Automatically Adjusting Content Taxonomies for Hierarchical Classification

Lei Tang
Dept. of CSE
Arizona State University
Tempe, AZ, U.S.
L.Tang@asu.edu

Jianping Zhang
America Online Inc.
22000 AOL way
Dulles, VA, U.S.
JZhang6805@aol.com

Huan Liu
Dept. of CSE
Arizona State University
Tempe, AZ, U.S.
huan.liu@asu.edu

Abstract

Hierarchical models have been shown to be effective in content classification. However, the performance of the model heavily depends on the given hierarchical taxonomy. We empirically show that different taxonomies can result in significant differences in hierarchical classification performance. Motivated by some real application problems, we aim to modify a content taxonomy automatically for different applications. In this work, we formulate the problem, discuss why it is feasible to achieve better performance in terms of classification performance via adjusting a given hierarchy, and present one effective solution to find better hierarchies compared with that of the given original hierarchy. Preliminary experiments on some real world data sets are reported and discussed.

1 Introduction

Effectively organizing web content into topic categories of taxonomies facilitates content management and retrieval. Many machine learning methods have been applied to content categorization, but most of them did not take advantage of the valuable information embedded in a hierarchical taxonomy. These methods are deemed as flat models. Methods exploiting the category dependency in a content hierarchy are called hierarchical models. It has been showed that hierarchical models outperform flat models in training efficiency, classification efficiency, and classification accuracy [4, 6, 3, 2, 8, 5]. Actually, the improvements in accuracy heavily depends upon the quality of the hierarchy. Content hierarchies are created for ease of content management or access, so semantically similar categories are grouped into a parent category. Semantically similar categories may not be similar in terms of lexical terms,so the hierarchy might not be the best taxonomy for classification. Moreover, even for the same categories, different people might organize it into different hierarchies. This motivates us to study different hierarchies.

In this paper, we empirically show that a better hierarchy can be created for a human provided semantics-based hierarchy. We also describe a method for creating such a hierarchy. On one hand, we could totally ignore the original semantics based hierarchy and create an entirely new hierarchy using some hierarchical clustering algorithms. The drawback of this approach is that it's hard to control the height of the hierarchy and the number of sub-categories under a parent category. On the other hand, we could start with the original hierarchy and gradually modify it to achieve a better hierarchy. This is the approach described in this paper.

The paper is organized as follows: We briefly introduce some of the potential content categorization applications in AOL [1] and formulate the problem. Then we provide an algorithm for creating a better hierarchy. We report experimental results on some real-world data and conclude in Section 6.

2 A Challenge

Different real world applications may require different content taxonomies. AOL's guardian report product reports child online activities including a list of websites visited by the child to his/her guardians. To be more informative, this list of visited websites should be organized according their categories. For this application, a shallow hierarchy is sufficient. Actually, only top level categories such as sports, game etc. satisfy the requirement of this application. On the other hand, a deep hierarchy is more desirable for a news feed classifier. For a bookmark management system, users may want to adjust the taxonomy to best fit his/her own need. Therefore, disparate hierarchical models are suitable for different tasks.

However, the existing general hierarchy is so often not designed for different applications so that its performance can be unsatisfactory. Since the original given hierarchy already provides some valuable information, it is reasonable to modify the hierarchy to a hierarchy with better classification performance than the original given hierarchy. This raises the issue for us to investigate different hierarchies in this work.

Actually, when we review the process of taxonomy generation and data collection, we notice that there might be some inconsistency between taxonomy and classification. Taxonomies are generated based on semantics difference, and different people might use different taxonomies on the same set of categories. But for hierarchical classification, these hierarchies might result in totally different performance. In this work, we hope to bridge the gap between the taxonomy and data. That is, given a taxonomy based on semantics, we aim to generate a new taxonomy which is better for classification.

3 Problem Formulation

Before we formalize our problem, we present several definitions concerning hierarchies as below:

Definition 1 (Consistent Hierarchy) *Given a hierarchy H and a set of categories S, if the categories at the leaf nodes of H is the same as S, then H is a consistent hierarchy with respect to S. If the categories of a data set D are S, we say, H is a consistent hierarchy for D, or H and D are consistent.*

Definition 2 (Optimal Hierarchy)

$$\begin{aligned} H_{opt} &= \arg\max_H p(D|H) \\ &= \arg\max_H \log p(D|H) \end{aligned}$$

where H is a consistent hierarchy for given data D.

In other words, the optimal hierarchy given a data set should be the one with maximum likelihood. The brute-force approach to find the optimal hierarchy is to try all the possible consistent hierarchies and output the optimal one. Unfortunately, even for a very small set of categories, there could be a huge number of consistent hierarchies. It is impractical to try all the possible hierarchies and pick the optimal. Thus, a more effective way should be explored.

The given hierarchy provides valuable information for classification and help reduce the search space to find the intended optimal hierarchy. To incorporate this knowledge into our problem formulation, we first give the definition of hierarchy difference.

Definition 3 (Similar Hierarchy) *If two hierarchies H and H' are both consistent with respect to a set of categories S, then H and H' are similar.*

Definition 4 (Hierarchy Difference) *Hierarchy difference is the minimum number of elementary operations(see below) to transform a hierarchy H into the hierarchy H' that is similar to H. Suppose the minimum number of operations is k, we denote the difference between H and H' as*

$$\| H' - H \| = k$$

In order to change a hierarchy to its similar hierarchy, we have three elementary operations:

Lift : roll up one node to upper level;

Shift : push down one node to its sibling;

Merge : merge two nodes to form a super node;

As shown in Figure 1, H_1 is the original hierarchy. H_2, H_3 and H_4 are obtained by lifting Node 6 to its upper level, shifting Node 3 under its sibling Node

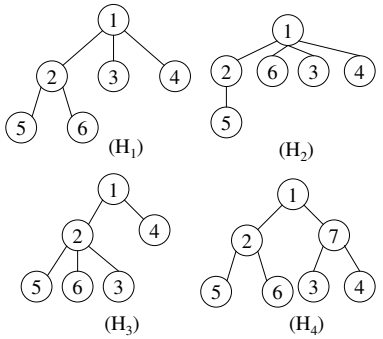

Figure 1: Hierarchy Elementary Operations

2, and merging Node 3 and 4, respectively. Node 7 is a newly generated node after modification. As the set of leaf node in our problem always remains unchanged, we do not require the operation of splitting one node into two.

Given explicit hierarchy difference, we have the constrained optimal hierarchy:

Definition 5 (Constrained Optimal Hierarchy) *Given a hierarchy H_0, if there exists a sequence of hierarchies $Q = \{H_1, H_2, \cdots, H_n\}$ such that*

$$\begin{aligned} p(D|H_i) &\geq p(D|H_{i-1}) \\ \parallel H_i - H_{i-1} \parallel &= 1 \qquad (1 \leq i \leq n) \end{aligned}$$

and

$$\begin{aligned} \forall H' \qquad & s.t. \parallel H' - H_n \parallel = 1 \\ we\ have \quad & p(D|H') \leq p(D|H_n) \end{aligned}$$

then H_n is a constrained optimal hierarchy for H_0 and D.

We assume that the optimal hierarchy for the data should reside among one of the constrained optimal hierarchies. If we consider the problem as an exact search problem, then a given hierarchy can be considered as a good starting point which can reach the optimal one following a short path. Then, we formulate our challenge as follows:

Hierarchy Search Problem: Given data D, and a hierarchy taxonomy H, Can we find a hierarchy H_{opt} such that

$$H_{opt} = \arg\max_{H} \log p(D|H)$$

where H is a constrained optimal hierarchy for D and H_0.

We can consider hierarchy search problem as searching in the hierarchy space. Our assumption assumes that the optimal hierarchy should reside within the vicinity of the given hierarchy. In nature, each constrained optimal hierarchy is a local optima starting from the given hierarchy. And the optimal hierarchy should be one of the constrained optimal hierarchies.

4 Approximate Solution

There exist some problems that prevent us from directly finding an effective solution for the hierarchy search problem.

- How to estimate the likelihood of data given a hierarchy ($P(D|H)$ in Definition 2)?
- While the hierarchy search problem proposes to select the best among the constrained optimal hierarchies, it is extremely computational expensive in reality to obtain all the constrained optimal hierarchies.
- How to find the neighbors of a hierarchy? Actually there could be a huge number of neighbors by performing only one elementary operation to a specific hierarchy especially when the number of nodes in the tree is large. Moreover, not all these operations lead to a better hierarchy. We need to filter out some bad movement in the hierarchy space.

So, rather than find the exact solution, we propose to use some heuristics to get an approximate solution. For the first problem, instead of directly estimating the likelihood of data given a hierarchy, we use some statistics to estimate the goodness of a hierarchy. Since in most classification tasks, people focus on macro-averaged recall or f-measure [7, 5], we use them as the classification performance of a hierarchical model to measure its likelihood. In particular, we use macro-averaged recall as our criterion to help estimate the conditional likelihood. Concerning the second problem, we exploit a greedy approach to find the best constrained hierarchy. That is, in each search step, we always choose the neighbor node with largest likelihood improvement until we reach a con-

strained optimal hierarchy.

Even so, we still need to consider the number of neighbors of a hierarchy, which actually could be huge by considering all the possible operations. Here, we use several reasonable heuristics to decrease the possible space. Before we present the heuristics, we'd like to give several definitions which facilitate the description of the heuristics.

Definition 6 (High Miss/Low Miss) *For a node in the hierarchy, if it's misclassified at the parent level, then this misclassification is called High Miss. If it is misclassified as its sibling under the same parent node, then it's a Low Miss.*

Heuristic 1 *If the High Miss of one node is significantly larger than the Low Miss of one node, that is,*

$$High\ Miss > Low\ Miss + \delta$$

where δ is a user defined parameter, we lift this node to the upper level.

Definition 7 (Ambiguity Score) *Given two classes A and B, suppose the percentage of class A classified as class B is P_{AB}, and the percentage of class B classified as Class A is P_{BA}, then ambiguity score = P_{AB} + P_{BA}.*

Heuristic 2 *We can calculate the Ambiguity Score for each pair of categories under the same parent node. For each subtree in the hierarchy, we pick the pair A and B with highest ambiguity score. If $|P_{AB} - P_{BA}| \leq \gamma$ where γ is a predefined threshold, then we merge A and B to form a super category; Otherwise, if $P_{AB} > P_{BA} + \gamma$, we shift Class A as B's child; If $P_{BA} > P_{AB} + \gamma$, then we shift Class B under Class A.*

Based on these three heuristics, the search space of hierarchies is sharply condensed. Then we can use a wrapper model to search for better hierarchies. The detailed algorithm is in Figure 2.

5 Experiments

We apply our algorithm to two real world data sets provided by AOL [1]: one is a data set concerning the topic of social study(*Soc*) obtained from AOL's web directory; the other one is Kids' data collected by AOL'kids department(*Kids*). Each data set is associated with a hierarchical taxonomy. The general information of these two data sets is shown in Table 1.

Input: A hierarchy H_0, training data T and a validation set V
Output: A approximate optimal hierarchy H_{opt} **Algorithm:**
1. $score_{best} = 0$, $H_{list} = \{H_0\}$
2. $flag = false$(To denote whether or not the hierarchy is changed)
3. For each hierarchy H_i in H_{list}, build a hierarchical model based on T and evaluate its performance on V. If the corresponding statistic(macro recall) *score* is larger than $score_{best}$, then $flag = true$, $score_{best} = score$ and $H_{opt} = H_i$.
4. If $flag == false$, return H_{opt}.
5. After H_{list} is empty, according to Heuristic 1 and 2, generate neighbors for H_{opt} by checking each node in H_{opt}. Add all these neighbors to H_{list}.
6. If H_{list} is empty, return H_{opt}; Otherwise, goto step 2.

Figure 2: Hierarchy Search Algorithm

	Soc	*Kids*
#classes	69	244
#nodes in the hierarchy	83	299
Height of the hierarchy	4	5
#instances	5248	15795

Table 1: Real-world Data Description

For this problem, we use macro-averaged recall to calibrate the conditional likelihood of data given a hierarchy. We set δ in Heuristic 1 to 0 and γ in Heuristic 2 to 0.01. Here, we use the training data as the validation data in the algorithm as well. That is, we keep adjusting the hierarchy until no training errors can be reduced. When we build the hierarchical model for hierarchy modification, we select 500 features using information gain at each node before building the classifier [7]. After we obtain the adjusted hierarchy,we evaluate it by selecting different number of features at each node to build the hierar-

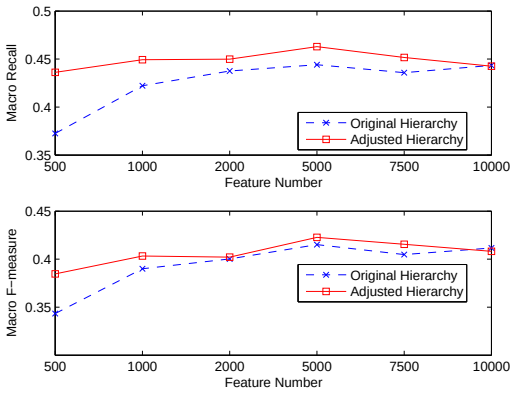

Figure 3: *Soc* Performance

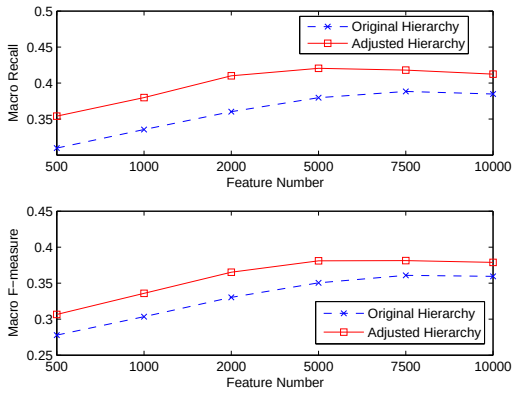

Figure 4: *Kids* Performance

chical model. As shown in Figure 3 and Figure 4, significant improvement over the original taxonomy is observed on both data sets in terms of macro recall and macro F-measure, especially when the number of features is relatively small. This shows that we can automatically adjust the content taxonomies for more accurate classifiers. One interesting result is: for *Soc*, when we select more and more features, the difference between the newly generated hierarchy and given hierarchy wanes. However, for *Kids*, the difference is independent of the number of features we selected.

6 Conclusions

Hierarchical models have been shown to be effective for classification when we have a predefined taxonomy in hand. Rather than taking it for granted, we suggest that the given hierarchy does not necessarily lead to the best classification performance, and propose to modify the original given hierarchy by studying the training data. We formulate the problem and provide an approximate solution by providing several heuristics. The real-world application data shows that our algorithm finds a better hierarchy which is similar to the given hierarchy but improves the classification significantly, especially when only a few features are selected. It is noticed that the hierarchy search space is highly connected and the greedy search in each step still generates too many neighbors. We are currently working on more efficient and effective methods.

References

[1] America Online Inc. http://www.aol.com/.

[2] Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *CIKM '04*, pages 78–87, 2004.

[3] Susan Dumais and Hao Chen. Hierarchical classification of web content. In *SIGIR '00*, pages 256–263, 2000.

[4] Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *ICML '97*, pages 170–178, 1997.

[5] Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36–43, 2005.

[6] Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *ICML '98*, pages 359–367, 1998.

[7] Lei Tang and Huan Liu. Bias analysis in text classification for highly skewed data. In *ICDM'05*, 2005.

[8] Yiming Yang, Jian Zhang, and Bryan Kisiel. A scalability analysis of classifiers in text categorization. In *SIGIR '03*, pages 96–103, 2003.