{t}^T \mathbf{t}(\mathcal{S}_i)) \\ &\mathbf{z} \succeq 0. \end{aligned} \tag{6}$$

Furthermore, we can generalize the indicator function $I(x)$ to a *concave* function $\Omega(x)$, which we term the **Label Kernel Function**. Then, the constraints in Equation 6 are generalized in the following form:

$$\forall \mathbf{t} \in \{0,1\}^m,\ \mathbf{z}_t^T \mathbf{t} \leq \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) \Omega(\mathbf{t}^T \mathbf{t}(\mathcal{S}_i))$$
$$\mathbf{z} \succeq 0 \tag{7}$$

A detailed discussion of the label kernel function $\Omega(x)$ appears in the later part of this section.

It is insufficient to identify the appropriate confidence scores $\mathbf{z}$ only with the constraints. Thus, we assume that among all the confidence scores that satisfy the constraints in Equation 7, the optimal solution $\mathbf{z}$ is the one that "maximally" satisfies the constraints. This assumption leads to the following optimization problem for $\mathbf{z}$:

$$\max_{\mathbf{z} \in \mathbf{R}^m} \quad \sum_{k=1}^{m} \alpha_k z_k$$
$$\text{s.t.} \quad \forall \mathbf{t} \in \{0,1\}^m : \mathbf{z}^T \mathbf{t} \leq \sum_{i=1}^{n} K(\mathbf{x}_i, \mathbf{x}_q) \Omega(\mathbf{t}^T \mathbf{t}(\mathbf{S}_i))$$
$$\mathbf{z} \succeq 0 \tag{8}$$

where $\{\alpha_k > 0\}_{k=1}^m$ are the weights for the class labels. Notice that the problem in Equation8 is a linear programming problem, and therefor the solution will be on the extreme points of the region bounded by the constraints.

Despite the simplicity, solving Equation 8 efficiently is not trivial. This is because:

- *Efficiency*: The number of constraints in Equation 8 is exponential in the number of classes m. When m is large (e.g., 100), the number of constraints will be too large to be handled by any linear programming algorithm.
- *Undetermined Weights*: The solution to Equation 8 depends on the weights $\{\alpha_k\}_{k=1}^m$, whose exact values are difficult to determine.

4 Efficient Learning Algorithm

In this section, we show that when the label kernel function $\Omega(x)$ is a concave function, there is a simple and greedy algorithm for finding the optimal solution to the problem in Equation 8. Furthermore, the solution only depends on the relative *order* of weights $\{\alpha_k\}_{k=1}^m$, and is independent of their exact values. The algorithm for estimating label confidence scores $\mathbf{z}$ is summarized in Figure 2. This greedy algorithm is based on the following theorem from discrete optimization [16]:

Input
- $\mathbf{x}_t$: the test example
- $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_m > 0$

Output: optimal label scores $(z_{t,1}, \ldots, z_{t,m})$ for $\mathbf{x}_t$

For k = 1,...,m
- Let class label set $\mathcal{T}_k = \{1, 2, \ldots, k\}$.
- $f(\mathcal{T}_k) = \sum_{i=1}^{n} K(\mathbf{x}_i, \mathbf{x}_t) \Omega(\mathbf{t}^T(\mathcal{T}_k) \mathbf{t}(\mathcal{S}_i))$
- $z_{t,k} = f(\mathcal{T}_k) - f(\mathcal{T}_{k-1})$

Figure 2. Algorithm for finding the optimal solution to Equation 8

Given: (1) a finite set $\mathcal{N}$, (2) a set function $f : 2^{\mathcal{N}} \rightarrow \mathbf{R}$ with $f(\phi) \geq 0$, and (3) a weight vector $\mathbf{w} \in \mathbf{R}^{|\mathcal{N}|}$. Then, the linear programming problem:

$$\max_{\mathbf{w} \in \mathbf{R}^{|\mathcal{N}|}} \quad \mathbf{w}^T \mathbf{x}$$
$$\text{s. t.} \quad \forall \mathcal{A} \subseteq \mathcal{N},\ \sum_{e \in \mathcal{A}} x(e) \leq f(\mathcal{A})$$
$$\forall e \in \mathcal{N}, x(e) \geq 0$$

can be solved by the following greedy algorithm if the set function f is submodular:

- Sort elements of $\mathcal{N}$ as $w(e_1) \geq w(e_2) \geq ... \geq w(e_n)$
- Let $\mathcal{V}_0 = \phi$
  For i =1,...,n, let
  $\mathcal{V}_i = \mathcal{V}_{i-1} + e_i$, and $x(e_i) = f(\mathcal{V}_i) - f(\mathcal{V}_{i-1})$.

The validity of applying the above theorem to our problem defined in Equation 8 relies on the fact that the function f in our algorithm, i.e.,

$$f(\mathbf{u}) = \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) \Omega(\mathbf{u}^T \mathbf{t}_i) \tag{9}$$

is a submodular function if $\Omega(x)$ is a concave function. We present a proof that f is submodular in the Appendix.

Remark: it is interesting that the kernel-based k Nearest-Neighbor is a special case of the algorithm in Figure 2, given by setting $\Omega(x) = x$.[1] This is because

$$\begin{aligned} z_{t,k} &= f(\mathcal{T}_k) - f(\mathcal{T}_{k-1}) \\ &= \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) \left(\mathbf{t}(\mathcal{T}_k) - \mathbf{t}(\mathcal{T}_{k-1}))^T \mathbf{t}(\mathcal{S}_i)) \right) \\ &= \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) \left(\mathbf{e}_k^T \mathbf{t}(\mathcal{S}_i)) \right) \\ &= \sum_{i=1}^{n} K(\mathbf{x}_t, \mathbf{x}_i) I(k \in \mathcal{S}_i) \end{aligned}$$

[1] The linear function x is both a concave and a convex function.

where $\mathbf{e}_k$ is the vector whose elements are all zero except that the k-th element is 1. In the last step of the above derivation, we use the property:

$$\mathbf{e}_k^T \mathbf{t}(\mathcal{S}_i) = \begin{cases} 1 & k \in \mathcal{S}_i \\ 0 & \text{otherwise.} \end{cases}$$

4.1 Selecting Weights α and the Label Kernel Function

This section discusses the impact of different choices for weights $\{\alpha_k\}_{k=1}^m$ and the label kernel function $\Omega(x)$.

Choice of Weights: The solution returned by the label propagation algorithm is dependent only on the relative *order* of the weights, $\{\alpha_k\}_{k=1}^m$. There are two straightforward choices for the weights:

1. order the weights α to be in the same order as class frequency, namely $\alpha_i \geq \alpha_j \longleftrightarrow p_i \geq p_j$;
2. order the weights to be in the reverse order of class frequency, namely $\alpha_i \geq \alpha_j \longleftrightarrow p_i \leq p_j$.

Above, p_i is the frequency of the i-th class in the training data. A potential problem with the first choice for α is that assigning large weights to the popular classes will mean that those classes will tend to be selected before the rare classes. Since popular classes are correlated with many more classes than rare classes, choosing the weights for the popular classes first could cause the test sample to overlap with the labeled samples heavily from the beginning. This is best illustrated using an example. Consider the two classes "water" and "whale", where the former is popular and the latter rare; every time "whale" appears in the label set of an training data, "water" also appears but not vice versa. If "water" were chosen before "whale", then no additional overlap information would be introduced when the weight for "whale" was determined. By contrast, the second choice (selecting weights in reverse order of class frequency) allows the rare classes to determine their confidence scores before the popular classes. In our example, this means that new overlapping information is introduced when the weight for "water" is selected after the weight for "whale". For these reasons, our experiments employ the second choice for the weights.

Choice of the label kernel function: As discussed above, a prerequisite for the label kernel function $\Omega(x)$ is that it should be concave. In addition to ensuring that $f(\mathbf{u})$ in Equation 9 is a submodular function, the choice of a concave function is also consistent with the principle of *Decreasing Marginal Returns* in Economics. Namely, that more information is gained from the first few observations than the repeated observation of the same evidence.

Table 1 lists four examples of label kernel functions: the δ function, the sigmoid function, the exponential function, and the count function. As indicated by the expressions in Table 1, these four functions behave very differently. The δ function outputs 1 whenever the input is positive. Thus, no matter how many class labels are shared between the class labels of a training example and the class labels of propagation, the amount of label confidence propagated from the training example remains the same. The exponential function is a monotonically-increasing function with a maximum value of 1 (for $x \geq 0$). Unlike both the δ function and the exponential function, which output zero when the input is zero, the sigmoid function has a non-zero output even when the input is zero. This allows for any training example to propagate confidence score to the test example even when its assigned classes do not overlap with the class labels of propagation. This property plays a role analogous to smoothing in information retrieval (e.g., [26]). Finally, as discussed in the previous section, the count function leads to the standard kernel-based kNN algorithm.

Table 1. Examples of label kernel functions used in experiments

δ function	$\delta(x) = \begin{cases} 0 \text{ if } x = 0 \\ 1 \text{ if } x > 0 \end{cases}$
sigmoid function	$F(x) = \frac{1}{1+e^{-x}}$
exponential function	$F(x) = 1 - 2^{-\alpha x}$
count function	$F(x) = x$

5 Experiments

In our experiments, we focus on the study of multi-label learning with a very large number of class labels. In particular, we address the following questions when examining the effectiveness of the proposed label propagation method:

- *Which label kernel functions is most effective*?
- *How effective is the proposed algorithm compared to other state-of-art approaches*?

5.1 Dataset

Our experiments employ two datasets: the Corel dataset for automatic image annotation and the CLEF dataset for text categorization. The first dataset contains 5000 images, from which 4500 images are used for training and 500 images are used for testing. Each image is segmented into several regions and the regions of similar features are clustered into 500 clusters, known as blobs in [6]. Then, each image is represented by a binary vector of these 500 blobs. 374 words are used for annotation and each of the images is tagged with several words to describe its content. The average number of annotation words for each image is 3.5. We treat each word as a class label to apply our algorithms. The

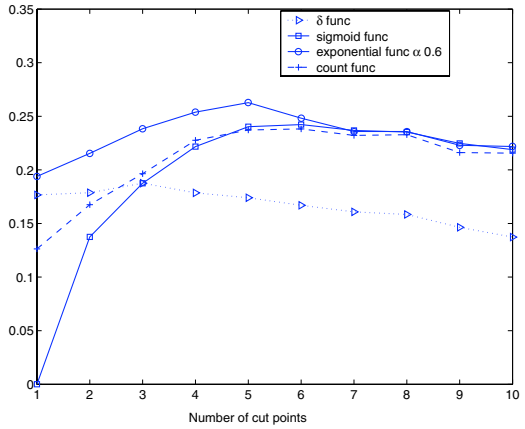

Figure 3. Averaged $F1$ measure for the Corel dataset

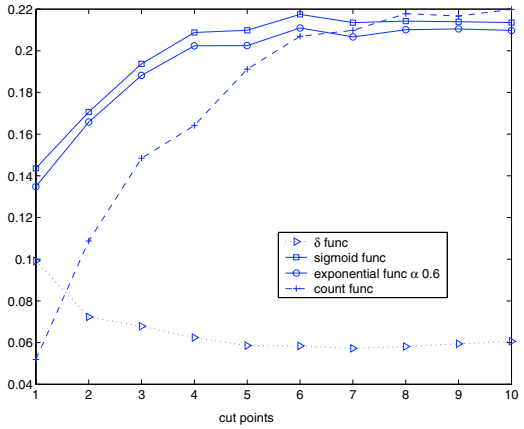

Figure 4. Average $F1$ measure for the CLEF dataset

CLEF dataset [9] includes a total of 28,533 documents that are assigned to 937 categories. The average number of categories assigned to each document is 4.6. Each document is represented as a vector of 18,736 terms after stemming and removing stopwords. 70% of the documents are randomly selected for training and the remaining for testing.

5.2 Evaluation Metrics

Our experiments focus on the evaluation of label ranking. Following [24], we use the micro $F1$ measurement, which is the average of $F1$ score across different class labels. Given the set of assigned class labels, $\mathcal{S}_i$ and by $\hat{\mathcal{S}}_i$ the set of predicted class labels $\hat{\mathcal{S}}_i$, the $F1$ measurement of the k-th class label is computed using:

$$F_1 = \frac{2p_k r_k}{p_k + r_k}$$

where p_k and r_k are the precision and recall of the k-th label, respectively and are calculated as follows:

$$p_k = \frac{|\{\mathbf{x}_i | k \in \mathcal{S}_i \wedge k \in \hat{\mathcal{S}}_i\}|}{|\{\mathbf{x}_i | k \in \hat{\mathcal{S}}_i\}|}$$

$$r_k = \frac{|\{\mathbf{x}_i | k \in \mathcal{S}_i \wedge k \in \hat{\mathcal{S}}_i\}|}{|\{\mathbf{x}_i | k \in \mathcal{S}_i\}|}.$$

Since some of the class labels are too rare to be predicted by any algorithm, only the class labels that are predicted at least once by either the proposed algorithm or the baseline algorithms are used for evaluation. In order to see the quality of ranked class labels, we evaluate the averaged F_1 measurement for the top k ranked class labels (k ranging from 1 to 10). Finally, the kernel function $K(\mathbf{x}, \mathbf{x}')$ is based on the relevance language model [13], which has been successfully applied to automatic image annotation [10]. More specifically, the similarity of a test example $\mathbf{x}_t$ to a training example, denoted by $K(\mathbf{x}_t, \mathbf{x}_i)$, is calculated as:

$$K(\mathbf{x}_t, \mathbf{x}_i) = \Pr(\mathbf{x}_t | \mathbf{x}_i) = \prod_k [p(k|\mathbf{x}_i)]^{x_{t,k}} \qquad (10)$$

where

$$p(k|\mathbf{x}_i) = \beta \frac{x_{i,k}}{\sum_{k'} x_{i,k'}} + (1-\beta) \frac{\sum_{j=1}^{n} x_{j,k}}{\sum_j = 1^n \sum_{k'} x_{i,k'}}.$$

In our experiments, the parameter β is set to be 0.8 based on cross validation.

5.3 Comparison of Different Label Kernel Functions

In this section, we compare the performance of different label kernel function $\Omega(x)$. Figures 3 and 4 show the averaged $F1$ measurement of the four kernel label functions on the Corel and CLEF datasets, respectively. Note that the count function corresponds to the kernel-based kNN approach.

From the performance on these two datasets, we see a similar trend in performance. First, we observe that, among the four label kernel functions, the δ function performs the worst for almost all ranks. This is due to the property of the δ function that gives a constant value regardless of the number of class labels that overlap between the assigned class labels of training examples and the propagated class labels. Second, the performance of the three kernel label functions, namely the sigmoid function, the exponential function, and the count function, differ significantly when the cut-off rank is small. Once the cut-off rank is large (e.g., 5 for the Corel dataset and 6 for the CLEF dataset), the three label kernel

functions deliver similar results. This is because when the number of selected class labels is large, all of the label kernel functions will be able to identify more or less the similar set of class labels. As a result, the $F1$ measurement of all three label kernel function are close to each other when the cut-off rank is large. Third, we see that, among the four functions, the exponential function appears to provide the best or close to the best performance on both datasets.

5.4 Experiments for Automatic Image Annotation

In this study, we compare the proposed label propagation approach to two established approaches for automatic image annotation using the Corel dataset. They are the statistical translation model [6], and the multi-class support vector machines (SVMs) [12]. To predict class labels for a test image, we will first apply the two baseline models to compute the score of each class label and rank the class labels in the in the descending order of their scores. Then, only the class labels before the cut-off rank will be chosen as the labels for the test image.

The $F1$ measurement of two baseline approaches at different cut-off ranks of the translation model is presented in Figure 5. For the purpose of comparison, we also include the result of the proposed correlated label propagation approach that uses the exponential function. First, we observe that both the proposed approach and the multi-class SVMs significantly outperforms the translation model across all ranking points. Second, the proposed approach achieve similar performance as the multi-class SVMs. In particular, the multi-class SVMs outperforms the proposed label propagation approach when the number of predicted class labels is smaller than 4, and the proposed approach starts to outperforms the multi-class SVMs afterwards. Given the computational simplicity of the proposed approach, we believe that it is advantageous to use the proposed approach for automatic image annotation than the SVMs, particularly when the number of labels is large.

6 Conclusion

This paper proposes a novel framework, *correlated label propagation*, for multi-label learning that explicitly addresses the problem of label dependence. Unlike previous approaches to multi-label learning that either treat class label independently or only take into count the pairwise correlations, our proposed algorithm exploits label correlations of any order. We formulate the proposed framework as a linear programming problem with an exponential number of constraints that cannot be practically solved using standard techniques. We provide an algorithm, based on properties of submodular functions that can solve this problem exactly

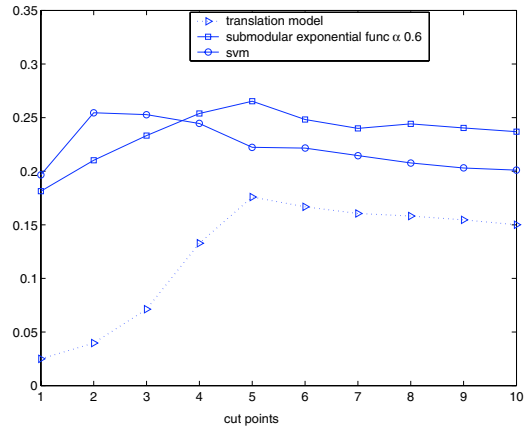

Figure 5. Average $F1$ measure for the correlated label propagation, the translation model and the support vector machine

and efficiently. We verify (both theoretically and experimentally) that the proposed approach is significantly more effective than the kernel-based kNN approach for multi-label learning. Our experiments also show that correlated label propagation is more effective than the statistical translation model for automatic image annotation.

Appendix A

Theorem 1 *Function $f(\mathbf{u})$ in Equation 9 is a submodular function if the label kernel function $\Omega(x)$ is concave.*

Proof

To show that $f(\mathbf{u})$ is submodular, we use the following necessary and sufficient conditions for submodular functions:

For any set $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{N}$ and element $e \in \mathcal{N} \setminus \mathcal{B}$,

$$f(\mathcal{A} \cup e) - f(\mathcal{A}) \geq f(\mathcal{B} \cup e) - f(\mathcal{B})$$

iff f is a submodular function. Using the binary vector representation of sets, the above condition can be written as:

$$f(\mathbf{t}(\mathcal{A} \cup e)) - f(\mathbf{t}(\mathcal{A})) \geq f(\mathbf{t}(\mathcal{B} \cup e)) - f(\mathbf{t}(\mathcal{B}))$$

holds when $\mathbf{t}(\mathcal{B}) \succeq \mathbf{t}(\mathbf{A})$ and $\mathbf{t}(e)^T \mathbf{t}(\mathcal{B}) = 0$. Using the expression of $f(\mathbf{u})$ in Equation 9, we have

$$f(\mathbf{t}(\mathcal{A} \cup e)) - f(\mathbf{t}(\mathcal{A})) = \sum_{i=1}^{n} K(x_i, x_t) \left(\Omega(\mathbf{t}^T(\mathcal{A} \cup e)\mathbf{t}(\mathcal{S}_i)) - \Omega(\mathbf{t}^T(\mathcal{A})\mathbf{t}(\mathcal{S}_i)) \right) \quad (11)$$

and

$$f(\mathbf{t}(\mathcal{B} \cup e)) - f(\mathbf{t}(\mathcal{B})) = \sum_{i=1}^{n} K(x_i, x_t) \left(\Omega(\mathbf{t}^T(\mathcal{B} \cup e)\mathbf{t}(\mathcal{S}_i)) - \Omega(\mathbf{t}^T(\mathcal{B})\mathbf{t}(\mathcal{S}_i)) \right) \quad (12)$$

Since $e \in \mathcal{N} \setminus \mathcal{B}$ and $\mathcal{A} \subseteq \mathcal{B}$, we have

$$\mathbf{t}(\mathcal{A} \cup e) = \mathbf{t}(\mathcal{A}) + \mathbf{t}(e),\ \mathbf{t}(\mathcal{B} \cup e) = \mathbf{t}(\mathcal{B}) + \mathbf{t}(e)$$

Thus,

$$\begin{aligned}(\mathbf{t}(\mathcal{A} \cup e) - \mathbf{t}(\mathcal{A}))^T \mathbf{t}(\mathcal{S}_i) &= (\mathbf{t}(\mathcal{B} \cup e) - \mathbf{t}(\mathcal{B}))^T \mathbf{t}(\mathcal{S}_i) \\ &= \mathbf{t}(e)^T \mathbf{t}(\mathcal{S}_i) \qquad (13)\end{aligned}$$

Furthermore, based on the property $\mathcal{A} \subseteq \mathcal{A} \cup e, \mathcal{B} \subseteq \mathcal{B} \cup e$, we have

$$\begin{aligned}\mathbf{t}(\mathcal{A})^T \mathbf{t}(\mathcal{S}_i) &\leq \mathbf{t}(\mathcal{A} \cup e)^T \mathbf{t}(\mathcal{S}_i), \mathbf{t}(\mathcal{B})^T \mathbf{t}(\mathcal{S}_i) \\ &\leq \mathbf{t}(\mathcal{B} \cup e)^T \mathbf{t}(\mathcal{S}_i). \qquad (14)\end{aligned}$$

Now, based on the properties in Equations 13 and 14, for any concave function $\Omega(x)$, we have

$$\begin{aligned}\Omega(\mathbf{t}(\mathcal{A})^T \mathbf{t}(\mathcal{S}_i)) + \Omega(\mathbf{t}(\mathcal{B} \cup e)^T \mathbf{t}(\mathcal{S}_i)) \leq \\ \Omega(\mathbf{t}(\mathcal{B})^T \mathbf{t}(\mathcal{S}_i)) + \Omega(\mathbf{t}(\mathcal{A} \cup e)^T \mathbf{t}(\mathcal{S}_i)) \qquad (15)\end{aligned}$$

The above inequality holds because of the following property of the concave function, i.e.,

$$\Omega(x) + \Omega(y) \leq \Omega(p) + \Omega(q)$$

if $x \leq p, q \leq y$ and $x + y = p + q$. By letting

$$\begin{aligned}x = \mathbf{t}(\mathcal{A})^T \mathbf{t}(\mathcal{S}_i), \quad y = \mathbf{t}(\mathcal{B} \cup e)^T \mathbf{t}(\mathcal{S}_i) \\ q = \mathbf{t}(\mathcal{B})^T \mathbf{t}(\mathcal{S}_i), \quad p = \mathbf{t}(\mathcal{A} \cup e)^T \mathbf{t}(\mathcal{S}_i)\end{aligned}$$

we have Equation 15.

Finally, substituting the inequality in Equation 15 into Equations 11 and 12, we obtain

$$f(\mathbf{t}(\mathcal{A} \cup e)) - f(\mathbf{t}(\mathcal{A})) \geq f(\mathbf{t}(\mathcal{B} \cup e)) - f(\mathbf{t}(\mathcal{B})).$$

References

[1] V. Athitsos, J. Alon, and S. Sclaroff. Efficient nearest neighbor classification using a cascade of approximate similarity measures. In *Proc. CVPR*, 2005.

[2] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proc. CIKM*, 2004.

[3] E. Chang, K. Goh, G. Sychay, and G. Wu. CBSA: 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.

[4] W. Chu and Z. Ghahramani. Preference learning with gaussian processes. In *Proc. ICML*, 2005.

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

[6] P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proc. ECCV*, 2002.

[7] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Proc. NIPS*, 2001.

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

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

[10] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *Proc. SIGIR*, 2003.

[11] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *Proc European Conference on Machine Learning*, 1998.

[12] T. Joachims. Making large-scale support vector machine learning practical. pages 169–184, 1999.

[13] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proc. SIGIR 2001*, 2001.

[14] C. Loader. *Local Regression and Likelihood*. Springer-Verlag, 1999.

[15] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *Proc. AAAI Workshop on Text Learning*, 1999.

[16] R. G. Parker. *Discrete Optimization*. Academic Press, 1988.

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

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

[19] S. N. S. Sung-Hyuk Cha. Nearest neighbor search using additive binary tree. In *Proc. CVPR*, 2000.

[20] J. Tesic and B. S. Manjunath. Nearest neighbor search for relevance feedback. In *Proc. CVPR*, 2003.

[21] N. Ueda and K. Saito. Parametric mixture models for multi-labeled text. In *Proc. NIPS*, 2002.

[22] G. M. Weiss and F. J. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *JAIR*, 19:315–354, 2003.

[23] C. Williams. Computation with infinite neural networks. *Neural Computation*, 10(5), 1998.

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

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

[26] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Information Systems*, 2(2), 2004.

[27] D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *Proc. NIPS*, 2005.

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

[29] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proc. ICML*, 2003.