On the merits of building categorization systems by supervised clustering

Charu C. Aggarwal, Stephen C. Gates, Philip S. Yu
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{ charu, scgates, psyu }@us.ibm.com

Abstract

This paper investigates the use of supervised clustering in order to create sets of categories for classification of documents. We use information from a pre-existing taxonomy in order to supervise the creation of a set of related clusters, though with some freedom in defining and creating the classes. We show that the advantage of using supervised clustering is that it is possible to have some control over the range of subjects that one would like the categorization system to address, but with a precise mathematical definition of each category. We then categorize documents using this a priori knowledge of the definition of each category. We also discuss a new technique to help the classifier distinguish better among closely related clusters. Finally, we show empirically that this categorization system utilizing a machine-derived taxonomy performs as well as a manual categorization process, but at a far lower cost.

1 Introduction

In recent years, the amount of online text data has grown greatly because of the increase in popularity of the World Wide Web. As a result, there is a need to provide effective content based retrieval, search, and filtering for these huge and unstructured online repositories. In this paper, we consider the problem of automated text categorization, in which we desire to find the closest matching subjects for a given text document. We assume that a pre-existing sample of training documents with the associated classes is available in order to provide the supervision to the categorization system.

Several text classifiers have recently been proposed, such as those discussed in [2, 4, 5]. These classifiers have shown excellent results on document collections such as the *Reuters* dataset or the US patent database [2] and to a somewhat lesser extent on the web with the *Yahoo*! taxonomy. Categorization of web documents has proven to be especially difficult because of the widely varying style, authorship and vocabulary in different documents.

Most of the above-mentioned categorizations are created using manual categorizations by subject experts. The apparent inaccuracy of automated classification methods on large document collections is a result of the fact that a large heterogeneous collection of manually categorized documents is usually a poor fit for any given classification model. Thus, it is interesting to investigate the construction of categorization systems which relax the restriction imposed by predefined sets of classes. We study the use of supervised clustering in order to create the categories which are related to, but not quite the same as a pre-existing taxonomy.

The fact that we actually know the model used to construct each partition in the clustering ensures that we can theoretically obtain a perfect accuracy on this categorization when the same model is used by the categorizer. Therefore the quality of categorization depends completely on the quality and coherence of each cluster in the new taxonomy, rather than the accuracy of a training procedure on the original taxonomy.

The use of unsupervised clustering for providing browsing capabilities has been discussed in earlier work [3]. Such methods do not use any kind of supervision from a pre-existing set of classes, and are attractive for creation of a small number of clusters such as fifty or so, though the clustering rapidly degrades in quality when it is used for finding more fine-grained partitions. Typically, when categories are related enough to contain overlapping vocabulary, unsupervised clustering methods are unable to create a good fine grained subject isolation.

We also propose a classifier based on the clustering model which is able to distinguish between very closely related categories. The importance of creating a classification process which is able to distinguish between

closely related subjects has been discussed in earlier work [2]. This work uses a hierarchical taxonomy for effectively distinguishing between closely related categories. Such classifiers are quite fast, though the accuracy can be sensitive to the quality of the hierarchical organization. Our system provides high-quality categorizations on a flat set of classes without compromising speed.

2 The Categorization System

In order to represent the documents, we used the vector space model. In the vector space model, it is assumed that each document can be represented as as *term vector* of the form $\overline{a} = (a_1, a_2, \ldots a_n)$. Each of the terms a_i has a weight w_i associated with it, where w_i denotes the normalized frequency of the word in the vector space.

A *centroid* of a set of documents is defined by a concatenation of the documents in the set after damping the word frequencies. The damping function ensures that the repeated presence of a word in a single document does not affect the centroid of the entire cluster excessively.

A *projection* of a document is defined by setting the term frequencies (or weights) of some of the terms in the vector representation of the document to zero. These are the terms which are said to be *projected out*. We will use the process of projection frequently in the course of the supervised clustering algorithm. Each cluster is represented by a seed vector containing only a certain maximum number of projected words. The aim in projection is to isolate a relatively small vocabulary which describes the subject matter of a cluster well.

Our first phase was to perform the feature selection in such a way so that only the more differentiating words are used in order to perform the clustering. Note that in unsupervised clustering methods, where a pre-existing taxonomy is not used, the feature selection is somewhat rudimentary in that only *stop words* (very commonly-occuring words in the English language) are removed. In our case, we used the *gini index* of the word in order to pick out relevant words far more aggressively.

Let there be K classes $C_1, C_2 \ldots C_K$ at the lowest level in the original taxonomy. Let $f_1, f_2 \ldots f_K$ be the number of occurances of that word in each of the K classes, and let $n_1 \ldots n_K$ be total word count for the documents in each of the K classes. Thus, the *fractional presence* of a word in a particular class is given by f_i/n_i. We define the *skew fraction* of a word for class i by $\frac{f_i/n_i}{\sum_{i=1}^{K} f_i/n_i}$. We shall denote this skew fraction by p_i.

The normalized gini index of a word with skew fractions $p_1 \ldots p_K$ is given by $1 - \sqrt{\sum_{i=1}^{K} p_i^2}$. If the word is distributed evenly across the different classes, then the gini index is $1 - 1/\sqrt{K}$. This is the maximum possible value of the gini index. On the other hand, when the word is highly correlated with particular categories and is very skewed in its distribution, then the normalized gini index is much lower.

```
Algorithm Topical(D)
begin
  S = Initial set of seed meta-documents;
  iteration := 0;
  Initialize words, threshold, minimum;
  while not(termination-criterion)
  do begin
    (S, D) = Assign(S, D);
    S = Project(S, words);
    S = Merge(S, threshold);
    S = Kill(S, minimum);
    iteration := iteration + 1;
    words = words * θ;
  end
end
```

Figure 1: The clustering algorithm

The clustering algorithm uses a seed-based technique in order to create the clusters. Traditional clustering methods have often used seed-based algorithms in order to serve as an anchor point for the creation of the clusters. In other words, seeds form an implicit representation of the cluster partitioning in which each item to be categorized is assigned to its closest seed based on some distance (or similarity) measure. In the context of information retrieval, a seed is a meta-document which can be considered as a pseudo-representation of a central point in a given cluster. Most current clustering algorithms use sets of seeds in order to define the implicit partitions.

Since the focus of the algorithm is on *supervised clustering*, we started off with a set of seeds which are representative of the classes in the original taxonomy. These representative seeds are constructed by finding the centroids of the corresponding classes. This choice of starting point (and features picked) ensures the inclusion of supervision information from the old taxonomy, but the subsequent clustering process is independent of any further supervision. Each seed consists of a vector in which the number of words with a non-zero weight is restricted to a predefined maximum. The algorithm gradually reduces the projected dimensionality in each iteration, as the clusters get more refined, and a smaller vocabulary is required in order to isolate the subject of the documents in that cluster. This technique of representing clusters by using both the documents and the projected dimensions in order to represent a cluster is referred to as *projected clustering* [1], and is an effective technique for the creation of clusters for very high dimensional data. The idea of using truncation for speeding up document clustering has been discussed in [7], though our focus for using projections is different, and is designed

in order to improve the quality of the clustering by iteratively refining the dimensions and clusters. Thus, the projected clustering technique merges the problem of finding the best set of documents and features for a cluster into one framework. More details on the advantages of using projected clustering for very high dimensional data may be found in [1]. The basic framework of the clustering algorithm is illustrated in Figure 1. A pseudo-code for each of the individual subroutines is omitted for lack of space. The following four steps are applied iteratively in order to converge to the final set of clusters in the taxonomy. We assume that the set of seeds available to the algorithm at any stage is denoted by S and the documents which are being clustered by the algorithm are denoted by D.

(1) **Document Assignment:** In each iteration, we assign the documents to their closest seed in S. The similarity of each document to its closest seed is calculated using the cosine measure. Thus, a new partition of the documents in D is created by the set of seeds S. After the assignment process, the old set of seeds S are discarded, and the new centroid of each partition is added to S as a seed. Those documents which are not close enough to any of the seeds may be permanently discarded as outliers. The procedure returns the new set of seeds S, and the pruned set of documents D.

(2) **Project:** In each iteration, we project out the words with the least weight from the centroids of the previous iteration. This ensures that only the terms which are frequently occuring within a cluster of documents are used for the assignment process. The number of terms which are projected out in each iteration is such that the number of non-zero weight terms reduces by a geometric factor in each iteration. We denote this geometric factor by θ. The use of an iterative projection technique is useful in finding the words which are most representative of the subject material of a cluster. This is because in the first few iterations, when the clusters are not too refined, a larger number of dimensions need to be retained in the projection in order to avoid premature loss of information. In later iterations, the clusters become more refined and it is possible to project down to a smaller number of words.

(3) **Merge:** In each iteration, we merge all the clusters where the similarity of the seeds in the corresponding partitions is higher than a predefined value (denoted by *threshold*). The merging process is implemented using a simple single linkage method [6]. Each cluster is represented by a node in an undirected graph, and an edge is added between the two nodes if the similarity of the seeds of the corresponding clusters is larger than the predefined threshold value. Each connected component in this graph is then treated as a supercluster. In other words, the documents in each connected component are assigned to a single cluster. The set of centroids of this reduced set of clusters is returned by the procedure. The simple linkage process is somewhat naive, though fast and effective for high values of *threshold*.

(4) **Kill:** In each iteration, we discard all those seeds from S such that the number of documents in the corresponding clusters is less than a predefined number. This predefined parameter is denoted by *minimum* in Figure 1. These documents either get re-distributed to other clusters or get classified as outliers in later iterations.

These procedures are applied iteratively in order to create the clusters.

2.1 Categorization Algorithm

The definition of each cluster ensures that it is possible to categorize any test document very easily by assigning it to the class for which the corresponding seed is the closest. As in the case of the clustering, the cosine measure is used in order to perform the classification.

The first step in the algorithm is to find the k closest cluster seeds to the test document. The similarity of each cluster to the test document is calculated by using the cosine measure of the test document to the seed corresponding to each cluster. The value of k is a user-chosen parameter, and is typically a small number compared to the total number of nodes in the taxonomy. These k categories are the candidates for the best match, and may often contain a set of closely related subjects. This ranking process is designed to re-rank these categories more appropriately.

An important feature which we added to our categorization process was a method for distinguishing between very closely related subjects in a flat set of classes. This is required because even a supervised clustering technique may not provide perfect subject isolation, and a small percentage of the documents do get clustered with documents from a closely related (though slightly inaccurate) category. Even though a theoretical accuracy of 100% can be obtained by reporting the cluster label for the most similar seed, it may sometimes be desirable to correct for the errors in the clustering process by using a context-sensitive comparison method.

We build a *domination matrix* on a subset of the universe of categories, such that we know that all of these categories are good candidates for being the best match. As we will see, the simplicity of this process ensures that speed is not compromised by the use of the flat organization of clusters.

In order to understand the importance of distinguishing among closely related subjects, let us consider

the seeds for two nodes in the taxonomy: **Business Schools** and **Law Schools**. Recall that our process of projection limits the number of words in each seed to only words which are relevant to the corresponding categories. Some examples of words (with non-zero weights) which could be represented in the seed vector of each of these categories are as follows:

(1) **Business Schools:** business (35) , management (31), school (22), university (11), campus (15), presentation(12), student(17), market(11),....

(2) **Law Schools:** law(22), university (11), school (13), examination (15), justice (17), campus (10), courts (15), prosecutor (22), student (15) ...

The categories have numerous words in their seeds in common and thus, for example, a document about Law Schools with an unusually high number of mentions of "school" may appear close to Business Schools.

In order to establish the relative closeness of two categories to a given document more accurately, we need to ignore the contributions of the words common to both categories to the cosine measure. This is done by performing a *relative seed subtraction* operation on the seed vectors of each of the categories. The seed subtraction operation is defined as follows: Let S_1 and S_2 be two seed vectors. Then, the seed $S_1 - S_2$ is obtained by taking the seed S_1 and setting the weight of all those words which are common to S_1 and S_2 to 0.

We say that the seed S_1 dominates the seed S_2 under the following conditions:

• The (cosine) similarity of S_1 to the test document T is larger than the similarity of S_2 to T by at least a predefined threshold referred to as the *domination threshold.*

• The (cosine) similarity of S_1 to T is not larger than the similarity of S_2 to T by the predefined threshold, but the similarity of $(S_1 - S_2)$ to T is larger than the similarity of $(S_2 - S_1)$ to T.

The use of a domination threshold ensures that it is only possible to reorder seeds whose similarity to the test document are very close together. For each pair of the closest k seeds to the test document, we compute the domination matrix, which is the pairwise domination of each seed over the other. In order to rank order the k candidate seeds, we compute the *domination number* of each seed. The *domination number* of a seed is equal to the number of seeds (among the remaining $(k-1)$ seeds) that it dominates. The k seeds are ranked in closeness based on their domination number; ties are broken in favor of the original ordering based on cosine measure.

If there are a total of K classes created by the clustering algorithm, then the categorization algorithm needs to perform $O(K + k^2)$ cosine similarity calculations. Further, since the projected dimensionality of each seed is restricted to a few hundred words, each similarity calculation can be implemented efficiently. Thus, the categorization system is extremely fast because of its simplicity, and scales almost linearly with the number of classes.

Case	Percentage
Better than *Yahoo*!	8%
Not as good as *Yahoo*!	8%
Both were equally correct	77%
Neither is correct	6%
Unknown	1%

Table 1: Survey Results

3 Performance Results

As indicated earlier, we used a scan of the *Yahoo*! taxonomy from November, 1996. This taxonomy contained a total of 167,193 Web documents, over a lexicon of approximately 700,000 words. We truncated the *Yahoo*! tree taxonomy to obtain a set of 1,500 classes corresponding to intermediate level nodes. The purpose was to use the lowest level nodes in the taxonomy which contained at least 50 or more documents.

In our implementation of the supervised clustering algorithm we first calculated the normalized gini index of the different words in the clusters, and removed about 10,000 words with the highest gini index. We also removed the very infrequently occuring words in order to remove misspellings and creative variations on ordinary words. Specifically, we removed all those words which occured in less than 7 documents out of the original training data set of 167,193 Web documents. At this stage, we were left with a lexicon of about 77,000 words. The algorithm started with about 500 projected words in each seed, and successively removed words from the seed, until the desired maximum of about 200 words was obtained in each seed. The value of the seed reduction factor θ was 0.7. The value of the parameter *minimum* used to decide when to kill a cluster was 8. The value of the merging threshold (the parameter *threshold* in Figure 1) was 0.95. The algorithm required a total of 3 hours to complete on a 233 MHz AIX machine with 100 MB of memory. We obtained a total of 1,167 categories in a flat taxonomy.

We labeled the nodes by examining the constituent *Yahoo*! categories in this set of newly created clusters. Typically, the clustering did a very excellent job in grouping together documents from very closely related categories in the *Yahoo*! taxonomy in a creative way.

The simplicity of the classifier ensured that it was extremely fast in spite of the very large number of classes used. The classifier required about two hours to catego-

rize about 160,000 documents. This averaged at about 45 milliseconds per categorization. This does not include the time for parsing and tokenizing the document. In fact, the parsing procedure dominated the overall time for categorization (0.1 seconds for parsing). Since most text categorization techniques would need to parse the document anyway, this indicates that our categorization system is within a reasonable factor of the best possible overall speed of parsing and classification.

It is hard to provide a direct comparison of our technique to any other classification method by using a measure such as classification accuracy. This is because our algorithm does not use the original set of classes, but it actually defines the categories on its own. As discussed earlier, the accuracy of such a classifier is high, whereas the actual quality of categorization is defined by clustering quality. In addition it is also impossible to use the traditional synthetic data techniques (used for unsupervised clustering algorithms) in order to test the effectiveness of our technique. Thus, we need to find some way of quantifying the results of our clustering technique on a real data set such as *Yahoo*!.

We therefore used a survey in order to measure the quality of the clustering. We randomly sampled 141 documents from the clusters obtained by our algorithm, and asked respondents to indicate how well the corresponding subject labels defined it with respect to the $1,500$ *Yahoo*! categories described above. For each document, we asked respondents to indicate one of the five choices indicated in Table 1. The results are indicated in the same table.

One interesting aspect of the results in Table 1, is that the quality of our categorization was as good as *Yahoo*! for 77% of the documents. Out of this 77%, the two categorizations reported the same label in 86% of the cases, while the remaining label pairs were judged by the respondents to be qualitatively similar. Among the remaining documents, the opinions were evenly split (8%:8%) as to whether *Yahoo*! or our scheme provided a better categorization.

On only 7% of the questions did a *majority* of the respondents prefer one categorization to the other, and these were evenly split (3.5% : 3.5%) in prefering one of the two methods. In the remainder of the cases when the methods yield different answers, the respondents often rated the two labels as being equivalent, or as neither being correct. This presumably reflects the well-known difficulties of dealing with pages with multiple subjects, handling subjects not yet in the taxonomy, and of agreeing to the "correct" category from a large taxonomy with many closely-related categories.

It is important to understand that while our taxonomy is qualitatively comparable to Yahoo!, it is far more amenable to automated categorization than the manually built Yahoo! taxonomy. Any other categorizer, which trains on the *Yahoo*! taxonomy, would provide a second level of inaccuracy because of the difficulty in modeling manual categorizations. Thus, this scheme may be used in order to categorize large libraries of documents almost as effectively as manual categorizations such as *Yahoo*!, yet at a far lower cost.

4 Conclusions and Summary

In this paper, we proposed methods for building categorization systems by using supervised clustering. We also discussed techniques for distinguishing closely related classes in the taxonomy. We built such a categorization system using a set of classes from the *Yahoo*! taxonomy. We showed that the supervised clustering created a new set of classes which were surveyed to be as good as the original set of classes in the *Yahoo*! taxonomy, but which are naturally suited to automated categorization. The result is a system which has much higher overall quality of categorization than systems based on manual taxonomies.

Acknowledgements

The authors would like to thank the staff of the IBM Research Library for their participation in the survey.

References

[1] C. C. Aggarwal et. al. A framework for finding projected clusters in high dimensional spaces. *ACM SIGMOD Conference*, 1999.

[2] S. Chakrabarti et. al. Scalable feature selection, classification and signature generation for organizing text databases into hierarchical topic taxonomies. *VLDB Journal*, 7:163-178, 1998.

[3] D. R. Cutting, et. al. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. *ACM SIGIR Conference*, 1992.

[4] D. Koller, M. Sahami. Hierarchically classifying documents using very few words. *International Conference on Machine Learning,* volume 14, Morgan-Kaufmann, July 1997.

[5] W. Lam, C. Y. Ho. Using a Generalized Instance Set for Automatic Text Categorization. *ACM SIGIR Conference*, 1998.

[6] R. Sibson. SLINK: An optimally efficient algorithm for the single link cluster method. *Computer Journal*, Volume 16, pages 30-34, 1973.

[7] H. Schutze, C. Silverstein. Projections for efficient document clustering. *ACM SIGIR Conf.*, 1997.