Document Classification with ACM Subject Hierarchy

Tao Wang
Department of Computer Science
Concordia University
Montreal, Quebec, Canada
e-mail: wan_tao@cs.concordia.ca

Bipin C. Desai
Department of Computer Science
Concordia University
Montreal, Quebec, Canada
e-mail: bcdesai@cs.concordia.ca

Abstract

Text Categorization or Text Classification (TC) has recently received increased research attention from Information Retrieval and Machine Learning communities, this focus is driven mostly by the ever growing demand for effective and efficient content-based document management. In the context of Digital Library or Web Portal application, the problem of text categorization is normally that of classification scheme with a topic hierarchy containing all the pre-defined categories. This paper describes our approach to building the hierarchical text classifier for the experimental CINDI Digital Library [1]. The classification system constructed features a top-to-down, coarse-to-fine categorization procedure. We evaluate our system's performance by experiment on a self-generated corpus of the Computer Science papers archived in ACM DL.

1 Introduction

Text categorization or text classification is a task that assigns one or more pre-defined categories to a document based on its content. Roughly, two kinds of classification problem are addressed: flat and hierarchical. In the former case, all categories or classes in category set C are treated equally. There is no specific relationship defined on C. Each document can be classified into, depending on the application, exactly one category (called multi-class classification) or any number of them (called multi-label classification). The Probabilistic Model, Support Vector Machine (SVM), k Nearest Neighbors (k-NN), and Centroid top a number of inductive learning methods for classifier construction that have been studied and experimented extensively [7], [8], [11].

In the context of Digital Library or Web Portal, the category space C is typically organized into a tree (e.g., ACM Classification Scheme) or a Directed Acyclic Graph (e.g., Google's Web Directory). This hierarchical classification problem, theoretically, may be treated as a number of independent binary ones. However, its feasibility for real world applications is very questionable because of the number of categories contained and the fine-grained nature of these categories. A hierarchical structure suggests a top-to-down, coarse-to-fine classification scheme [3], [9]. The key is to preserve the potential categories at each level of hierarchy for further classification while aggressively cutting off unnecessary branches. Unfortunately, most of research work reported do not provide detail of their methods. The commercial systems are trade-secrets, and their accuracy cannot be verified. In this paper, we present our approach for hierarchical TC. Specifically, we propose a method to control the number of categories selected, and examine two different methods for re-ranking categories at each level. The experimental results to date show a satisfactory improvement over other schemes previously used [10].

The rest of this paper is structured as follows: Section 2 formulates a hierarchical text classification task; Section 3 describes our approach for hierarchical TC; we present the experiment results in Section 4; Section 5 gives our conclusions.

2 An Framework for Hierarchical Text Categorization

The classification schemes in hierarchical TC are usually given in one of the following two forms:

• *A Tree* The well-known "parent-child" relationship represents generality or specificity between two categories. The set of categories are partitioned into *real categories* for holding document and *virtual categories* for further classification. Usually, all leaf nodes are mapped to real categories, and all internal nodes to virtual ones, as illustrated in ACM Classification Scheme [2].

• *A Graph* More strictly, it is a Directed Acyclic Graph (DAG). An edge represents generality or specificity between two related categories. However, a category may have more than one direct general category, and every category can be a real one. The graph-formed classification scheme implies that documents can be categorized into the same category but along different paths. Google's and Yahoo's [3] web directories are two typical examples.

2.1 A Framework

We characterize a hierarchical text classification system, denoted by $\mathcal{HS}$, as a 4-tuple, $\mathcal{HS} = (\Omega, H, C_r, F)$, where:

• Ω represents a corpus or dataset, consisting of a set D of pre-labeled documents. Ω has two distinct formats, depending on the type of classification:

$$\textit{multi-class:} \quad \Omega = \{(d_i, c_j) \mid d_i \in D,\ c_j \in C\}, \qquad (1)$$

$$\textit{multi-label:} \quad \Omega = \{(d_i, C_i) \mid d_i \in D,\ C_i \subseteq C_r\}. \qquad (2)$$

• H is a DAG, denoted by $H := (C, E)$, standing for a classification scheme. C is a set of categories contained in H, and E defines the structure of these categories. As such, an edge $(c', c'') \in E$ means category c' is more general (broader) than c''.

[1] https://cindi.encs.concordia.ca

[2] http://www.acm.org/class/1998/ccs98.txt

[3] http://www.google.com/dirhp, http://dir.yahoo.com/

• C_r denotes a set of real categories into which each document can be classified, $C_r \subseteq C$.
• F is the unknown hierarchical classification function to be learned on Ω by means of some learning method (algorithm) that maps each document to one (multi-class) or any number of categories (multi-label):

$$\textit{multi-class:} \quad F_{mc} : D' \rightarrow C_r, \tag{3}$$
$$\textit{multi-label:} \quad F_{ml} : D' \rightarrow 2^{C_r}; \tag{4}$$

where, $D' \supset D$ denotes a set of all possible documents. The accuracy of F can be evaluated, by approximation, on a part of Ω.

In what follows, we narrow the hierarchical TC problem with such properties: a) a "tree" classification scheme; b) multi-label classification; c) all leaf nodes being real categories.

3 Approaches for Hierarchical Text Classification

3.1 Related Works

Intuitively, a hierarchical classification function F_{ml} can be constructed by combining total $\mid C \mid$ independent binary classifiers, each of them corresponding to a node in H. This method would work well for a small hierarchy as given in [6]. However, if the number of categories in H is thousands, tens of thousands, or even larger, applicability of this idea to real world application is very questionable because both training and classification processes involves $\mid C \mid$ repetitions. Another possible approach is to flatten the hierarchical structure, resulting in a "flat" $\mid C_r \mid$ multi-label classification task. Since the value C_r is large, global discerning among these classes relying on only one classifier is extremely difficult.

Many researchers suggests that one should take advantage of the hierarchical structure in doing hierarchical TC, specifically, by breaking a large classification into a number of smaller ones to improve effectiveness, and more importantly, by selecting potential edges for further classification to increase efficiency. For instances, Susan & Chen [3] utilize a hierarchical model for web page categorization using part of MSN Web Directory. The authors proposed a method of combining both upper and lower categorization results to determine the final classification at lower level. Even though justified by the following experiments, the specifics for this combination method remains unknown. Avancini et al. [1] suggested a "soft pruning" scheme in branch selection in order to recover some promising branches cut by threshold. Sun et al. [9] compared this "top-down method with flattened counterpart on the well-known Reuters corpus. The experimental results clearly favored the former. All of these works have shown to some degree the advantage of "top-down" approach in hierarchical TC.

3.2 Our Approach

Herein, we present our approach for hierarchical TC. Particularly, we specify and formulate our methods in ranking categories on the same level for further classification. Adopting the top-down model, we first construct total number of $\mid Parent(H) \mid$ local independent classifiers, where $Parent(H)$ is a set of parent nodes in H. Each parent corresponds to a flat multi-label classifier with the associated classification scheme consisting of its children categories. An input document at first goes through the root classifier (at level 0). The output results then trigger the connected level 1 classifiers for finer classification, which, in turn, trigger some level 2 classifiers. The document finally reaches the leaf nodes which contain the classification results.

3.2.1 Flat Multi-label Classifiers

The flat local multi-label classifier serves the basic block in our hierarchical classification system. The strength of the underlying learning methods for flat classifier construction, therefore, have a significant impact on the global effectiveness of the system. We choose the following two types of classifiers because of their efficiency and average effectiveness.

1. *Naive Bayes Classifier* The multi-label version of Naive Bayes classifier is given by:

$$P(C = c_i \mid D = d_j) = \frac{P(C = c_i) \times P(D = d_j \mid C = c_i)}{P(D = d_j)}, \tag{5}$$

where the event spaces is a set of documents D; C and D are two random variables. $P(D = d_j)$ is the probability that a randomly picked document has the same representation as d_j; $P(C = c_i)$ as the probability that a randomly picked document belongs to c_i. The value $P(C = c_i \mid D = d_j)$ can be used for ranking categories $c \in C$.

2. *Centroid Classifier* We modify the classic centroid computation to reflect the multi-label property. Here, the centroid of each category c_i can be simply represented by sum of each document in c_i weighted by inverse of number of categories labeled for that document:

$$\odot_{c_i} = \frac{1}{\mid Cat_i(d_j) \mid \times \mid D_{c_i} \mid} \sum_{i=1}^{|D_{c_i}|} d_j, \tag{6}$$

where, D_{c_i} is a set of documents in c_i; $\mid Cat_i(d_j) \mid$ is the number of labeled categories associated with d_j. If some document has only c_i class label, it is viewed as a perfect positive example for c_i and therefore giving the maximum weight $\frac{1}{|D_{c_i}|}$.

The cosine-similarity between d_j and c_i

$$Sim(d_j, c_i) = \frac{\odot_{c_i} \cdot d_j}{\mid \odot_{c_i} \mid\mid d_j \mid} \tag{7}$$

measures the strength of d_j belonging to c_i that can be used for ranking categories for d_j.

3.2.2 Methods for Branch Selection

Once the local multi-label classifier are learned, we need to estimate a set Φ of parameters, $\Phi = \{\phi_0, \phi_1, \ldots, \phi_{K-1}\}$, each of which determines the maximum number of branches at some level selected for next classification. For example, the value of ϕ_1 represents the maximum number of branches at level 1 that can be preserved for next level classification. Here, we use

$$\phi_i = \arg\max_{d_j \in Tr(D)}\{Cat^i(d_j)\} + (i+1) \quad (8)$$

to obtain parameter ϕ_i, which is the maximum number of categories at level i from all training document plus $i+1$.

To re-rank the categories at the same level, but in different classifiers (except for level 0 where there is only "root" classifier), the following two schemes are applied:

1. *Ranking by addition* For some document d_j, the degree of d_j belonging to some category c^n at level n is given by $V(d_j, c^n) + V(d_j, c^{n-1})$, where, $V(d_j, c^n)$ is the value given by local classifier, $V(d_j, c^{n-1})$ is the value returned by another local classifier. Also note that c^{n-1} is the parent of category c^n.
2. *Ranking by multiplication* Similar settings to the above, but we re-rank the categories at some level based on their values $V(d_j, c^n) \times V(d_j, c^{n-1})$

4 Dataset Generation and Analysis

Machine Learning based classification assumes that a corpus of pre-labeled documents is available. This section presents our method in generating the required dataset for classifier construction. We also discuss the characteristics of this corpus.

4.1 Overview of ACM Classification Scheme

The latest ACM Classification Scheme (ACMCS-98) is adopted in our system. It is a tree-like hierarchy that consists of around $2,000$ categories. According to stipulation by ACM's classification committee, only leaf categories are used for categorization destination that total around $1,600$. Note that that this structure is more complicated than ones appearing in benchmark corpora such as "Reuters", "20 Newsgroups". Also, scientific papers categorization might be harder than news story classification [1]. Therefore, the results of classification could compare infavorably with other classification results reported in the literature.

4.2 Labeled Documents

As almost every document stored in ACM Digital Library has been labeled, it provide us with a very rich set of labeled documents for the sake of classifier training. Specifically, for each leaf category $c_i \in C_r$, we query the ACM DL with code for c_i (for example, codes "F.4.1" for category Mathematical Logic", and "H.3.2" for category "Information Storage"). We then choose the most N relevant documents to the query code as the positive examples for that category. The positive examples for some external category can be easily obtained by simply taking union of documents belonging to its child categories.

Category Sets	Level One	Level Two	Level Three
Number of Category	6	72	648

TABLE I
CATEGORY STATISTICS OF SUBTREE GENERATED FROM ACMCS, $Lev(i)$ DENOTES A SET OF NODES WITH DEPTH i

4.3 Properties of Scientific Documents

From the corpus generated, the following facts are observed when each document is represented by a bag-of-word format:

1. *High-dimensioned Space* If every word (excluding stop words) appears in the documents from corpus Ω is qualified for a text feature, the whole Ω would yield a feature space comprised of around $50,000$ words.
2. *Sparse Document Vectors* Comparing to a highly dimensional feature space, a document in Ω contains, on average, a few thousand features
3. *Low Feature Overlapping* When two random document vectors with the same or very close category are compared to each other, it shows that only a fews of words are shared by both.

5 Experiments and Results

5.1 Experimental Settings

For the subject hierarchy, we prune from ACM CS a couple of branches, resulting in a leaner topic tree. The number of categories at each level is detailed in Table 1:

As with the set of labeled documents, we split them into training $Tr(\Omega)$, validating $Va(\Omega)$, and testing portions $Te(\Omega)$ with a ratio $Tr(\Omega) : Va(\Omega) : Te(\Omega) = 7 : 1 : 2$. Total text features extracted from documents classified under this subtree are $45,000$ after stop-word removal and simple stemming processing, slightly less than the feature number generated using whole corpus. These extracted features are further reduced dramatically by: a) Term's Document Frequency of at least 2; b) either using statistics "Information Gain" or "Odds Ratio" [11]. At last, we choose around 100 features with highest measures for each classifier.

5.2 Evaluations

Finally, we evaluate the effectiveness of the proposed hierarchical TC system with the widely used F_1^m micro-average measure, which combines micro-average precision p^m and micro-average r^m as

$$F_1^m = \frac{2 \cdot p^m \cdot r^m}{p^m + r^m}. \quad (9)$$

Measures p^m and r^m are defined on the test set $Te(D')$ by, respectively,

No.	1	2	3	4	5	6	...	72	Micro-average
Level 0	.801	/	/	/			/	/	.752
Level 1	.718	.679	.665	.684	.675	.684	/	/	.681
Level 2	.635	.623	.619	.647	.633	.601	...	.643	.638
Performance of hierarchical classifier: .614									

TABLE II

F_1 MEASURES OF EACH INDEPENDENT CLASSIFIER AND OF LEVEL AVERAGE

$$p^m = \frac{\sum_{i=1}^{|C_r|} \mid S_i \cap T_i \mid}{\sum_{i=1}^{|C_r|} \mid S_i \mid}, \tag{10}$$

$$r^m = \frac{\sum_{i=1}^{|C_r|} \mid S_i \cap T_i \mid}{\sum_{i=1}^{|C_r|} \mid T_i \mid}, \tag{11}$$

where, $S_i \subset Te(D')$ is a set of documents classified by our classifier into category $c_i \in C_r$, and $T_i \subset Te(D')$ stands for a set of documents with a label $c_i \in C_r$. Since we deal with the hierarchical classification, the above performance evaluation has been done on a level basis.

5.3 Results and Discussion

We have observed from the experiments the following facts: 1) Naive Bayes based local classifiers outperform Centroid based classifiers; 2) Ranking by addition scheme is better than Ranking by multiplication one; 3) using "Information Gain" for feature selection is better than "Odds Ratio". Table 2 records the performance of our system using the ranking-by-addition scheme.

The main reason of our system's performance compared to published classifiers, may be attributed to the fact that classification scheme and textual data which we deal with are more complicated than the widely used Reuters-21578 standard corpus. It is not quite reasonable to compare their performances directly. Nevertheless, the results justify suitability of top-down method for the large hierarchical TC.

6 Conclusions

Text categorization is an effective paradigm in managing large number of documents. In this paper, we present a generic method for large-scale hierarchical text classification in the context of Digital Library. We evaluate the implemented classification system using a self-generated corpus and ACM Classification Scheme. The final experimental results justify the feasibility of our method.

References

[1] H. Avancini, A. Rauber1, and F. Sebastiani, Organizing digital libraries by automated text categorization, *Workshop on Artificial Intelligence for Cultural Heritage and Digital Libraries* 2001

[2] M. F. Caropreso, S. Matwin, and F. Sebastiani, A learner-independent evaluation of the usefulness of statistical phrases for automated text classification, in A. G. Chin (Ed.), *Text Database and Document Management: Theory and Practice* (2001) 78-102.

[3] S. Dumans and H. Chen, Hierarchical classification of Web content, *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (2000), 256-263

[4] J. Fagan, Automatic phrase indexing for document retrieval, in C. T. Yu, and C. J. Van Rijsbergen (Eds.), *Proceedings of the 10th annual international ACM SIGIR conference on Research and development in information retrieval (ACM SIGIR'87)* (USA, 1994) 91-101.

[5] D. D. Lewis, An evaluation of phrasal and clustered representations on a text categorization task, in: N. Belkin, P. Ingwersen, and A. M. Pejtersen (Eds.), *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Denmark 1992) 37-50.

[6] M.E. Ruiz and P. Srinivasan, Hierarchical neural networks for text categorization, *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (1999), 281-282

[7] M. Sasaki and K. Kita, Rule-based text categorization using Hierarchical category. In *Proceedings of IEEE International Conference on System*, pages 2827-2830, 1999.

[8] F. Sebastiani, Machine learning in automated categorization, *ACM Computing Surveys* 34 (2002) 1-47.

[9] A. Sun and E.P. Lim, Hierarchical Text Classification and Evaluation, *First IEEE International Conference on Data Mining* (ICDM'01) 2001, 521-529

[10] T. Wang and B.C. Desai *Proceeding of Canadian Conference on Electrical and Computer Engineering 2006* (CCECE06), 1847-1852.

[11] Y. Yang, An evaluation of statistical approach to text categorization, *Information Retrieval* 1 (1999) 69-90.