RC23517 (C0502-003) February 7, 2005
Computer Science

IBM Research Report

Automatic Generation of Hierarchical Taxonomies for Text Categorization

Li Zhang, Tao Li, Shi Xia Liu, Yue Pan
IBM Research Division
China Research Laboratory
HaoHai Building, No. 7, 5th Street
ShangDi, Beijing 100085
China

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Automatic Generation of Hierarchical Taxonomies for Text Categorization

ABSTRACT

Although considerable research has been conducted in the field of hierarchical text categorization, little has been done on automatically collecting labeled corpus for building hierarchical taxonomies. In this paper, we propose an automatic method of collecting training samples for building hierarchical taxonomies. In our method, the category node is initially defined by some keywords, the web search engine is then used to construct a small set of labeled documents, and topic tracking algorithm based on document length normalization is applied to enlarge the training corpus on the bases of the seed documents. We also design a method to check the consistency of the collected corpus. The above steps produces a flat category structure which contains all the categories for building the hierarchical taxonomy. Next, linear discriminant projection approach is utilized to construct more meaningful intermediate levels of hierarchies in the generated flat set of categories. Experimental results show that training corpus is good enough for statistical classification methods.

1. INTRODUCTION

An important component of business intelligence for many enterprises is to keep track of the market and competitor information. Generally the enterprises need to organize the information into a hierarchical category tree, and automatically collect the latest information from the web and document pools.

Text categorization, as a fundamental and effective tool that can automatically classify all kinds of documents into predefined categories, has been receiving much attention and numerous approaches have been developed in the literature [21, 17, 5, 14, 19, 15, 33, 25]. Experimental results show that the classification accuracy achieved by automatic approaches is as good as human performance and thus makes text categorization an attractive technique for information organization [26]. However, in practice, the performance of the classification methods depends on the number of available training samples. Test with reuters-21578 [34] shows that the classification precision and recall are good with common categories (with more than 300 training samples), but poor with rare categories (sometimes with less than 10 training samples). Furthermore, the classification accuracy depends on the quality of user labeled corpus. If the corpus is poorly labeled, the classification accuracy will decrease greatly.

Generally speaking, statistical machine learning methods need a large high-quality training corpus to train the classifier. So, how to prepare training corpus efficiently and effectively is a big problem that the enterprise must resolve. Several attempts have been made to solve this problem. Among them, the most commonly used method is active learning [9, 18, 27]. This technique is well suited to the cases where the enterprise has a large unlabeled corpus. In active learning, the machine prompts the user to label the most informative document for the classifier to learn. Human can operate interactively to label the samples. Active learning can significantly reduce the number of training samples and achieve a reasonable classification quality. But in many cases, the enterprise does not have such a large scale of corpus. A computer-aided tool is needed to help them gather corpus from the web to build the enterprise taxonomy.

In order to build enterprise taxonomies efficiently and effectively, the following issues should be carefully studied. 1) how to collect large corpus from the web efficiently and document pools; 2) how to evaluate the quality of the training corpus; 3) how to organize the collected corpus into hierarchical structure.

To address these issues, we design an easy and convenient system to assist the enterprise to prepare the large training corpus from the web and document pools. At the beginning, the technicians of the enterprise can define the flat category structure freely. Then they can submit some keywords to the search engine to get some search results. Our system can automatically analyze the documents and find seed documents for each category node. The keywords of each category are refined by the seed documents and can be used in document length normalization. Next, the document length normalization and topic tracking technique [8] are used to collect good training samples from the web and document pools according to several predefined file pathes. A consistency checking method can be utilized to help the user check the quality of the corpus. And finally the hierarchy of the taxonomy can be generated via linear discriminant projection. Experimental results show that the training corpus is good enough for statistical classification methods.

The remainder of the paper is organized as follows. Section 2 describes the method to generate a flat category structure. Section 3 presents the linear discriminant projection approach to construct more meaningful hierarchical category tree. Section 4 shows the experiments and results. Section 5 discusses related work. Final conclusions are given in section 6.

2. GENERATION OF A FLAT CATEGORY STRUCTURE

In this section, we present in detail our methods of selecting training samples from the web and document pools. Section 2.1 provides an overview of the selection procedure, Section 2.2 and Section 2.3 describe describe seed documents selection and large corpus collection respectively. Section 2.4 introduces the consistency checking method.

2.1 An Overview of the Selection Procedure

Training samples are collected by two steps: seed documents selection and large corpus collection. The first step is to construct several seed documents. Keywords are first submitted to a search engine and search results are downloaded (via a crawler) into the local system. Next, automatic seed selection method is employed to choose good documents as seeds from the search results. Then, relevance feedback technique is applied to extract many relevant terms. The relevant terms are utilized to enlarge the keyword set for the category.

In the second step, the seed documents are used to discover more relevant documents from the web and document pools. The enlarged keyword set is utilized to normalize the length of seed documents and documents being processed. Given the web URL set and document pools the enterprise is interested in, the crawler will crawl the web and download the documents to the local file system. Topic tracking algorithm, based on document length normalization is used to detect whether the incoming document is related to the corresponding topic. And finally related documents will be saved as training samples.

To guarantee the quality of the training corpus, after the large corpus is collected, a consistency checking method is used to help the enterprise technicians check the training data.

2.2 Seed Construction & Keyword Generation

Relevance feedback has been proved to be an effective way for query reformulation in information retrieval [2]. Its basic idea is to analyze the terms in each relevant and non-relevant documents, and enhance the weight of terms in relevant documents and weaken the weight of the terms in non-relevant documents.

Salton and Buckley compare twelve different feedback procedures in their paper [24]. For the vector feedback methods, three methods were tested:

$$\text{Rocchio Regular: } Q_1 = Q_0 + \beta\frac{1}{n_1}\sum_{i=1}^{n_1} R_i - \gamma\frac{1}{n_2}\sum_{j=1}^{n_2} U_j \quad (1)$$

$$\text{Ide Regular: } Q_1 = Q_0 + \sum_{i=1}^{n_1} R_i - \sum_{j=1}^{n_2} U_j. \quad (2)$$

$$\text{Ide dec-hi: } Q_1 = Q_0 + \sum_{i=1}^{n_1} R_i - U_j. \quad (3)$$

where Q_0 is the vector for the initial query, R_i is the vector for relevant document i, U_j is the vector for non-relevant document j, n_1 is the number of relevant documents, n_2 is the number of non-relevant documents.

In our system, instead of reconstructing the query, relevance feedback is used to refine the category's centroid vector in seed selection, and to enlarge keywords related to the category.

For each category, some query words can be submitted to a search engine, but the search results can not be treated as seed documents directly, because the search engine's performance is a common problem, many non-relevant documents are often mixed with good documents. Human being can mark good documents one by one, but it is somewhat time consuming and in many cases, the samples selected by human may not be the optimal. We propose an automatic seed selection method to select good documents from the search results.

Suppose C_1,C_2,...,C_n are n categories, and D_1,D_2,...,D_n are search result document sets for each category. For each category C_i, the seeds are selected as follows:

1. compute the representative category vector V_i, where V_i can be computed with two ways, one is the centroid method, the other is the refined centroid method via relevance feedback equation (1);
2. compute the similarity of each documents $d_{i,j} \in Di$ with the vector V_i and sort the documents in descending order of the similarity values.
3. top m documents are selected as seed documents.

In our system, each category is initially described by several keywords which are used to normalize document length in the next step. Initially, the keyword is the category name by default. While the user submits a query to search engine and then selects some items to download, the query words are automatically added into the keyword set. After the seed documents are selected, relevance feedback technique is employed to analyze the documents and enlarge the keyword set.

For each category, term weight is computed according to the above query refinement formulas. The category's seed documents are treated as relevant documents, the other categories' seeds are non-relevant documents. The terms are then sorted in descending order of their weights. Top k terms in the list are added to category's keyword set. These keywords will be used to normalize the document length in topic tracking phase.

2.3 Document Length Normalization Based Topic Tracking

2.3.1 Topic Tracking

Topic tracking is one of the TDT research tasks that try to discover topically related material in news streams [30]. The goal of topic tracking is to keep track of incoming documents similar to several predefined example documents. Typically the tracking system is given 1-4 documents, called seed documents, and asked to monitor the news stream of future document on the same topic. In our system, by searching the web, the user can collect several seed documents using the method in 2.2, then topic tracking technique is applied to gather large number of relevant documents.

Our topic tracking algorithm is based on that described in [8]. The document-document similarity function is based on a symmetrized version of the Okapi formula [22]:

$$Ok(d^1, d^2; cl) = \sum_{w \in (d^1 \cap d^2)} t_w^1 t_w^2 idf(w, cl). \quad (4)$$

t_w^i is the adjusted term frequency of word w in document i,

$$t_w^i = \frac{\hat{t}_w^i}{\alpha + \hat{t}_w^i}, \text{ where } \hat{t}_w^i = \frac{c_w^i}{\sum_w c_w^i}. \quad (5)$$

and $idf(w, cl)$ is a cluster-dependent weight,

$$idf(w, cl) = idf_0(w) + \lambda\frac{2n_{w,cl}}{n_w + n_{cl}}, \quad (6)$$

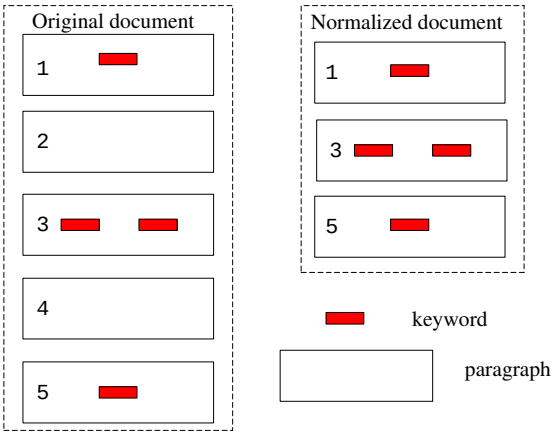

Figure 1: Document Length Normalization

where $idf_0(w)$ is the traditional inverse document frequency, n_w is the number of documents that contain word w, n_{cl} is the number of documents in cluster cl, $n_{w,cl}$ is the number of documents in the cluster which contain the word; λ is an adjustable parameter.

The cluster is represented by the centroid of the documents it contains, then the similarity score of a document with the cluster is given by

$$Sim(d, cl) = |cl|^{-1} \sum_{d' \in cl} Ok(d, d'; cl). \quad (7)$$

Because the content of a cluster changes over the course of time and its centroid is updated dynamically, the scoring formula performs well in the topic tracking task.

2.3.2 *Document Length Normalization*

We construct a classifier based on the above topic tracking algorithm. This classifier is utilized to get training samples for the category node which is described by several keywords and seed documents. However, since the documents are from the web and document pools, their content lengths are diversely different, this significantly influences the classification accuracy. In this section, we propose a keyword-based document length normalization and use it to improve topic tracking algorithm. The main idea of this method is to use the keyword set $K = \{k_1, k_2, ..., k_p\}$ to normalize the length of the seed document and the document being classified. Its major objective is to remove noises from the document, and thus to increase the accuracy of classification. Compared with the method proposed by Singhal et al., our method allows the user to customize the text categorization according to his preference. This is achieved by allowing the user to define the keywords for the keyword-based document length normalization.

Figure 1 illustrates the keyword-based document length normalization. This method first finds the given keywords in the document. Then the paragraph which contains the matched keyword(s) is extracted. Those extracted paragraphs are combined together to construct the normalized document of the original one. Based on the keyword-based document length normalization, we improve the above topic tracking algorithm. The improved method consists of the following steps:

1. Construct the normalized document for each seed document;
2. For each incoming document, construct its normalized one;
3. Apply tracking algorithm on these normalized documents.

Compared with the traditional topic tracking method, our method improves the performance of classification. The comparison between these two methods is illustrated by the experimental results shown in Table 5.

2.4 Consistency Checking

With several seed samples, topic tracking can collect similar documents according to document's time sequence. When a large corpus is gathered, we must make sure that all the documents are consistent with the category tree. Aggarwal et al. [1] used dominate matrix method to distinguish between very closely related topics in category tree. Different from their method, we use Shannon entropy to measure the data consistency.

In machine learning area, researchers have observed that it is a good way to determine the final category on a new document by inducing multiple classifiers and picking the answer with the most votes [27, 32]. In our system, we consider two classifiers: one is topic tracking, and the other is the traditional k-means clustering method. We first run k-means clustering without time sequence, and then compare the k-means result with the topic tracking result. There are two problems we want to detect: the first one is whether the child category under a certain category are well constructed, the second one is whether the documents in a particular category are mislabeled to another category.

Suppose node C is a category in the category tree, its child nodes are $C_1, ..., C_i, ..., C_n$, the document set D is classified into the child categories via the topic tracking method. K-means clustering method is used to run clustering on the document set D, cluster number is the child node number n. We use Shannon entropy measure to compare the result of the k-means clustering method with that of the topic tracking method. Shannon entropy based measure [28] is a popular method to evaluate the quality of the clustering algorithms. It asserts that the best entropy that can be obtained is when each cluster contains the optimal number of members. In the ideal case, the k-means clustering method result is the same with the topic tracking result, this means the category C is well constructed. If these two results are not identical, then the documents labeled differently are candidates for checking.

According to shannon entropy theory, for cluster S_j, the category distribution p_{ij} is the probability that a member in cluster S_j belongs to category C_i:

$$p_{ij} = \frac{|C_i \cap S_j|}{n_j}, \quad (8)$$

where n_j is the document number of the cluster S_j. The entropy of every cluster S_j is calculated using the standard entropy formula:

$$E_j = -\sum_{i=1}^{n} p_{ij} \log p_{ij}. \quad (9)$$

The total entropy is then calculated as

$$E = \frac{1}{n} \sum_{j=1}^{n} n_j * E_j. \quad (10)$$

The lower the entropy, the better the clustering result. We set a threshold, when the entropy is greater than the threshold, the specified category C 's structure should be inspected. The following steps are executed to detect which samples are not well labeled. For each k-means cluster S_j, we find its most matching category node C_i whose documents are collected by topic tracking algorithm. Then in C_i, the documents which do not belong to cluster S_j are picked out and presented to the user for checking.

3. AUTOMATIC HIERARCHY GENERATION VIA LINEAR DISCRIMINANT PROJECTION

The first step in automatic hierarchy generation is to infer class relationships (e.g., measure the similarities between categories). The similarity should reflect the intrinsic characteristics of the categories and provide dependence relationships for efficient learning methods. Once the similarity measure has been determined, a hierarchical clustering method can be used to build the hierarchy.

One simple way to infer class relationships is to compare the class representatives. As we all know, however, the data dimension is very high in vector space model for document analysis. It has been shown that in a high dimensional space, the distance between every pair of points is almost the same for a wide variety of data distributions and distance functions due to the *curse of dimensionality* [3]. In our work, we utilize linear discriminant projection for generating more meaningful intermediate levels of hierarchies in large flat sets of categories. Linear discriminant projection approach first transforms all the documents onto a low-dimensional space and then clusters the categories into hierarchies according to the distances of the centroids of each category in the transformed space. Discriminant analysis approaches are well known to learn discriminative feature transformations in statistical pattern recognition literature [10]. Fisher discriminant analysis [7] finds discriminative feature transform as eigenvectors of matrix $T = S_w^{-1}\S_b$ where S_w is the intra-class covariance matrix and S_b is the inter-class covariance matrix. Basically T captures both compactness of each class and separations between classes and hence eigenvectors corresponding to largest eigenvalues of T would constitute a discriminative feature transform. The transformed feature space would reflect the inherent similarity structure between the classes.

3.1 Linear Discriminant Projection Approach

In this section, we briefly describe the linear discriminant projection approach for inferring class relationships. Its core idea is to compare the class representatives in a low-dimensional space so that the comparison is more "meaningful". More specifically, after finding the transformation, the similarity between classes is defined to be the distance between their centroids in the transformed spaces. The notations used later in the discussion are listed in the Table 1.

Notations	Descriptions
A	document-term matrix
n	number of data points, i.e., documents
N	number of the dimensions, i.e, terms
k	number of class
S_i	covariance matrix of the i-th class
S_b	between-class scatter matrix
S_w	within-class scatter matrix
G	reduction transformation
m_i	centroid of the i-th class
m	global centroid of the training set

Table 1: Notations

3.1.1 Finding the Transformation

Given a document-term matrix $A = (a_{ij}) \in \Re^{n\times N}$, where each row corresponds to a document and each column corresponds to a particular term, we consider finding a linear transformation $G \in \Re^{N\times \ell}$ ($\ell < N$) that maps each row a_i ($1 \le i \le n$) of A in the N-dimensional space to a row y_i in the ℓ-dimensional space. The resulting data matrix $A^L = AG \in \Re^{n\times \ell}$ contains ℓ columns, i.e. there are ℓ features for each document in the reduced (transformed) space. It is also clear that the features in the reduced space are linear combinations of the features in the original high dimensional space, where the coefficients of the linear combinations depend on the transformation G. Linear discriminant projection tries to compute the optimal transformation matrix G such that the class structure is preserved. More details are given below.

Assume there are k classes in the data set. Suppose m_i, S_i, P_i are the mean vector, covariance matrix, and a prior probability of the i-th class, respectively, and m is the total mean. For the covariance matrix S_i for the ith class, we can decompose it as $S_i = X_i X_i^T$, where X_i has the same number of columns as the number of data points in the i-th class. Define the matrices

$$H_b = [\sqrt{P_1}(m_1 - m), \cdots, \sqrt{P_k}(m_k - m)] \in \Re^{N\times k},$$
$$H_w = [\sqrt{P_1}X_1, \cdots, \sqrt{P_k}X_k] \in \Re^{N\times n}.$$

Then the between-class scatter matrix S_b, the within-class scatter matrix S_w, and the total scatter matrix S_t are defined as follows [10]:

$$S_b = \sum_{i=1}^{k} P_i(m_i - m)(m_i - m)^T = H_b H_b^T,$$
$$S_w = \sum_{i=1}^{k} P_i S_i = H_w H_w^T.$$

In the lower-dimensional space resulting from the linear transformation G, the within-cluster and between-cluster matrices become

$$S_w^L = (G^T H_w)(G^T H_w)^T = G^T S_w G,$$
$$S_b^L = (G^T H_b)(G^T H_b)^T = G^T S_b G.$$

An optimal transformation G would maximize Trace(S_b^L) and minimize Trace(S_w^L). A common optimization for computing optimal G is

$$G^* = \arg\max_G \text{Trace}\left(\left(G^T S_w G\right)^{-1} G^T S_b G\right).$$

The solution can be readily obtained by solving a eigenvalue decomposition problem on $S_w^{-1}S_b$, provided that the within-class scatter matrix S_w is nonsingular. Since the rank of the between-class scatter matrix is bounded above by $k-1$, there are at most $k-1$ discriminant vectors.

3.1.2 Extension on General Cases

In general, the within-class scatter matrix S_w may be singular especially for document-term matrix where the dimension is very high. A common way to deal with it is to use generalized eigenvalue decomposition [11, 16]

Let $K = [H_b\ H_w]^T$, which is a $k+n$ by N matrix. By the generalized singular value decomposition, there exist orthogonal matrices $U \in \Re^{k\times k}$, $V \in \Re^{n\times n}$, and a nonsingular matrix $X \in \Re^{N\times N}$, such that

$$\begin{bmatrix} U^T & 0 \\ 0 & V^T \end{bmatrix} KX = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \\ \Sigma_2 & 0 \\ 0 & 0 \end{bmatrix}, \qquad (11)$$

where

$$\Sigma_1 = \text{diag}(\overbrace{1, \cdots, 1}^{r}, \alpha_1, \cdots, \alpha_s \overbrace{0, \cdots, 0}^{t-r-s}),$$

$$\Sigma_2 = \text{diag}(\overbrace{0, \cdots, 0}^{r}, \beta_1, \cdots, \beta_s, \overbrace{1, \cdots, 1}^{t-r-s}),$$

$$t = \text{rank}(K), \quad r = t - \text{rank}(H_w^T),$$

$$s = \text{rank}(H_b) + \text{rank}(H_w) - t,$$

satisfying

$$1 > \alpha_1 \geq, \cdots, \geq \alpha_s > 0,$$

$$0 < \beta_1 \leq, \cdots, \leq \beta_s < 1,$$

$$\text{and} \qquad \alpha_i^2 + \beta_i^2 = 1 \quad \text{for} \quad i = 1, \cdots, s.$$

From Eq. (11), we have

$$(X^T H_b)(X^T H_b)^T = \begin{bmatrix} \Sigma_1^T \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix},$$

$$(X^T H_w)(X^T H_w)^T = \begin{bmatrix} \Sigma_2^T \Sigma_2 & 0 \\ 0 & 0 \end{bmatrix}.$$

Hence a natural extension of the proposed linear discriminant projection in Section 3.1.1 is to choose the first $q = r + s$ columns of the matrix X in Eq. (11) as the transformation matrix G^*.

3.2 Defining the Similarity and Hierarchy Generation

After finding the transformation G, we define the similarity between classes to be the distance between their centroids in the transformed spaces. In other words, two categories are similar if they are "close" to each other in the transformed space. The linear discriminant projection finds the transformation that preserves the class structure by minimizing the sum of squared within-class scatter while maximizing the sum of squared between-class scatter and hence the distances in the transformed space should be able to reflect the inherent structure of the dataset.

After obtaining the similarities/distances between classes, we use the Hierarchical Agglomerative Clustering (HAC) algorithm of [12] to generate automatic topic hierarchies from a given set of flat classes. The result of hierarchical clustering is a dendrogram where similar classes are organized into hierarchies. We choose UPGMA (Unweighted Pair-Groups Method Average method), which is known to be simple, efficient and stable [12]. In UPGMA, the average distance between clusters is calculated from the distance between each point in a cluster and all other points in another cluster. The two clusters with the lowest average distance are joined together to form the new cluster.

4. EXPERIMENTS

In this section, we introduce some detailed information about system implementation. Then, experimental results are reported. In the experiments, we study the following issues: the effectiveness of automatic seed selection method, performance evaluation for document length normalization based topic tracking, and data quality checking. Finally a sample of hierarchical taxonomy built by our method is presented.

4.1 System Implementation

In our implementation, each individual system component is developed independently and works together in a work flow. The followings are some detailed information.

1. The crawler is multi-threaded and runs in a mode that can be configured by a configuration file; The configuration file specifies the depth of the crawling, the pattern that the downloaded page's URL must satisfy, and etc. With this crawler, the system can monitor several web sites, by using a "start" URL and a value for depth (in web pages).

2. The content extractor transforms all kinds of document format into plain text. It also removes unrelated content from the downloaded web page, such as advertisements and menu links.

3. The duplication detector maintains both a memory index and a hard disk index for duplication computation, which makes it fast for processing large collection of documents.

4. Document summarization engine extracts important sentences based on the analysis of the document structure, such as its title, authors, sections, and paragraphs. In the summarization result, the main document structure and consecutive sentences are kept, which makes it more readable.

4.2 Seed Document Selection

We use top ten categories from reuters-21578 dataset to test our automatic seed selection method. The query words for each category are generated simply from the category description file in the package, which are illustrated in table 2. The query words are transformed into lower case and no stemming are made while searching in the dataset, all of the documents containing any of the query words are selected as seed candidates. After selecting seeds, their original labels are used to evaluate the seed is good or not. We use precision to evaluate the seed selection accuracy.

Table 2: Query Words

Category	Keyword(s)
earn	Earnings, Forecasts
acq	Mergers, Acquisitions
money-fx	Money,Foreign, Exchange
crude	Crude, Oil
grain	Grain
trade	Trade
interest	Interest, Rates
heat	Heating, Gas, Oil
wheat	wheat
ship	Shipping

We designed two baselines. One baseline is the average ratio of the right seed documents in the candidate documents for each category. The average ratio of the top ten categories is 37%. This indicates that most of the search results are not suitable as seed documents, although they contain the query words. Table 3 reports the performance of our automatic method. The result shows that our method is significantly better than the baseline. Seed selection with refined centroid method has the best performance, it reach 82% with 5 seeds per category and 85% with 10 seeds per category respectively. The experimental results also illustrates that the refined centroid method improves greatly the traditional centroid method by nearly 10 percent, this is mainly because by using relevance feedback, the weights of the words most frequently appeared in non-relevant documents are decreased so that the refined centroid vectors reflect the characteristics of the topics.

Another baseline is, in the search result, each candidate document is ranked according to its similarity with query vector, just

Table 3: Seed Selection Performance

Method	5 seeds	10 seeds
centroid	0.72	0.76
refined centroid	0.82	0.85

simulating the search result returned by a search engine. In our experiments, we simply use the traditional cosine similarity formula to rank the candidate documents with the query vector. With this baseline, top 10 of the ranking list are treated as seed documents. The average accuracy is 87%, which is much better than baseline 1, because the ranking is a good indicator for the topical documents.

In order to compare with baseline 2, we use our automatic seed selection method to select top 5 good seeds out of the 10 seeds generated by baseline 2. Table 4 shows the performance of our method. The result shows that even with ranking list of the candidate documents, our method can also improve the accuracy of the seeds.

Table 4: Seed Selection Performance from Ranking List

Method	Accuracy
centroid	0.88
refined centroid	0.90

4.3 Performance Evaluation

In order to investigate the effectiveness of our method in collecting training samples, we use ModeApte split of reuters-21578 dataset to do the performance evaluation. In the ModeApte training set, there are totally 9,604 documents, and 7,775 documents are labeled. Among the 7,775 documents, 6,552 documents are single labeled and this leads to 54 categories that have more than one sample in the training set. Without loss of generality, we suppose the category tree is constructed with these 54 nodes and the tree structure is flat.

We compare our classification method with SVM algorithm [20]. Micro-average precision, recall and F_1 measure are used to evaluate the classification performance. In our experiments, each time n documents are selected for each category from the 6,552 documents. These n documents are seeds for topic tracking and training samples for SVM algorithm as well. Because the training samples in ModeApte data set are not uniformly distributed, the total document number does not increase linearly with n.

In document length normalization, 20 keywords are selected to filter the document content. The three formulas described in section 2.2 are used to compute the keyword weight respectively. In formula 3, the non-relevant document is ignored in our experiments.

In order to investigate how the document length normalization influences the classification result, we also run the standard topic tracking algorithm without document length normalization.

All the results are illustrated in Table 5 and Figure 2. Here, "TR" represents topic tracking method. DLN refers to "Document Length Normalization", and DLN 1, 2, 3 represent Rocchio Regular, Ide Regular and Ide dec-hi method respectively.

Table 5 shows that with small labeled corpus, our method can achieve better performance than SVM method. Especially, with 5 training samples for each category and using document length normalization with Ide Regular formula, the performance of our method is better than that of SVM algorithm with 30 training documents for each category. This is mainly due to the incremental updating of the cluster centroid and dynamic term weighting scheme.

It follows from the comparison result in Table 5 that the document length normalization based topic tracking improves the classification accuracy greatly. We can further concludes from Table 5 that the method of Ide Regular method performs better than the other two methods.

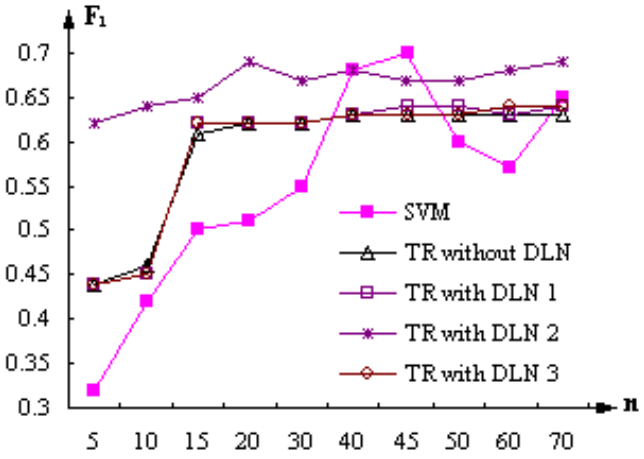

Figure 2: Performance evaluation results

4.4 Data Quality Checking

In this experiment, we choose ten categories from the reuters-21578 training set, each category has 5 seeds. Test data are those belonging to only one of these ten categories in the ModApte test set, we get 114 documents. We run topic tracking without document length normalization, which means the categorization has weak classification performance. After that, we run k-means clustering on the same test dataset. Cluster number is ten. The Shannon entropy is listed in Table 6.

Table 6: Shannon Entropy Measure of K-means Clustering

Cluster Num	Label	Entropy
0	livestock	1.0478
1	gold	0.5004
2	ipi	0.2071
3	sugar	0.5864
4	iron-steel	0.7848
5	natgas	0.4142
6	gold	0.0000
7	bop	0.3986
8	sugar	0.5004
9	heat	0.0000
Total Entropy		0.4180

For each cluster, we label the documents it contains to the category that the cluster is mostly matched. Then we compare the cluster labels with the topic tracking labels, this leads to 33 documents to be checked. Among these 33 documents, only 4 documents are correctly labeled by topic tracking, and the other 29 documents are incorrectly labeled. This shows that our consistency checking method is very useful to refine the quality of the corpus.

4.5 Hierarchy generation

A sample taxonomy built by our tool is a computer market intelligence category tree for a customer that provides information technology consulting service. The hierarchical category tree contains 47 nodes. The topics of the categories include macro environ-

Table 5: Performance Evaluation Results

Method	n=5, doc=246			n=10, doc=439			n=15, doc=598			n=20, doc=728			n=30, doc=950		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
SVM	0.23	0.54	0.32	0.36	0.49	0.42	0.44	0.58	0.50	0.44	0.60	0.51	0.47	0.65	0.55
TR without DLN	0.39	0.51	0.44	0.41	0.53	0.46	0.60	0.63	0.61	0.60	0.63	0.62	0.60	0.65	0.62
TR with DLN 1	0.39	0.50	0.44	0.41	0.52	0.45	0.62	0.63	0.62	0.60	0.63	0.62	0.62	0.63	0.62
TR with DLN 2	0.62	0.61	0.62	0.67	0.62	0.64	0.66	0.63	0.65	0.72	0.67	0.69	0.71	0.64	0.67
TR with DLN 3	0.39	0.50	0.44	0.40	0.51	0.45	0.61	0.63	0.62	0.60	0.63	0.62	0.61	0.63	0.62
Method	n=40, doc=1120			n=45, doc=1191			n=50, doc=1257			n=60, doc=1380			n=70, doc=1490		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
SVM	0.62	0.74	0.68	0.66	0.74	0.70	0.54	0.68	0.60	0.50	0.67	0.57	0.59	0.72	0.65
TR without DLN	0.62	0.65	0.63	0.62	0.64	0.63	0.62	0.64	0.63	0.61	0.64	0.63	0.62	0.64	0.63
TR with DLN 1	0.62	0.64	0.63	0.63	0.64	0.64	0.63	0.64	0.64	0.62	0.64	0.63	0.63	0.64	0.64
TR with DLN 2	0.71	0.65	0.68	0.71	0.65	0.67	0.70	0.65	0.67	0.71	0.66	0.68	0.72	0.67	0.69
TR with DLN 3	0.62	0.64	0.63	0.62	0.64	0.63	0.63	0.64	0.63	0.63	0.64	0.64	0.63	0.64	0.64

ment (such as economy, politics, regulations, and technologies), information technology products and services (such as PC, laptop, server, storage, IT service, networking and telecommunications, software, peripheral, and etc.), and vertical industry (such as construction, education, consumer, media, transportation, health care, and etc.). All the samples are from the web, total 1901 documents are gathered for this category tree.

Figure 3 shows the hierarchies of the subtree of this taxonomy. Each block represents the similarity between the corresponding row and column categories. The darker the color is, the more similar the categories are. We can observe from the dendrogram that most of the semantic similarity of categories is reasonable.

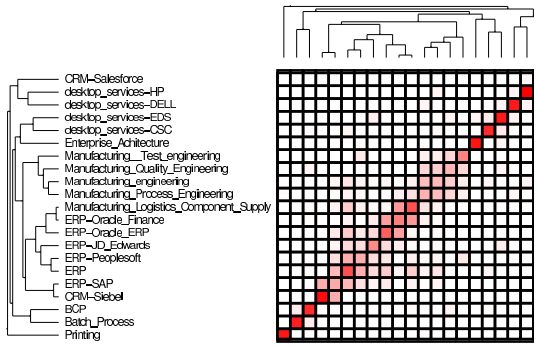

Figure 3: Hierarchy Generation on the Example Dataset

5. RELATED WORK

Training dataset is very important for text categorization. However, good test collections are very scare by now. This is because constructing a new dataset for text categorization requires huge manual effort to label the documents. Recently several research papers have been focused on generating the labeled datasets automatically.

Davidov et al. [4] describe a system for automatically acquiring labeled datasets for text categorization from the World Wide Web by capitalizing on the existing hierarchical directory structures such as the Open Directory. They define parameters to control the difficulty of the generated datasets for categorization. However, their dataset generation does not consider the content of the datasets.

Our work also share some commonalities with clustering and summarizing web search results [23, 13, 35, 36, 31, 29, 6]. Their methods try to group the search results into clusters and provide easy access and browsing ways for user to get information. Our method is different with them at picking good seed documents from the search results and building hierarchical taxonomies.

We compare our method with other similar taxonomy building methods in terms of pre-existing data, taxonomy and human labor needed. Table 7 shows that our method focuses on building taxonomies without pre-existing taxonomy and data. Compared with the other two methods, less efforts are needed by our method.

6. CONCLUSIONS

In this paper, we present an automatic method of collecting training samples for building hierarchical taxonomies, which can help the enterprise prepare training samples for text categorization task. The main characteristic of this method is that it can start with several keywords and gather high quality large hierarchical corpus automatically. Experimental results show that our automatic seed selection method is effective in selecting good documents. Our document length normalization based topic tracking method can achieve better performance than traditional classification algorithm, especially in the case of small training corpus. And our consistency checking method is effective to guarantee the quality of the data. Furthermore, the generated hierarchy taxonomies improve classification performance in most times.

7. REFERENCES

[1] C. C. Aggarwal, S. C. Gates, and P. S. Yu. On the merits of building categorization systems by supervised clustering. In *Proceedings of KDD-99, 5th ACM International Conference on Knowledge Discovery and Data Mining*, pages 352–356, 1999.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison& Wesley, 1999.

[3] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbor meaningful? In *ICDT Conference*, 1999.

[4] D. Davidov, E. Gabrilovich, and S. Markovitch. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In *Proceedings of The 27th Annual International ACM SIGIR Conference*, pages 250–257, Sheffield, UK, 2004. ACM Press.

Table 7: Taxonomy Building Methods Comparison

Method	pre-existing data	pre-existing taxonomy	human labor	
			initial corpus	quality checking
our method	No	No	Low	Low
Aggarwal	Labeled	Yes	Middle	Low
Active Learning	Unlabeled	No	Middle	-

[5] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM 98*, pages 148 – 155, 1998.

[6] P. Ferragina and A. Gulli. The anatomy of a clustering engine for web, books, news snippets. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 395–398, 2004.

[7] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, (7):179–188, 1936.

[8] M. Franz, J. S. McCarley, T. Ward, and W.-J. Zhu. Unsupervised and supervised clustering for topic tracking. In *SIGIR'01*, pages 310–317, 2001.

[9] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[10] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic Press, New York, 2nd edition, 1990.

[11] P. Howland and H. Park. Generalizing discriminant analysis using the generalized singular value decomposition. *IEEE PAMI*, 26(8):995–1006, 2004.

[12] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[13] Z. Jiang, A. Joshi, R. Krishnapuram, and L. Yi. Retriever: Improving Web Search Engine Results Using Clustering. Technical report, University of Maryland Baltimore County, October 2000.

[14] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, 1998.

[15] D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In C. Nédellec and C. Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[16] T. Li, S. Zhu, and M. Ogihara. Efficient multi-way text categorization via generalized discriminant analysis. In *ACM CIKM*, pages 317–324, 2003.

[17] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48, 1998.

[18] A. K. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In J. W. Shavlik, editor, *Proceedings of ICML-98, 15th International Conference on Machine Learning*, pages 350–358, Madison, US, 1998. Morgan Kaufmann Publishers, San Francisco, US.

[19] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *In IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

[20] D. Pavlov, J. Mao, and B. Dom. Scaling-up support vector machines using boosting algorithm. In *15th International Conference on Pattern Recognition (ICPR 2000)*, pages 219–222, 2000.

[21] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[22] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *The 3d Text REtrieval Conference (TREC-3)*, 1995.

[23] D. G. Roussinov and H. Chen. Information navigation on the web by clustering and summarizing query results. *Information Processing and Management*, 37(6):789–816, 2001.

[24] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41:288–297, 1990.

[25] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

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

[27] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, 1992.

[28] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[29] Y. Wang and M. Kitsuregawa. Link-based clustering of web search results. In *Proceedings of the Second International conference on Web-Age Information Management (WAIM'2001)*, 2001.

[30] C. L. Wayne. Multilingual topic detection and tracking: Successful research enabled by corpora and evaluation. In *Language Resources and Evaluation Conference (LREC) 2000*, pages 1487–1494, 2000.

[31] D. Weiss. Introduction to search results clustering. In *Proceedings of the 6th International Conference on Soft Computing and Distributed Processing, Rzeszów, Poland*, 2002.

[32] S. M. Weiss, C. Apte, F. J. Damerau, D. E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. Maximizing text-mining performance. *IEEE Intelligent Systems*, 14(4):2–8, 1999.

[33] E. D. Wiener, J. O. Pedersen, and A. S. Weigend. A neural network approach to topic spotting. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 317–332, Las Vegas, US, 1995.

[34] Y. M. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR'99*, pages 42–49, 1999.

[35] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to web search results. In *Proceedings of the Eighth International World Wide Web Conference*, 1999.

[36] D. zhang and Y. Dong. Semantic, hierarchical, online clustering of web search results. In *Proceedings of the 6th Asia Pacific Web Conference (APWEB)*, 2004.