2012 IEEE 12th International Conference on Data Mining

Hierarchical Multilabel Classification with Minimum Bayes Risk

Wei Bi, James T. Kwok
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
{weibi, jamesk@cse.ust.hk}

***Abstract*—Hierarchical multilabel classification (HMC) allows an instance to have multiple labels residing in a hierarchy. A popular loss function used in HMC is the H-loss, which penalizes only the first classification mistake along each prediction path. However, the H-loss metric can only be used on tree-structured label hierarchies, but not on DAG hierarchies. Moreover, it may lead to misleading predictions as not all misclassifications in the hierarchy are penalized. In this paper, we overcome these deficiencies by proposing a hierarchy-aware loss function that is more appropriate for HMC. Using Bayesian decision theory, we develop a Bayes-optimal classifier with respect to this loss function. Instead of requiring an exhaustive summation and search for the optimal multilabel, the proposed classification problem can be efficiently solved using a greedy algorithm on both tree- and DAG-structured label hierarchies. Experimental results on a large number of real-world data sets show that the proposed algorithm outperforms existing HMC methods.**

***Keywords*-hierarchical classification; multilabel classification; Bayesian decision theory**

I. Introduction

Multilabel classification, which allows an instance to have multiple labels, has been gaining a lot of interest in recent years. It has found successful applications in diverse domains including text classification [1], image annotation [2], video annotation [3], and bioinformatics [4]. Recent surveys on multilabel classification can be found in [5], [6].

To handle the myriad of labels, it is often useful to organize them into hierarchies. This can be achieved with the help of domain experts, or be automatically created from the data using procedures such as hierarchical clustering [7] or Bayesian network structure learning [8]. This whole collection of labels may then be arranged as a tree, as in text categorization [9], or more generally, in a directed acyclic graph (DAG), as in the Gene Ontology [10]. It is well-known that the use of this hierarchy information can boost classification performance [4], [9], [11]. While most of the current research efforts are concerned with the easier tree-structured hierarchies [5], recently more powerful algorithms that can be used on both tree- and DAG-structured hierarchies have also been proposed [4], [12], [13].

On the other hand, the loss function is of central importance in any classification problem. For multilabel classification, the most commonly used loss functions are the zero-one loss and Hamming loss [3], [14]. Usually, the Hamming loss is more informative, as is illustrated in Figure 1. Here, predictions A and B have the same zero-one loss, but A, which is intuitively less accurate, has a larger Hamming loss. For data sets with significant class imbalance, the F1-measure may be a better measure in balancing contributions of recall and precision [15]. Other loss functions, such as the rank loss, are particularly relevant in multilabel ranking [16]. Once a loss function is chosen, it is highly desirable to integrate this into the learning algorithm. Petterson and Caetano [17] proposed a reverse multilabel learning formulation in which macro-precision, macro-recall, macro-F_β and Hamming loss can be directly optimized. Lampert [18] derived a maximum-margin formulation that can be used with different loss functions. Some Bayes-optimal classifiers are also recently derived in [19], [20].

However, when used in hierarchical classification, a major

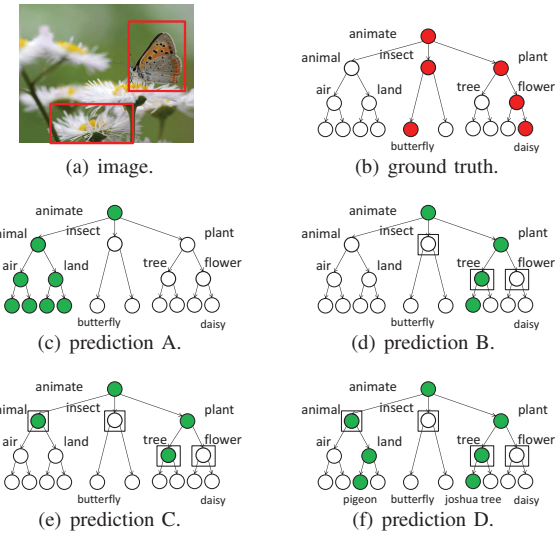
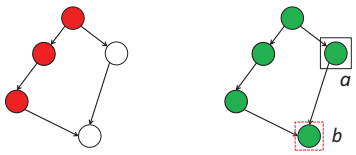


Figure 1. An example illustrating the deficiencies of the various loss functions. Please refer to the text for details.

1550-4786/12 $26.00 © 2012 Crown Copyright
DOI 10.1109/ICDM.2012.42

deficiency of these loss functions is that they do not incorporate hierarchy information. Referring back to Figure 1, predictions B and C differ in two nodes, and have the same Hamming loss. However, misclassifications at the upper hierarchy levels are often considered more important in hierarchical classification [9], [11], and so C should be inferior. To alleviate this problem, Cesa-Bianchi *et al.* [11] proposed the hierarchical loss (H-loss) that only counts the first classification mistake (shown as square in Figure 1) along a prediction path from the root to the labeled nodes. As can be seen from Figure 1, B is then considered better than C. Cesa-Bianchi *et al.* [21] further proposed the B-SVM algorithm that can produce Bayes-optimal decisions w.r.t. this H-loss. In [15], this is again extended to the cost-sensitive learning setting where false positives and false negatives are weighted differently.

However, the H-loss is not without its limitations. First, it can only be used on tree-structured label hierarchies. On DAGs, since the root may have multiple paths to a node, how to define the "first"classification mistake in the H-loss can be ambiguous (Figure 2). The second problem associated with the H-loss can be illustrated by predictions C and D in Figure 1. As can be seen, C and D have the same H-loss. However, since the H-loss only penalizes the first classification mistake, D further predicts (wrongly) that the image belongs to a specific animal (*pigeon*) and a specific tree (*joshua tree*), without incurring additional H-loss. On the other hand, C is more prudent and only predicts the image as belonging to *animal* and *tree*. Apparently, prediction D can be more misleading in many practical applications.



(a) ground truth. (b) prediction.

Figure 2. Ambiguity in defining the H-loss. Node a is always counted as a classification mistake, but node b is a classification mistake only if the left path is taken.

In this paper, we first introduce a loss function which is more appropriate for hierarchical multilabel classification (HMC). Similar to the H-loss, it weights the misclassifications according to their positions in the hierarchy, while avoiding the deficiencies of the H-loss discussed above. Next, using Bayesian decision theory, we derive the optimal prediction rule by minimizing the conditional risk with respect to the proposed loss. On both tree- and DAG-structured label hierarchies, the conditional risk can be efficiently computed and minimized by simple greedy algorithms, without the need for summing and searching over an exponential number of label combinations.

The rest of this paper is organized as follows. Section II briefly reviews some related work. Section III introduces the loss function to be used in this paper, and Section IV shows how the resultant conditional risk can be efficiently computed and minimized. Experimental results are presented in Section V. In the last section, we gives some concluding remarks. All the proofs are in the appendix.

Notations: In the following, $\mathcal{H}$ denotes the label hierarchy. Its nodes are indexed as 0 (for the root), $1, 2, \ldots, N-1$, where N is the number of nodes in $\mathcal{H}$. For a node i, we use $\text{pa}(i)$ to denote its (unique) parent when $\mathcal{H}$ is a tree, and $\text{Pa}(i)/\text{anc}(i)/\text{sibl}(i)$ to denote the set of its parent(s) / ancestors / siblings. Finally, $I(\cdot)$ is the indicator function that returns 1 when the argument holds, and 0 otherwise.

II. Related Work

A. Hierarchy Constraints in Hierarchical Classification

In hierarchical classification, we are given a set of $(\mathbf{x}, \mathbf{y})$'s, where $\mathbf{x}$ is the input and $\mathbf{y}$ is the multilabel $[y_0, \ldots, y_{N-1}]^T \in \{0,1\}^N$ denoting the memberships of $\mathbf{x}$ to each of the nodes in $\mathcal{H}$. The label hierarchy can be a tree, or more generally, an arbitrary DAG. For tree-structured label hierarchies, a node i (except the root) can be labeled positive only if its parent is also labeled positive, i.e.,

$$y_i = 1 \Rightarrow y_{\text{pa}(i)} = 1. \tag{1}$$

For DAG-structured label hierarchies, there are two interpretations of its hierarchy constraint [4], [13]. One is the AND-interpretation, which means that a node can be labeled positive only if all its parents are positive. The other is the OR-interpretation, which means a node can be labeled positive as long as one of its parents is positive. In this paper, we adopt the AND-interpretation which is more common. Thus, for each node i (except the root), we have

$$y_i = 1 \Rightarrow \mathbf{y}_{\text{Pa}(i)} = \mathbf{1}, \tag{2}$$

where $\mathbf{y}_{\text{Pa}(i)}$ is the subvector of $\mathbf{y}$ with indices from $\text{Pa}(i)$.

B. Hierarchical Multilabel Classification: CSSA

Recently, Bi and Kwok [13] proposed a novel hierarchical multilabel classification algorithm which can be used on both tree- and DAG-structured hierarchies. A key step is to find the multilabel $\hat{\mathbf{y}}$ that is (i) most similar to a crudely estimated multilabel $\tilde{\mathbf{y}}$; (ii) consistent with the label hierarchy; and (iii) has a pre-determined number of nodes (say, L) predicted positive. For the label tree, they formulated this as

the following optimization problem:

$$\max_{\{\psi_i\}_{i\in\mathcal{H}}} \quad \sum_{i\in\mathcal{H}} \psi_i \tilde{y}_i \tag{3}$$

$$\text{s.t.} \quad \psi_i \le \psi_{\text{pa}(i)} \quad \forall i \in \mathcal{H}\backslash\{0\}, \tag{4}$$

$$\psi_0 = 1,\ \psi_i = \{0,1\}, \tag{5}$$

$$\sum_{i=0}^{N-1} \psi_i = L. \tag{6}$$

Here, ψ_i is a binary indicator such that $\psi_i = 1$ denotes that node i is predicted positive in $\hat{\mathbf{y}}$; and 0 otherwise. Constraint (4) encodes the hierarchy constraint in (1); while constraint (6) requires that L nodes are predicted positive. Bi and Kwok [13] showed that problem (3) can be solved efficiently, in $O(N \log N)$ time, via a greedy algorithm called *Condensing Sort and Selection Algorithm* (CSSA), which is originally used in signal processing [22].

When the label hierarchy is a DAG, one only has to replace constraint (4), which is used to encode the hierarchy constraint for label trees, to

$$\psi_i \le \psi_j \quad \forall i \in \mathcal{H}\backslash\{0\}, \forall j \in \text{Pa}(i), \tag{7}$$

which corresponds to the hierarchy constraint in (2). The resultant optimization problem can also be solved efficiently by a DAG extension of CSSA (called CSSAG). For more details, interested readers are referred to [13].

As mentioned above, both CSSA and CSSAG require the user to pre-determine the number of positive labels L in $\hat{\mathbf{y}}$. Moreover, no loss function is explicitly considered in their formulation.

III. Hierarchical Loss Function

A. Definition

For a pattern $\mathbf{x}$, let $\mathbf{y}$ be its ground truth multilabel and $\hat{\mathbf{y}}$ the predicted multilabel. As discussed in Section I, the decision on each label node represents the classifier's cognition on that label. Thus, every mistake on any node should be taken into account. In this paper, we introduce the following loss function, which will be called HMC-loss in the sequel.

$$\ell(\hat{\mathbf{y}}, \mathbf{y}) = \alpha \sum_{i:y_i=1\wedge\hat{y}_i=0} c_i + \beta \sum_{i:y_i=0\wedge\hat{y}_i=1} c_i. \tag{8}$$

Here, each misclassified node i incurs a fixed cost $c_i \ge 0$. The first term corresponds to the loss incurred due to false negatives, while the second term is due to the false positives; and $\alpha, \beta \ge 0$ weight the false negatives/positives differently. Cai and Hofmann [14] defined a similar loss function, but only in the context of hierarchical multiclass classification.

The hierarchy information can be incorporated into $\ell(\hat{\mathbf{y}}, \mathbf{y})$ via the setting of c_i's. In HMC, misclassifications at the upper hierarchy levels (which correspond to more generic concepts) are often treated as more expensive than those at the lower levels (which correspond to more specific concepts). Thus, we can incorporate the structure information by penalizing the upper-level misclassified nodes more heavily. When the label hierarchy is a tree, this can be achieved by following the H-loss in [9] and define c_i's as

$$c_i = \begin{cases} 1 & i = 0 \text{ (the root)} \\ \frac{c_{pa(i)}}{n_{\text{sibl}}(i)} & i > 0 \end{cases}, \tag{9}$$

where $n_{\text{sibl}}(i)$ is the number of siblings of i (including i). Intuitively, the penalty cost associated with a parent node is equally shared by all its children. More generally, when the label hierarchy forms a DAG, we extend (9) as

$$c_i = \begin{cases} 1 & i = 0 \\ \sum_{j\in\text{Pa}(i)} \frac{c_j}{n_{\text{child}}(j)} & i > 0 \end{cases}. \tag{10}$$

where $n_{\text{child}}(j)$ is the number of children of j.

B. Special Cases

Note that the proposed HMC-loss in (8) is quite flexible. Even for the special case where all c_i's are 1, it already encompasses a number of loss functions popularly used in both flat and hierarchical classification, including:

- $\alpha = \beta = 1$: $\ell(\hat{\mathbf{y}}, \mathbf{y})$ then reduces to the exclusive OR of $\hat{\mathbf{y}}$ and $\mathbf{y}$. This is often called the Hamming loss (or symmetric loss), and is the most frequently used loss function in multilabel classification [19].
- Suppose that a given number of labels are to be predicted, i.e., the size of the support[1] of $\hat{\mathbf{y}}$, $|\text{supp}(\hat{\mathbf{y}})|$, is a constant. On setting $\alpha = 0, \beta = 1/|\text{supp}(\hat{\mathbf{y}})|$,

  $$\ell(\hat{\mathbf{y}}, \mathbf{y}) = 1 - \frac{|\text{supp}(\hat{\mathbf{y}} \wedge \mathbf{y})|}{|\text{supp}(\hat{\mathbf{y}})|} = 1 - \frac{\sum_i I(\hat{y}_i = y_i = 1)}{\sum_i I(\hat{y}_i = 1)}.$$

  Note that $\sum_i I(\hat{y}_i = y_i = 1)$ is the number of true positives, and $\sum_i I(\hat{y}_i = 1)$ is the number of predicted positives. Hence, $\ell(\hat{\mathbf{y}}, \mathbf{y})$ becomes $1 -$ precision,[2] and minimizing $\ell(\hat{\mathbf{y}}, \mathbf{y})$ becomes maximizing precision.
- For the special case of hierarchical multiclass classification, both $\hat{\mathbf{y}}$ and $\mathbf{y}$ consist of one single path. On setting $\alpha = 0$ and $\beta = 1$, $\ell(\hat{\mathbf{y}}, \mathbf{y})$ becomes the height of the lowest common ancestor between $\hat{\mathbf{y}}$ and $\mathbf{y}$ (Figure 3), which is similar to the hierarchical cost advocated in [23]. On the other hand, if we set $\alpha = \beta = 1$, $\ell(\hat{\mathbf{y}}, \mathbf{y})$ becomes the length of the path between the most specific nodes of $\hat{\mathbf{y}}$ and $\mathbf{y}$ minus 1 (as the lowest common ancestor of $\hat{\mathbf{y}}$ and $\mathbf{y}$ is correctly classified, and thus not counted in $\ell(\hat{\mathbf{y}}, \mathbf{y})$). This is similar to the hierarchical cost used in [24].

[1] The support of a vector $\mathbf{y}$ is $\text{supp}(\mathbf{y}) = \{i : y_i \neq 0\}$.

[2] Precision $= \frac{\text{number of true positives}}{\text{number of true positives+number of false positives}}$.

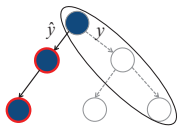


Figure 3. When $\alpha = 0$ and $\beta = 1$, $\ell(\hat{\mathbf{y}}, \mathbf{y})$ (circled in red) measures the height of the lowest common ancestor (which is equal to 2 in this example).

IV. Minimizing the Risk

Given a loss function $\ell(\hat{\mathbf{y}}, \mathbf{y})$ and a test pattern $\mathbf{x}$, the conditional risk (or expected loss) $\mathcal{R}(\hat{\mathbf{y}})$ is defined as the expectation of $\ell(\hat{\mathbf{y}}, \mathbf{y})$ over all possible $\mathbf{y}$'s:

$$\mathcal{R}(\hat{\mathbf{y}}) = \sum_{\mathbf{y}} \ell(\hat{\mathbf{y}}, \mathbf{y}) P(\mathbf{y}|\mathbf{x}). \tag{11}$$

Here, $P(\mathbf{y}|\mathbf{x})$'s are either known or, more typically, estimated from the data. As in [11], we assume that the labels of a group of sibling nodes in the label hierarchy are conditionally independent given their parent label(s). This simplification is standard in Bayesian networks and also commonly used in HMC [25], [26]. Thus, for the tree label hierarchy, we have

$$P(\mathbf{y}|\mathbf{x}) = \prod_{i \in \mathcal{H} \setminus \{0\}} P(y_i | y_{\text{pa}(i)}, \mathbf{x}). \tag{12}$$

Moreover, $P(y_i = 1 | y_{\text{pa}(i)} = 0, \mathbf{x}) = 0$ as such a label combination violates the hierarchy constraint (1). Similarly, for a DAG label hierarchy, we have

$$P(\mathbf{y}|\mathbf{x}) = \prod_{i \in \mathcal{H} \setminus \{0\}} P(y_i | \mathbf{y}_{\text{Pa}(i)}, \mathbf{x}), \tag{13}$$

and $P(y_i = 1 | \mathbf{y}_{\text{Pa}(i)}, \mathbf{x}) = 0$ if $y_j = 0$ for any $j \in \text{Pa}(i)$. With this simplification, we only need to train estimators for $p(y_i = 1 | y_{\text{pa}(i)} = 1, \mathbf{x})$ (or $p(y_i = 1 | \mathbf{y}_{\text{Pa}(i)} = \mathbf{1}, \mathbf{x})$) for each node i, using methods such as logistic regression or support vector machines. The algorithm to be proposed is nevertheless independent of the way these probability estimators are learned.

From Bayesian decision theory [27], the optimal $\hat{\mathbf{y}}^*$ is the one that minimizes the risk, i.e.,

$$\begin{aligned} \hat{\mathbf{y}}^* = \arg \quad & \min_{\hat{\mathbf{y}}} \quad \mathcal{R}(\hat{\mathbf{y}}) \\ & \text{s.t.} \quad \hat{\mathbf{y}} \text{ satisfies the hierarchy constraint.} \end{aligned} \tag{14}$$

Obviously, the two key issues in obtaining $\hat{\mathbf{y}}^*$ are

1) How to efficiently compute $\mathcal{R}(\hat{\mathbf{y}})$ for a particular $\hat{\mathbf{y}}$, without exhaustively summing all the 2^N combinations of $\mathbf{y}$'s in (11)?
2) How to efficiently minimize $\mathcal{R}(\hat{\mathbf{y}})$, without exhaustively enumerating all the 2^N possible combinations of $\hat{\mathbf{y}}$?

We will address these in the following sections.

A. Efficient Computation of $\mathcal{R}(\hat{\mathbf{y}})$

Let p_i be the probability that node i is positive given $\mathbf{x}$. By the hierarchy constraint in (1) or (2), all the ancestors of i must also be positive. Hence, for a valid multilabel $\mathbf{y}$,

$$p_i \equiv P(y_i = 1|\mathbf{x}) = P(y_i = 1, \mathbf{y}_{\text{anc}(i)} = \mathbf{1}|\mathbf{x}). \tag{15}$$

The following proposition shows that the risk $\mathcal{R}(\hat{\mathbf{y}})$ in (11) can be easily computed from these p_i's for both tree- and DAG-structured label hierarchies. The proof can be found in Appendix A.

Proposition 1.

$$\mathcal{R}(\hat{\mathbf{y}}) = \alpha \sum_{i:\hat{y}_i=0} c_i p_i + \beta \sum_{i:\hat{y}_i=1} c_i (1 - p_i). \tag{16}$$

The first term on the right is due to contributions from the false negatives, while the second term is due to the false positives. Given the p_i's, $\mathcal{R}(\hat{\mathbf{y}})$ can be computed in $O(N)$ time.

The p_i's can be efficiently computed. First, consider a label tree. On using (12), we have

$$\begin{aligned} p_i &= P(y_i = 1 | y_{\text{pa}(i)} = 1, \mathbf{x}) \prod_{j \in \text{anc}(i) \setminus \{0\}} P(y_j = 1 | y_{\text{pa}(j)} = 1, \mathbf{x}) \\ &= P(y_i = 1 | y_{\text{pa}(i)} = 1, \mathbf{x}) p_{\text{pa}(i)}. \end{aligned}$$

Thus, one can recursively obtain all p_i's by starting from the root (with $p_0 = 1$ as the root is always labeled positive) and traverse the tree with breadth-first-search (BFS) or depth-first-search (DFS). The total time is $O(N)$.

For a label DAG, we have, on using (13),

$$p_i = P(y_i = 1 | \mathbf{y}_{\text{Pa}(i)} = \mathbf{1}, \mathbf{x}) \prod_{j \in \text{anc}(i) \setminus \{0\}} P(y_j = 1 | \mathbf{y}_{\text{Pa}(j)} = \mathbf{1}, \mathbf{x}).$$

Observe that $\text{anc}(i) = \text{Pa}(i) \cup \{\text{anc}(j)\}_{j \in \text{Pa}(i)}$. Hence, the $\text{anc}(i)$ sets for all i's can be obtained recursively by traversing the hierarchy using BFS, which takes $O(N + E)$ time (E is the number of edges in the hierarchy). Moreover, note that all these $\text{anc}(i)$'s only need to be computed once and then stored, as part of preprocessing. With $\text{anc}(i)$, each p_i can be computed in $O(|\text{anc}(i)|)$ time. To obtain an upper bound on the total time $\sum_{i=1}^{N} |\text{anc}(i)|$, consider visiting the nodes in a certain topological order. For node i, $|\text{anc}(i)|$ is upper-bounded by the number of nodes placed before it in this order. Hence, $\sum_{i=1}^{N} |\text{anc}(i)| \leq \sum_{i=1}^{N} i = O(N^2)$.

B. Efficient Minimization of $\mathcal{R}(\hat{\mathbf{y}})$

Next, we consider how to find the $\hat{\mathbf{y}}^*$ that minimizes $\mathcal{R}(\hat{\mathbf{y}})$ in (16). First, notice that problem (14) can be decomposed into N subproblems, as

$$\hat{\mathbf{y}}^* = \arg \min_{L=1,\ldots,N} \mathcal{R}(\hat{\mathbf{y}}^*_{(L)}), \tag{17}$$

where each subproblem finds the optimal multilabel with L nodes labeled positive:

$$\hat{\mathbf{y}}^*_{(L)} = \arg \quad \min_{\hat{\mathbf{y}}} \quad \mathcal{R}(\hat{\mathbf{y}}) \qquad (18)$$
$$\text{s.t.} \quad |\text{supp}(\hat{\mathbf{y}})| = L,$$
$$\hat{\mathbf{y}} \text{ satisfies the hierarchy constraint.}$$

The following proposition shows how we can obtain $\hat{\mathbf{y}}^*_{(L)}$ given the p_i's. Proof can be found in Appendix B. It can be seen that δ_{n_i} in (19) is the reduction in risk by predicting node n_i positive in the multilabel.

Proposition 2. *The L nodes that are labeled positive in $\hat{\mathbf{y}}^*_{(L)}$ can be obtained as*

$$\arg \max_{\{n_1,n_2,\ldots,n_L\}\subset\mathcal{H}} \sum_{i=1}^{L} \delta_{n_i}, \qquad (19)$$

where

$$\delta_i = c_i \left(\alpha p_i - \beta(1-p_i)\right). \qquad (20)$$

To solve (19), we associate a binary indicator $\psi_i \in \{0,1\}$ with each node i, such that $\psi_i = 1$ denotes that node i is selected by $\hat{\mathbf{y}}^*_{(L)}$, and 0 otherwise. The objective can then be written as $\sum_{i=0}^{N-1} \psi_i \delta_i$. Moreover, the hierarchy constraints can be enforced by adding constraints (4) for label trees, or (7) for label DAGs. Thus, we obtain the following reformulations of (18).

Corollary 1. *For a label tree, problem (18) can be reformulated as the following problem:*

$$\max_{\{\psi_i\}_{i\in\mathcal{H}}} \quad \sum_i \psi_i \delta_i \qquad (21)$$
$$\textit{s.t.} \quad \psi_i \le \psi_{pa(i)} \quad \forall i \in \mathcal{H}\backslash\{0\},$$
$$\psi_0 = 1,\ \psi_i = \{0,1\},\ \sum_{i=0}^{N-1} \psi_i = L.$$

Corollary 2. *For a label DAG, problem (18) can be reformulated as the following problem:*

$$\max_{\{\psi_i\}_{i\in\mathcal{H}}} \quad \sum_i \psi_i \delta_i \qquad (22)$$
$$\textit{s.t.} \quad \psi_i \le \psi_j \quad \forall i \in \mathcal{H}\backslash\{0\}, \forall j \in \textit{Pa}(i),$$
$$\psi_0 = 1,\ \psi_i = \{0,1\},\ \sum_{i=0}^{N-1} \psi_i = L.$$

Interestingly, both (21) and (22) are of the same form as the optimization problem in (3), except that $\tilde{y}_i$ in (3) is now replaced by δ_i. Thus, we can reuse the efficient CSSA (resp. CSSAG) algorithm in [13] for the tree (resp. DAG) label hierarchy.

Recall from (17) that we need to first compute the risks for $L = 1, \ldots, N$ using the above procedure, and then pick the L with the smallest risk. A straightforward procedure is to run CSSA or CSSAG $N-1$ times (the case for $L = 1$ trivially yields the multilabel with only the root labeled positive). However, since CSSA/CSSAG is a greedy algorithm, by the optimal substructure property, the optimal solution of size L_1 contains all the optimal solutions of sizes $L_2 < L_1$. Thus, we can simply set $L = N$, and keep track of the optimal solution obtained for each intermediate value of L. The total time to obtain the optimal solution (with the L value yielding the smallest risk) is then still $O(N \log N)$.

The complete algorithms, which will be called HIROM (HIerarchical Risk-Optimizing Multilabel classification), for tree- and DAG-structured label hierarchies are shown in Algorithm 1 and 2, respectively. Recall that CSSA/CSSAG in [13] requires as input the number of labels (L) to be predicted. On the other hand, HIROM can automatically determine the value of L.

Algorithm 1 HIROM for tree label hierarchies.

1: Traverse the tree $\mathcal{H}$ using BFS or DFS to compute p_i for all $i \in \mathcal{H}$.
2: Compute δ_i for all $i \in \mathcal{H}$ using (20).
3: Use the CSSA algorithm in [13] with the computed δ_i's, and obtain $\{\hat{\mathbf{y}}^*_{(1)}, \ldots, \hat{\mathbf{y}}^*_{(N)}\}$.
4: $\hat{\mathbf{y}}^* \leftarrow \arg\min_{\hat{\mathbf{y}}^*_{(L)}} \mathcal{R}(\hat{\mathbf{y}}^*_{(L)})$.

Algorithm 2 HIROM for DAG label hierarchies.

1: Perform BFS and obtain anc(i)'s for all $i \in \mathcal{H}$.
2: Compute all p_i's with anc(i)'s.
3: Compute δ_i for all $i \in \mathcal{H}$ using (20).
4: Use the CSSAG algorithm in [13] with the computed δ_i's, and obtain $\{\hat{\mathbf{y}}^*_{(1)}, \ldots, \hat{\mathbf{y}}^*_{(N)}\}$.
5: $\hat{\mathbf{y}}^* \leftarrow \arg\min_{\hat{\mathbf{y}}^*_{(L)}} \mathcal{R}(\hat{\mathbf{y}}^*_{(L)})$.

V. Experiments

A. Setup

In this section, we perform experiments on a large number of real-world data sets commonly used in multilabel classification. These include 21 data sets with tree-structured label hierarchies (Table I):

- Five of them are subsets of RCV1v2 [28].[3] These contain documents in the REUTERS topics categories. The label hierarchy has 104 nodes constructed from the REUTERS topics categories.
- Enron [29]:[4] This is a text data set for email analysis, and its label hierarchy describes the relationships of the email users.
- imageClef07A and imageClef07D:[5] These are image data sets from the ImageCLEF 2007 competition. Both have the same set of features extracted from X-ray

[3] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html
[4] http://www.cs.cmu.edu/~enron/
[5] http://ir.ohsu.edu/image/2007data.html

images, but with output labels coming from different hierarchies.

- Caltech101 [30]:[6] This is another image data set on object annotation, with 101 label classes. We use the label hierarchy in the related Caltech256 data set [31], and take the 101 classes as leaf nodes. The total number of nodes in the resultant hierarchy is increased to 143.
- Twelve genomic data sets [4]:[7] These contain different aspects of genes in the yeast genome, with annotations from MIPS's Functional Catalog (Funcat).

Table I
TREE-STRUCTURED DATA SETS USED IN THE EXPERIMENTS.

data set	#training	#test	dim	#label
rcv1v2 subset1	3,000	3,000	47,236	104
rcv1v2 subset2	3,000	3,000	47,236	104
rcv1v2 subset3	3,000	3,000	47,236	104
rcv1v2 subset4	3,000	3,000	47,236	104
rcv1v2 subset5	3,000	3,000	47,236	104
imageclef07a	10,000	1,006	80	97
imageclef07d	10,000	1,006	80	47
enron	988	660	1,001	57
caltech101	4,572	4,572	21504	143
seq	2,580	1,339	489	500
pheno	1,009	582	170	456
struc	2,978	1,313	19,629	500
hom	2,539	1,315	47,035	500
cellcycle	2,909	1,281	77	500
church	2,911	1,281	26	500
derisi	2,450	1,275	63	500
eisen	1,587	837	79	462
gasch1	2,480	1,284	173	500
gasch2	2,488	1,291	52	500
spo	2,437	1,266	83	500
expr	2,488	1,291	551	500

For the DAG-structured data sets[8] (Table II), they are the same set of genomic data sets, but the patterns are annotated from the Gene Ontology (GO) [4]. There are three subgraphs in its label DAG, and the first subgraph is used in the experiment. For pre-processing, as in [15], we remove labels with fewer than 10 positive training instances.

Each of the constituent probabilities in $P(\mathbf{y}|\mathbf{x})$ (namely, $P(y_i = 1|y_{\text{pa}(i)} = 1)$ in (12) for tree hierarchies, and $P(y_i = 1|\mathbf{y}_{\text{Pa}(i)} = \mathbf{1})$ in (13) for DAG hierarchies) are obtained from a SVM. Specifically, for each node i, we first train a binary SVM, using as training examples those patterns that the parent(s) of i is labeled positive [21]. Then, we convert the SVM output to a probability estimate using the procedure in [32]. The SVM parameters are tuned by putting aside one third of the training set as validation set. We use the linear SVM with the C parameter chosen from the set $\{2^{-10}, 2^{-9}, \ldots, 1, \ldots, 2^9, 2^{10}\}$.

[6]http://www.vision.caltech.edu/Image_Datasets/Caltech101/
[7]http://dtai.cs.kuleuven.be/clus/hmcdatasets/
[8]http://dtai.cs.kuleuven.be/clus/hmcdatasets/

Table II
DAG-STRUCTURED DATA SETS USED IN THE EXPERIMENTS.

data set	#training	#test	dim	#label
seq	2,517	1,311	489	143
pheno	983	573	170	58
struc	2,467	1,285	19,629	142
hom	2,477	1,289	47,035	140
cellcycle	2,422	1,258	77	141
church	2,420	1,258	26	141
derisi	2,396	1,252	63	141
eisen	1,580	834	79	101
gasch1	2,426	1,261	173	141
gasch2	2,433	1,268	52	141
spo	2,383	1,243	83	139
expr	2,434	1,268	551	141

The proposed HIROM algorithm will be compared with the following methods:

1) CSSA (resp. CSSAG) for data sets with tree-structured (resp. DAG-structured) label hierarchies [13]: As discussed in Section II-B, this is most similar to the proposed algorithm, except that the loss function is not used in the formulation of CSSA. Recall that CSSA requires as input the number of labels to be predicted (L). Here, we first run HIROM and use the number of labels obtained as input to CSSA.
2) H-SVM [11]: This trains a binary classifier at each node. On prediction, if $p(y_i = 1|y_{\text{pa}(i)} = 1, \mathbf{x})$ or $p(y_i = 1|\mathbf{y}_{\text{Pa}(i)} = \mathbf{1}, \mathbf{x})$ is ≥ 0.5, node i is predicted positive and the process continues to its children.
3) B-SVM [15]: We use the cost-sensitive extension of the B-SVM in [21]. It is optimal w.r.t. the H-loss, and can only be used on tree-structured label hierarchies.

For the label tree, we set the c_i of each node using (9) for HIROM and B-SVM. For the label DAG, we use (10) for HIROM. Moreover, as in [15], we set $\alpha = \lambda \cdot \beta$ while keeping $\alpha + \beta = 2$, where λ is a parameter that balances the misclassification cost between positive and negative examples.

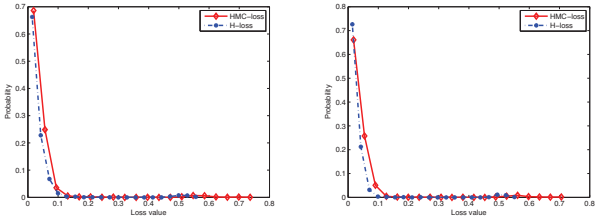
B. Classification Performance

In this experiment, we set $\lambda = \frac{n_-}{n_+}$, where n_- (resp. n_+) is the number of negative (resp. positive) training labels. This is a standard setting used in cost-sensitive learning [15]. Table III shows the HMC-loss values (averaged over the test samples) on the various tree-structured data sets. As can be seen, HIROM achieves the smallest loss as expected. Note that the B-SVM, though it minimizes the H-loss instead of the HMC-loss, is also quite competitive. This is because when the misclassification happens near the bottom of the hierarchy, the difference between HMC-loss and H-loss can become very small (Figure 4). Hence, B-SVM also approximately minimizes the HMC-loss.

Table IV shows the testing losses on the DAG-structured data sets. Again, HIROM consistently outperforms CSSA

Table III
TESTING HMC-LOSS VALUES ON THE TREE-STRUCTURED DATA SETS.

data set	HIROM	CSSA	H-SVM	B-SVM
rcv1v2 subset1	**0.040**	0.064	0.113	0.042
rcv1v2 subset2	**0.040**	0.057	0.099	0.043
rcv1v2 subset3	**0.043**	0.060	0.109	0.046
rcv1v2 subset4	**0.042**	0.064	0.116	0.045
rcv1v2 subset5	**0.042**	0.062	0.110	0.046
imageclef07a	**0.124**	0.166	0.217	0.132
imageclef07d	**0.045**	0.088	0.095	0.048
enron	**0.146**	0.149	0.304	0.158
caltech101	**0.051**	0.096	0.183	0.057
seq	**0.084**	0.251	0.350	0.091
pheno	**0.104**	0.309	0.406	0.105
struc	**0.090**	0.263	0.368	0.093
hom	**0.084**	0.223	0.324	0.090
cellcycle	**0.089**	0.257	0.384	0.097
church	**0.092**	0.276	0.391	0.099
derisi	**0.090**	0.266	0.390	0.098
eisen	**0.094**	0.265	0.373	0.100
gasch1	**0.086**	0.242	0.363	0.095
gasch2	**0.089**	0.255	0.377	0.101
spo	**0.090**	0.281	0.387	0.099
expr	**0.085**	0.236	0.358	0.094



(a) HIROM solution. (b) B-SVM solution.

Figure 4. Distributions of the values of HMC-loss and H-loss, as obtained by HIROM and B-SVM.

and H-SVM. Note that the bottom-up strategy used in B-SVM cannot be extended to handle DAG label structures.

To further illustrate the difference between HIROM and B-SVM, Figure 5 shows the predictions of two test samples from the Caltech101 data set. On both samples, HIROM yields a smaller HMC-loss but a larger H-loss than B-SVM. Apparently, though both algorithms cannot obtain the correct solution, the prediction of HIROM (which minimizes the HMC-loss) is still more informative that of B-SVM (which minimizes the H-loss), and has fewer misclassifications at the more specific (i.e., lower) levels of the hierarchy. This echoes our discussion in the previous sections that the HMC-loss can be a better measure than the H-loss in HMC.

C. Variation in Misclassification Costs

In this experiment, we vary the relative misclassification cost of positive and negative examples, λ, from $\{\frac{1}{10}, \frac{1}{9}, \ldots, \frac{1}{2}, 1, 2, \ldots, 9, 10\}$, and compare the performance of HIROM, CSSA, H-SVM and B-SVM. Note that only

Table IV
TESTING HMC-LOSS VALUES ON THE DAG-STRUCTURED DATA SETS.

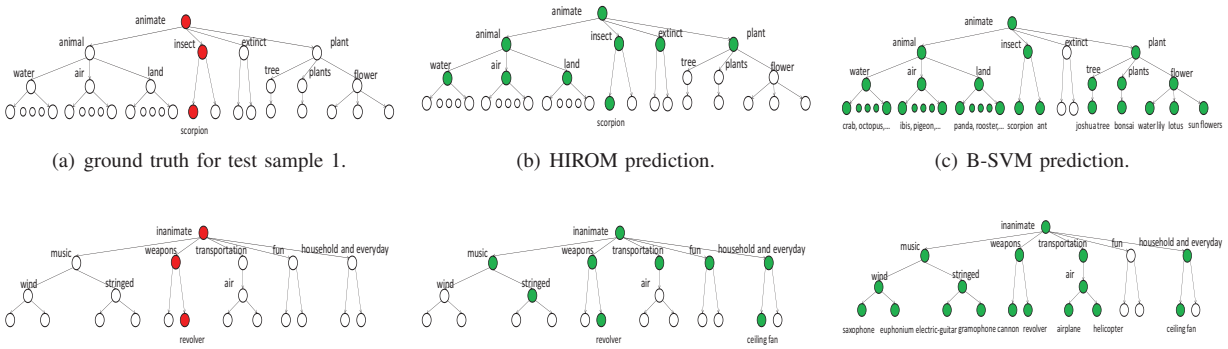
data set	HIROM	CSSAG	H-SVM
seq	**0.11**	0.31	2.27
pheno	**0.21**	0.27	2.18
struc	**0.13**	0.35	2.26
hom	**0.10**	0.35	2.27
cellcycle	**0.12**	0.24	2.26
church	**0.13**	0.25	2.26
derisi	**0.12**	0.24	2.26
eisen	**0.14**	0.24	2.25
gasch1	**0.12**	0.23	2.26
gasch2	**0.12**	0.23	2.26
spo	**0.13**	0.24	2.26
expr	**0.12**	0.23	2.26

HIROM and B-SVM are cost-sensitive and thus depend on the setting of λ.

Figure 6 shows the false positive rate $\text{FP} = \frac{1}{n_{\text{test}}} \sum_{i:y_i=0 \wedge \hat{y}_i=1} c_i$ (where n_{test} is the number of test patterns), the false negative rate $\text{FN} = \frac{1}{n_{\text{test}}} \sum_{i:y_i=1 \wedge \hat{y}_i=0} c_i$, and the HMC-loss. Because of the lack of space, only results on some data sets are reported. As can be seen, when α is increased, false negatives are penalized more heavily while false positives are penalized less. Thus, HIROM's FP increases and FN decreases, and vice versa. Since H-SVM is not cost-sensitive, its FP and FN remain constant over the whole range of λ. CSSA is also cost-insensitive. However, as we use the number of labels (L) obtained by HIROM as input to CSSA, its performance also changes with λ. As for the HMC-loss, recall that it is a weighted sum of FP and FN. With increasing α, the HMC-loss obtained by HIROM rises and then drops. Overall, HIROM outperforms the other methods, with B-SVM closely trailing behind (because it approximately minimizes the HMC-loss as discussed in Section V-B). However, note again that B-SVM cannot be used on the more complicated DAG-structured label hierarchies.

VI. CONCLUSION

In this paper, we proposed a new loss function (HMC-loss) for hierarchical multilabel classification. It can trade off false negatives and false positives, and can also weight the classification errors differently according to the label hierarchy. Compared to the traditional H-loss, the HMC-loss is more informative and can be used on both tree- and DAG-structured label hierarchies. Following the Bayesian decision theory, we further developed a Bayes-optimal classifier that minimizes the expectation of this HMC-loss. Both the computation and minimization of the risk can be efficiently obtained without exhaustive enumeration on an exponential number of possible multilabels. Experimental results on a large number of real-world data sets with both tree- and DAG-structured hierarchies demonstrate that the proposed



(a) ground truth for test sample 1. (b) HIROM prediction. (c) B-SVM prediction.

(d) ground truth for test sample 2. (e) HIROM prediction. (f) B-SVM prediction.

Figure 5. Two test samples from the Caltech101 data set, and the corresponding predictions by HIROM and B-SVM.

algorithm outperforms existing hierarchical multilabel classification methods.

ACKNOWLEDGMENT

This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant 614012).

REFERENCES

[1] R. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2, pp. 135–168, 2000.

[2] 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.

[3] B. Hariharan, L. Zelnik-Manor, S. Vishwanathan, and M. Varma, "Large scale max-margin multi-label classification with priors," in *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[4] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Machine Learning*, vol. 73, no. 2, pp. 185–214, 2008.

[5] C. Silla and A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 1–42, Jan. 2010.

[6] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds., 2010, pp. 667–685.

[7] K. Punera, S. Rajan, and J. Ghosh, "Automatically learning document taxonomies for hierarchical classification," in *Proceedings of the 14th International Conference on World Wide Web*, 2005.

[8] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, 2010, pp. 999–1008.

[9] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor, "Kernel-based learning of hierarchical multilabel classification models," *Journal of Machine Learning Research*, vol. 7, pp. 1601–1626, 2006.

[10] A. Clare, "Machine learning and data mining for yeast functional genomics," Ph.D. dissertation, University of Wales, 2003.

[11] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Incremental algorithms for hierarchical classification," *Journal of Machine Learning Research*, vol. 7, pp. 31–54, 2006.

[12] Z. Barutcuoglu and O. Troyanskaya, "Hierarchical multi-label prediction of gene function," *Bioinformatics*, vol. 22, pp. 830–836, 2006.

[13] W. Bi and J. Kwok, "Multi-label classification on tree- and DAG-structured hierarchies," in *Proceedings of the 28th International Conference on Machine Learning*, 2011, pp. 17–24.

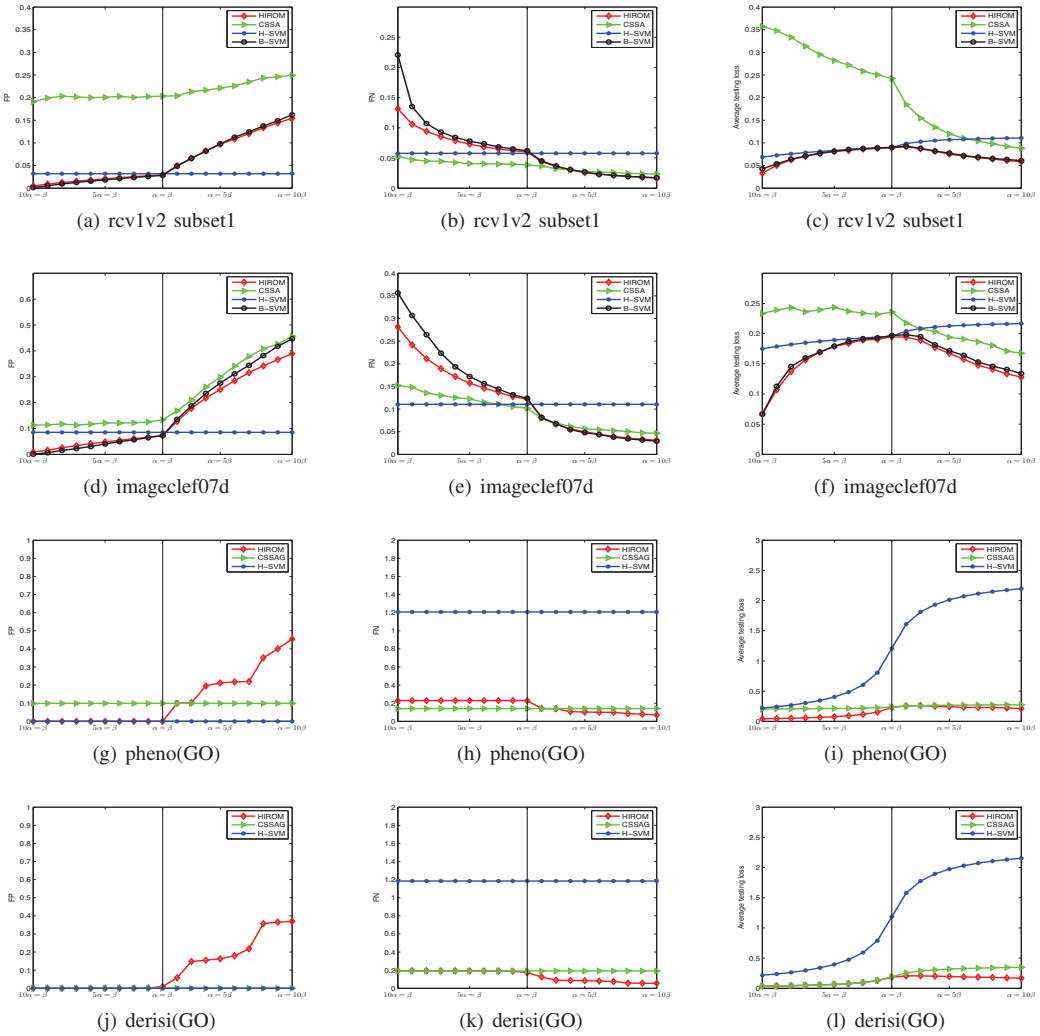
[14] L. Cai and T. Hofmann, "Hierarchical document categorization with support vector machines," in *Proceedings of the 13th International Conference on Information and Knowledge Management*, 2004, pp. 78–87.

[15] N. Cesa-Bianchi and G. Valentini, "Hierarchical cost-sensitive algorithms for genome-wide gene function prediction," in *Proceedings of the Third International Workshop on Machine Learning in Systems Biology*, 2009, pp. 14–29.

[16] K. Brinker and E. Hüllermeier, "Case-based multilabel ranking," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 702–707.

[17] J. Petterson and T. Caetano, "Reverse multi-label learning," in *Advances in Neural Information Processing Systems 23*, 2010.

[18] C. Lampert, "Maximum margin multi-label structured prediction," in *Advances in Neural Information Processing Systems 24*, 2011.



(a) rcv1v2 subset1 (b) rcv1v2 subset1 (c) rcv1v2 subset1

(d) imageclef07d (e) imageclef07d (f) imageclef07d

(g) pheno(GO) (h) pheno(GO) (i) pheno(GO)

(j) derisi(GO) (k) derisi(GO) (l) derisi(GO)

Figure 6. FP (left), FN (center) and HMC-loss (right) obtained by the various methods at different weightings of α and β. The label hierarchies for the top two data sets are tree-structured, while those for the bottom two are DAG-structured.

[19] K. Dembczynski, W. Cheng, and E. Hüllermeier, "Bayes optimal multilabel classification via probabilistic classifier chains," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 279–286.

[20] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hüllermeier, "An exact algorithm for F-measure maximization," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 223–230.

[21] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Hierarchical classification: Combining Bayes with SVM," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 177–184.

[22] R. Baraniuk and D. Jones, "A signal-dependent time-frequency representation: Fast algorithm for optimal kernel design," *IEEE Transactions on Signal Processing*, vol. 42, no. 1, pp. 134–146, 1994.

[23] J. Deng, A. Berg, K. Li, and F.-F. Li, "What does classifying more than 10,000 image categories tell us?" in *Proceedings of the 11th European Conference on Computer Vision*, 2010, pp. 71–84.

[24] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proceedings of the 21st International Conference on Machine learning*, 2004.

[25] N. Cesa-Bianchi, C. Gentile, A. Tironi, and L. Zaniboni, "Incremental algorithms for hierarchical classification," in *Advances in Neural Information Processing Systems 17*, 2005, pp. 233–240.

[26] J. Zaragoza, L. Sucar, and E. Morales, "Bayesian chain classifiers for multidimensional classification," in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2011, pp. 2192–2197.

[27] J. Berger, *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[28] D. Lewis, Y. Yang, T. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[29] B. Klimt and Y. Yang, "The Enron corpus: A new dataset for email classification research," in *Proceedings of the 18th European Conference on Machine Learning*, 2004, pp. 217–226.

[30] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[31] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, Tech. Rep., 2007.

[32] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

APPENDIX A.
PROOF OF PROPOSITION 1

Using the definition of $\ell(\hat{\mathbf{y}}, \mathbf{y})$ in (8), the risk $\mathcal{R}(\mathbf{y})$ can be decomposed accordingly as

$$\mathcal{R}(\hat{\mathbf{y}}) = \alpha \text{FN}(\hat{\mathbf{y}}) + \beta \text{FP}(\hat{\mathbf{y}}), \tag{23}$$

where

$$\text{FN}(\hat{\mathbf{y}}) = \sum_{\mathbf{y}} \left(\sum_{i:y_i=1 \wedge \hat{y}_i=0} c_i \right) P(\mathbf{y}|\mathbf{x}), \tag{24}$$

$$\text{FP}(\hat{\mathbf{y}}) = \sum_{\mathbf{y}} \left(\sum_{i:y_i=0 \wedge \hat{y}_i=1} c_i \right) P(\mathbf{y}|\mathbf{x}) \tag{25}$$

are contributions due to false negatives and false positives, respectively. Based on the definition of p_i in (15), we first introduce the following two lemmas.

Lemma 1. *FN*$(\hat{\mathbf{y}}) = \sum_{i:\hat{y}_i=0} c_i p_i$.

Proof: (24) can be rewritten as

$$\text{FN}(\hat{\mathbf{y}}) = \sum_{i:\hat{y}_i=0} c_i \sum_{\mathbf{y}} P(y_i = 1, \mathbf{y}_{\mathcal{H}\setminus\{i\}}|\mathbf{x}).$$

With $y_i = 1$, the hierarchy constraint in (1) or (2) requires that $\mathbf{y}_{\text{anc}(i)} = \mathbf{1}$. Hence, $\sum_{\mathbf{y}} P(y_i = 1, \mathbf{y}_{\mathcal{H}\setminus\{i\}}|\mathbf{x}) = P(y_i = 1, \mathbf{y}_{\text{anc}(i)} = \mathbf{1}|\mathbf{x}) = p_i$, and the result follows. ■

Lemma 2. *FP*$(\hat{\mathbf{y}}) = \sum_{i:\hat{y}_i=1} c_i(1 - p_i)$.

Proof: (25) can be rewritten as

$$\begin{aligned}
\text{FP}(\hat{\mathbf{y}}) &= \sum_{i:y_i=0 \wedge \hat{y}_i=1} \sum_{\mathbf{y}} c_i P(\mathbf{y}|\mathbf{x}) \\
&= \sum_{i:\hat{y}_i=1} \sum_{\mathbf{y}:y_i=0} c_i P(\mathbf{y}|\mathbf{x}) \\
&= \sum_{i:\hat{y}_i=1} c_i P(y_i = 0|\mathbf{x}) \\
&= \sum_{i:\hat{y}_i=1} c_i (1 - p_i).
\end{aligned}$$

■

On combining both lemmas, we immediately obtain Proposition 1.

APPENDIX B.
PROOF OF PROPOSITION 2

Recall that $\hat{\mathbf{y}}_{(L)}$ is the multilabel with L nodes labeled positive. Without loss of generality, assume that these L nodes (denoted $n_1, n_2, \ldots, n_L$) are sorted in topological order. Let $\hat{\mathbf{y}}_{(l)}$ be the multilabel with the first l of these nodes labeled positive, and denote the corresponding set of nodes be $S_l = \{n_1, n_2, \ldots, n_l\}$. Recall that the root is always labeled positive. Hence, $\hat{\mathbf{y}}_{(1)}$ is the multilabel with only the root node labeled positive, and $S_1 = \{0\}$.

From (18) and using telescoping, we have

$$\begin{aligned}
\hat{\mathbf{y}}^*_{(L)} &= \arg \max_{\hat{\mathbf{y}}:|\text{supp}(\hat{\mathbf{y}})|=L} \mathcal{R}(\hat{\mathbf{y}}_{(1)}) - \mathcal{R}(\hat{\mathbf{y}}) \\
&= \arg \max_{n_1,n_2,\ldots,n_L} \sum_{i=1}^{L} \delta_{n_i},
\end{aligned} \tag{26}$$

where $\delta_{n_1} \equiv 0$, and $\delta_{n_i} \equiv \mathcal{R}(\hat{\mathbf{y}}_{(i-1)}) - \mathcal{R}(\hat{\mathbf{y}}_{(i)})$. From (23), δ_{n_i} can be rewritten as

$$\begin{aligned}
\delta_{n_i} &= (\alpha \text{FN}(\hat{\mathbf{y}}_{(i-1)}) + \beta \text{FP}(\hat{\mathbf{y}}_{(i-1)})) \\
&\quad -(\alpha \text{FN}(\hat{\mathbf{y}}_{(i)}) + \beta \text{FP}(\hat{\mathbf{y}}_{(i)})) \\
&= \alpha(\text{FN}(\hat{\mathbf{y}}_{(i-1)}) - \text{FN}(\hat{\mathbf{y}}_{(i)})) \\
&\quad +\beta(\text{FP}(\hat{\mathbf{y}}_{(i-1)}) - \text{FP}(\hat{\mathbf{y}}_{(i)})).
\end{aligned} \tag{27}$$

Note that $S_i = S_{i-1} \cup \{n_i\}$ and $\hat{\mathbf{y}}_{(i)}$ is a valid multilabel. Thus, on using Lemma 1, we have

$$\begin{aligned}
\text{FN}(\hat{\mathbf{y}}_{(i-1)}) - \text{FN}(\hat{\mathbf{y}}_{(i)}) &= \sum_{j:\hat{\mathbf{y}}_{(i-1),j}=0} c_j p_j - \sum_{j:\hat{\mathbf{y}}_{(i),j}=0} c_j p_j \\
&= c_{n_i} p_{n_i}.
\end{aligned} \tag{28}$$

Similarly, on using Lemma 2,

$$\begin{aligned}
\text{FP}(\hat{\mathbf{y}}_{(i-1)}) - \text{FP}(\hat{\mathbf{y}}_{(i)}) &= \sum_{j:\hat{\mathbf{y}}_{(i-1),j}=1} c_j(1 - p_j) \\
&\quad - \sum_{j:\hat{\mathbf{y}}_{(i),j}=1} c_j(1 - p_j) \\
&= -c_{n_i}(1 - p_{n_i}).
\end{aligned} \tag{29}$$

On combining (27), (28), and (29), thus, $\delta_{n_i} = c_{n_i}(\alpha p_{n_i} - \beta(1 - p_{n_i}))$.