Hypertext Information Retrieval for Short Queries

Chia-Hui Chang and Ching-Chi Hsu
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan 106
E-mail: {chia, CCHsu} @ails1.csie.ntu.edu.tw

Abstract

Keyword-based query model has been an immediate and efficient way to specify and retrieve related information on the Web. However, conventional document ranking based on an automatic assessment of document relevance to the query may not be the best approach when little information is given as in most cases. In order to clarify the ambiguity of the short queries given by users, we propose concept-based relevance feedback *for Web information retrieval. This idea is to help users formulate their queries by having users give two to three times more feedback for traditional query methods. We apply* clustering techniques *to initial search results to provide concept-based browsing. We will show how clustering improves performance over conventional similarity ranking, and most importantly, the assistance of cluster-based representation can reduce the browsing labor for short queries.*

Keywords: data presentation, document clustering, relevance feedback, concept-based feedback

1 Introduction

The World Wide Web is one of great richness and diversity. It contains a complete universe of online information. However, it has become almost impossible to look for specific information without getting lost among large amounts of mixed data. Given any query to the search engines on the web, you probably get hundreds or thousands of "hits" in return. Thus, online information searching often turns out to be a process of endless browsing. Besides, there is no way for the user to know what has been missed and how to adjust their queries.

The problems arise from two perspectives: one is the difficulty in query formulation and the other is the inherent word ambiguity in natural language. More specifically, most users find it is easier to answer yes/no questions rather than describe what they really want. This situation is best illustrated through the scenario of information search on the Web, where queries are usually of two words long [2] (which are defined here as *short queries*). In such cases, ambiguity occurs because a large number of documents are considered to "match" the query by conventional search techniques. For example, the query for "agent technology" will result in response including Usenet news reader — Agent, real estate agents and intelligent agent technology etc. Though different ranking algorithms have been used to rank the documents according to their relevance to the query, the effects is subjected to controversy since the query says too few about what the user really wants.

To remedy such shortcomings in keyword-based information searching environment, the feedback from the user is necessary to clarify the ambiguity and to help formulate appropriate query expression. However, feedback can be a tiring process. Thus, a better interaction between users and the system should be devised. Of current information systems, there are two ways to let the user give feedback. One is to have the user feedback related terms suggested by the system, such as the LiveTopics provided by AltaVistaand the one provided by Excite.The other alternative is through the feedback of documents such that enforcement learning can be applied. For example, the "More Like This" function provided by Lycosand the "Search for more documents like this one" function provided Excite etc.

In the first approach, the user is asked to modify the query specifically by adding new words suggested by the system; while in the second approach, the query is modified by the feedback of a relevant document. Through the feedback information, the query is expanded to better formulate what the user wants. Nevertheless, neither of the two mechanisms is perfect enough. Individual terms usually transfer too little information, while browsing documents may cost a lot of time. In fact, the search results returned by similarity ranking from indexers often present clustering characteristic on several topics. Part of the reason is the free publication and multiplication of the Web such that replicated pages are repeated in the search results. Thus, grouping these initial search results can speed up the browsing process and make it easier for feedback.

In this paper, we propose concept-based relevance feedback as well as a joined mechanism for query expansion. This is continuing research from our previous work based on document clustering [1]. The idea is to organize the initial documents retrieved by the original query into conceptual groups such that the user can get a quick overview of what the query retrieves in the minimal amount of time possible. Under this designing philosophy, document clustering becomes the main step toward concept-based browsing. The feasibility of this approach is based on the *cluster hypothesis* which states that similar documents are more related to the same topic than documents that are less similar to each other.

In the following, we will first review some of the past research on query expansion and relevance feedback. Next, we will describe the main techniques employed to implement a concept-based information retrieval, including document clustering, keywords extraction, and query expansion. Evaluation for these techniques are presented and preliminary result is shown to see the effects.

2 Past Research in Query Expansion

Query expansion has long been suggested as a technique for dealing with the fundamental issue of word mismatch in information retrieval. The problem of word mismatch occurs when different words are used to describe a same concept. Thus, even the best of the information retrieval systems have a limited recall. Often users may retrieve a few relevant documents in response to their queries, but almost never all the relevant documents [10]. From another point of view, query expansion may also be a good approach to relieving the burden in query formulation. By introducing new words into the original query, the effect of ambiguity in words can be reduced. Therefore, query expansion can be used to solve low precision in current information systems.

There are three main approaches for query expansion. One is manually-constructed thesaurus such as WordNet used in [9]. Another is automatically-constructed thesaurus from the corpus to be searched. The other is the query-specific approach that suggests related terms from initial search results retrieved by similarity ranking.

In the first approach, term suggestion is supported by manual thesauri, say Roget's thesaurus or WordNet, which provide for every term a list of broader, narrower and related terms. However, they are difficult to use because of multiple meanings for many words. Thus, only small improvements are possible with longer queries which provide clues for which word senses are involved, but expanding shorter queries actually degraded performance [9].

The second approach is to analyze the corpus being searched and discover word relationships based on their co-occurrence patterns at a document level. One of the earliest studies of this type was carried out by Sparck Jones who clustered words based on co-occurrence in documents to expand the query [4]. This kind of approach, called *global corpus analysis*, is computationally intensive but the computations are done only once per corpus and this information is used to expand any particular query thereafter [10].

The last resort which is also a suitable technique for personal web assistant is the *query-specific analysis* which involves only the top ranked documents retrieved by the original query. In recent work by Xu and Croft, queries are modified automatically by the assumption that the top ranked documents are relevant (implicit feedback). They also apply techniques borrowed from global corpus analysis to the initial search results and get even better results [10]. Also in Harman's work, the query is expanded or modified based on the (explicit) feedback of documents from the user. The number of terms added to the query is experimented to achieve best precision improvement when 20–40 terms are added to the query [3].

From another perspective, two models have been adopted in query expansion for relevance feedback. One is the vector space model initiated by Rocchio in 1971 [8]. The other is the probabilistic model proposed by Robertson and Sparck Jones in 1976 [6]. The basic module in Rocchio's algorithm is the merging of document vectors with the original query vectors such that queries are automatically expanded by adding all the terms not in the original query but in the initial documents (relevant and irrelevant). While the Robertson and Sparck Jones's model is based on the distribution of "query terms" in relevant and irrelevant documents (Various ways of term weighting in probabilistic model are compared in [7] by Robertson and Walker). Since the probabilistic model did not envision query expansion but only the reweighting of terms based on relevance judgments, several reasonable sets of sorting are tried to select non-query terms for query expansion [3].

3 Concept-based Information Search

The difficulty to formulate a request and the inherent word ambiguity in natural language can be overcome by concept-based relevance feedback. To demonstrate the idea, let us first give the search result of the query "TREC conference proceeding" as an example (Figure 1). In this search result, two clusters are displayed. The first is the Text Retrieval Conference publications from NIST (National Institute of Standards & Technology) and the second is the pages from Texas Real Estate Commissions. Each of the clusters is represented with a set of words after the topic number and a sample document describing the cluster. The group size is also given to indicate the number of retrievals in the cluster. Note that the sample document has played an important role in describing a cluster since it helps the user to decide whether or not each cluster is relevant to the query. To see the detailed contents of each cluster, one can click on the topic number of each cluster as shown by Figure 2. To give feedback for each cluster, the user can click on the smiling face in front of each cluster to clear (include) all of the articles under that cluster, or the frowning face to select (exclude) all of the choices under that cluster. The user can also select single articles under each cluster if the remaining articles are not relevant to the query. In the next section, we will describe the implementation of this concept-based feedback in detail.

To give an overview, the operations of the Web Search Assistant proceeds as follows. When a query is first commenced, it is forwarded to a multi-thread search engine. The Web assistant then collects the top n documents for further processing. We call this the preprocessing phase for data gathering. Next, document clustering is applied to organize similar documents into groups. Then, each cluster is digested and named by a set of words and a sample document.

3.1 Document clustering

Cluster analysis has long been applied in information systems to improve the efficiency and effectiveness of retrieval for quite a time [5]. The basic principle behind clustering is to select objects that have high similarities into the same cluster. Various clustering algorithms differ in their use of the similarity measures between groups. For document clustering, the measure of association between documents is determined by the inner product of the two document vectors, $v(d_i)$, and $v(d_j)$:

$$S_{d_i,d_j} = v(d_i) \cdot v(d_j) = \sum_{t \in d_i \cap d_j} w_{i,t} w_{j,t} \qquad (1)$$

where $w_{i,t}$ is the weight of term t_k in document d_i and is computed by the augmented normalized term frequency by inverse document frequency (TFxIDF):

$$w_{i,t} = \frac{(0.5 + 0.5\frac{tf_{i,t}}{tfmax_i}) \log \frac{N}{df_t}}{\sqrt{\sum_{k \in d_i} (0.5 + 0.5\frac{tf_{i,k}}{tfmax_i}) \log \frac{N}{df_k}}} \qquad (2)$$

with

$tf_{i,t}$	=	the frequency of term t in document d_i,
$tfmax_i$	=	the largest $tf_{i,t}$ in document d_i,
df_k	=	the document frequency of term t in our collection G, where documents are sampled from the Web, and
N	=	the number of documents in G.

The clustering algorithm we adopt here is a variation of the hierarchical agglomerative clustering methods (HACM) [5]. The general algorithm for the HACM is derived by identifying the two closest clusters and combining them in one. The commonly used HACM include single link, complete link, and group average link, which differ in their measures of the intercluster similarity. To get an intermediate structure of the clustering results, we use the *group average values* of the pairwise links within the cluster as the similarity measure. Formally, two groups C_1 and C_2 that maximize S_{C_1,C_2} are selected for coalesce.

$$S(C_1, C_2) = \frac{\sum_{d_i \in C_1 \cup C_2} \sum_{d_j \neq d_i} S_{d_i,d_j}}{|C_1 \cup C_2|(|C_1 \cup C_2| - 1)} \qquad (3)$$

3.2 Cluster digesting and naming

The second step of the postprocessing is cluster digesting and naming. By cluster digesting, we mean the selection of representative words from a cluster. Naming means the assignment of a representative document to that cluster. To select representative words for each cluster, we propose a two-phrase selection algorithm which combines the majority principle and the probabilistic term weighting. During the first phrase, words that have *sufficient* postings in the initial document set are selected. Since approximately 75% of the words in the initial query results have less than four postings, only 25% of words are considered in the second phrase. At the second phrase, the selected words are ranked by their *normalized cue validity*, which are computed by multiplying the *cue validity* to the inverse document frequency. The normalized cue validity of a term t with respect to a cluster C is formulated as follows:

$$ncv_{t,C} = \frac{P(t|C)}{P(t|C) + P(t|\bar{C}) + \epsilon} \cdot \log \frac{1}{P(t|G)} \qquad (4)$$

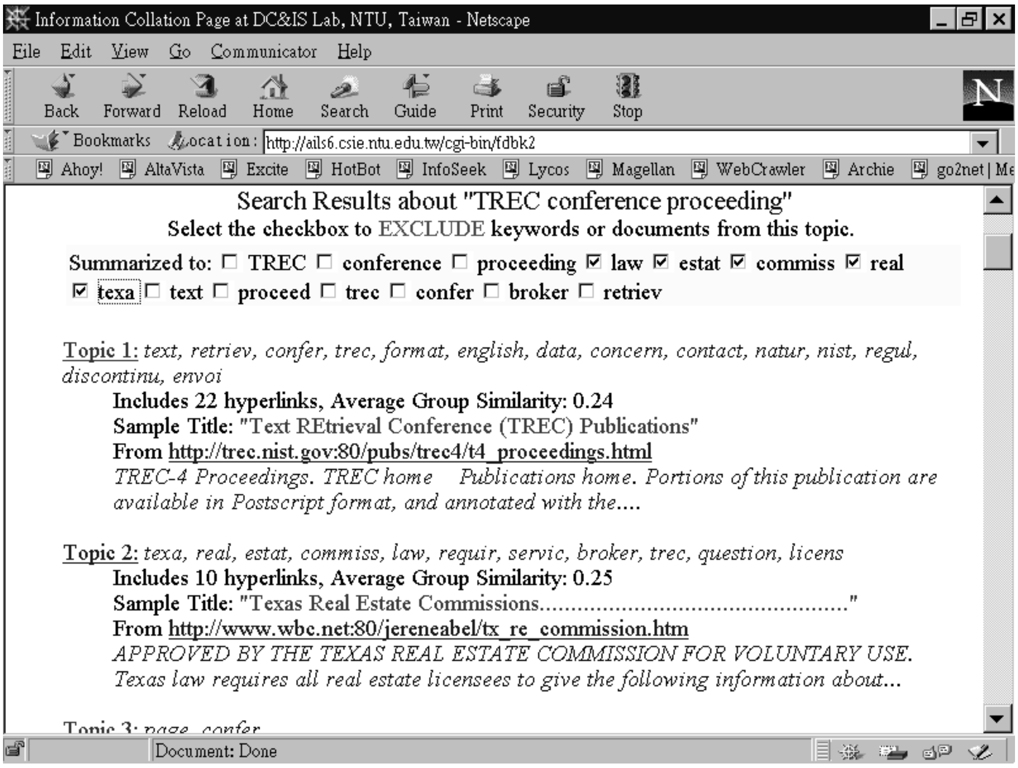

Figure 1: **The first two clusters for query "TREC conference proceeding"**

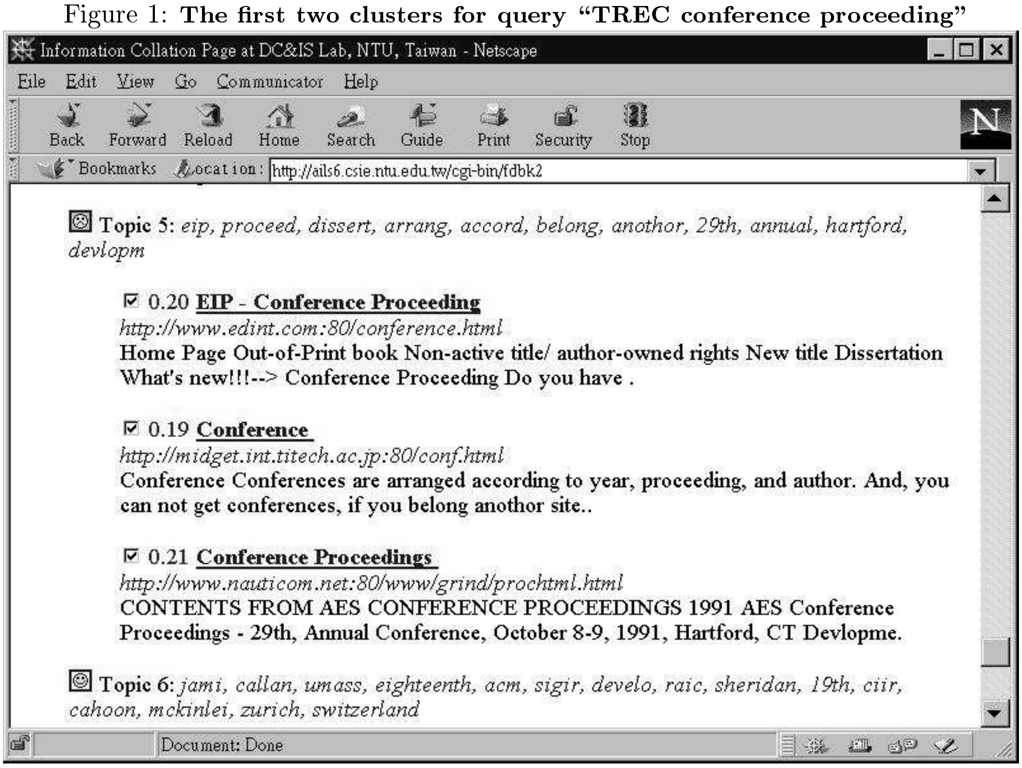

Figure 2: **Feedback signs: frowning face or smiling face.**

where $P(t|C)$ represents the relative posting of term t in the cluster (document set) C. The complement set, $\bar{C}$, includes those documents that are not in cluster C from the initial query result. The inverse document frequency is given by the log $\frac{1}{P(t|G)}$, where the global corpus G (as discussed in Eq. 2) represents the document set that the Web Search Assistant has collected. The global corpus contains 15309 documents and continues growing as more documents get in the database.

The idea of cue validity, a similar measure to the probability weighting, is to highlight those words that have a distinguished occurring frequency in a cluster with respect to its complement. A small value ϵ is included in the denominator of the cue validity to avoid the case where $P(t|\bar{C})$ is zero such that the cue validity $\frac{P(t|C)}{P(t|C)+P(t|\bar{C})+\epsilon}$ equals 1. Finally, the top 40–50 terms with the highest $ncv_{t,C}$ constitute the summary vector for the cluster.

As for the selection of a sample document, two parameters are considered. The first is the proximity of a document to the cluster centroid. The second is the URL (Universal Resource Locator) depth, defined as the number of slashes "/" in a URL, of a document. For those documents that are closest to the cluster centroid, the one with the least URL depth is chosen as the sample document.

3.3 Query modification mechanisms

In this section, we discuss query modification with relevance feedback. In this feedback scenario, the user first decides which groups are most relevant to the query, then uses the smiling face, frowning face, and checkboxes to indicate cluster or document relevance for further refinement (See Figure 2). Two approaches are employed for query modification.

In the first approach, the query is modified by the cluster digests constructed above. Such an approach is intuitive since less relevant words may have been eliminated at the digesting step. To modify a query at iteration t, the summary vectors of relevant clusters and irrelevant clusters are merged with the query vector Q_t as follows:

$$Q_{t+1} = Q_t + \sum_{C_i \in \Phi_t} \beta \cdot V(C_i) - \sum_{C_j \in \Psi_t} \gamma \cdot V(C_j) \quad (5)$$

where
- Φ_t and Ψ_t are the relevant and irrelevant cluster sets,
- $V(C_i)$ is the summary vectors for cluster C_i, and
- β and γ are specified as the numbers of documents per cluster and control the relative contributions of and relevant and irrelevant clusters.

In the second approach, the cluster boundaries are eliminated such that the search results are divided into relevant and irrelevant sets according to the relevance of individual documents. Given R_t and S_t as relevant and irrelevant document sets respectively, the document-based modification is formulated as follows. The idea is to exploit the modification method in document-based feedback and employs it under the idea of cluster-based feedback with individual document refinement.

$$Q_{t+1} = Q_t + \beta \cdot V(R_t) - \gamma \cdot V(S_t) \quad (6)$$

3.4 Implementation and Interface

The Web search assistant described in this paper consists of a standalone application server that interacts with browsing clients through the Common Gateway Interface (CGI). There is no actual database containing Web pages. The similarity ranking results are returned by a multi-thread search engine which forwards queries to six online search engines (including AltaVista, Excite, InfoSeek,Lycos, Magellanand WebCrawler[1]. The server collates the search results and performs several procedures as discussed above to present the output. During later interactions, the user provides feedback to modify search results and the system adapts to the new search criteria.

Continuing with the search results of the "TREC conference proceeding" query, the "Summarized to" section in Figure 1 contains words that are extracted via implicit cluster digesting, where n documents in the initial search results are taken as a relevant set and the global corpus G as the complement set. This is done under the assumption that almost all documents in an information database are likely to be irrelevant to the specific query, thus the global collection G can be viewed as a complement set to any queries. The summarized vector is then merged with the original query to give weights to documents. The user can *exclude* unsolicited words by clicking the checkboxes. In this example, the query is summarized to "*real, estat, law, commiss, texa, text, proceed, trec, confer, broker, retriev,*" where the words "*texa, real, estat, commiss, law, broker*" are clicked to be excluded in the refine step. The decision of which words to select can be implied from the clustering results, since the words "*texa, real, estat, commiss, law, broker*" are part of the cluster digest for Texas Real Estate Commissions.

For the interface design of the cluster-based feedback, we adopt a document-feedback-like approach which facilitates both cluster feedback and document feedback. The smiling and frowning faces in front of each cluster are feedback signs that indicate the rel-

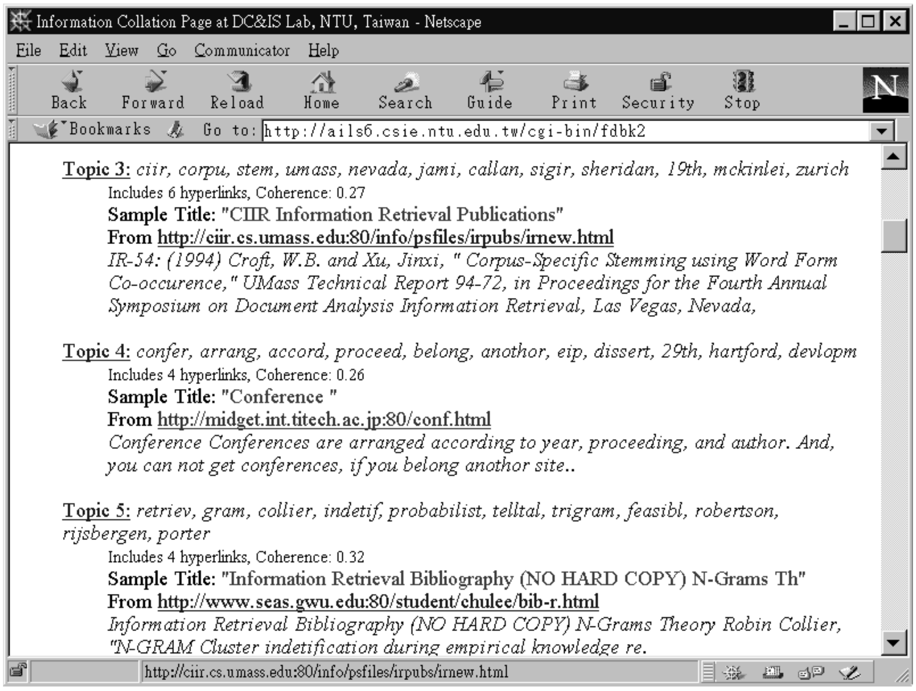

Figure 3: The result of query "TREC conference proceedings" after refinement.

evance of each cluster. For example, clicking on the smiling face will turn the feedback sign into a frowning face and cause the checkboxes of all documents in that cluster to be selected for exclusion, and vice versa. Users can also indicate relevance of individual document by clicking the checkbox in front of the document. Finally, the refine button at the bottom of the result page can be clicked to give feedback such that query modification approaches are applied. For example, refinement of the "TREC conference proceeding" query is shown in Figure 3, where new relevant clusters such as "CIIR Information Retrieval Publications" and "Information Retrieval Bibliography (NO HARD COPY) N-Grams" as well as irrelevant clusters such as "Conference" are generated after feedback.

4 Evaluation of System Performance

In this section, we evaluate the overall system performance. To evaluate performance, the traditional evaluation methods *precision* (the fraction of visited documents that are relevant) and *recall* (the fraction of relevant documents that have been visited) are employed. However, for an online Web searching task, the Web generally lacks the relevant sets that are necessary to measure *recall*. Hence, the *relative recall* which is compared to a retrieved set is used instead. Thirteen trials are used in the experiments. After the multi-thread search engine provides the results, the relevance of each document according to the search is determined. The relevances of these search results are then recorded to compute the precision of each initial query. The search results (with 10 hyperlinks retrieved at each transaction from one search engine) are combined into an order dependent on each search engine's response time. Assuming a fairly equivalent response time from each search engine, this document ordering is equivalent to the ordering of some similarity ranking. For various document cutoffs (i.e. 30, 60, 90, 120, 150), the precision values are then recorded as baselines for later comparisons, where query modification techniques are applied.

To see the clustering results, the percentage of documents relevant in each cluster is depicted in Figure 4. To obtain this result, the clusters in each group are first divided into three groups according to the number of documents in a cluster. The clusters are then sorted according to the percentage of documents relevant to the query (in descending order). Theoretically speaking, for clusters that contain only one document, a relevant (1) or irrelevant (0) score is assigned. Thus, the ideal shape of the curve should look like that of a step function. However, for non-singleton clusters, the curve should look like that of a sigmoid function. That is, the percentage of relevant documents in a cluster is either as high as 1 or as low as 0, such that a cluster

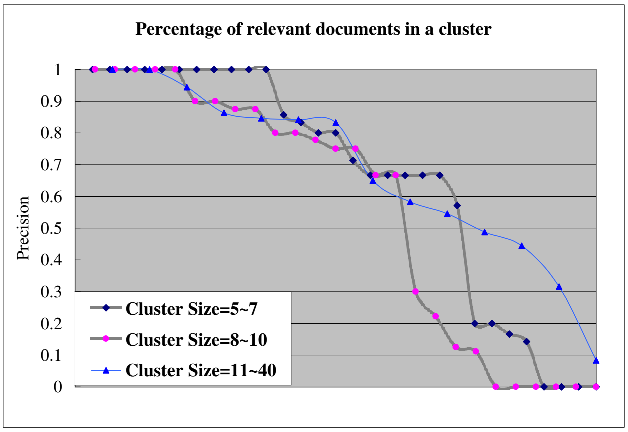

Figure 4: **The percentage of relevant documents found for each cluster ranked by decreasing precision.**

is either relevant or irrelevant.

As plotted in Figure 4, the curve of cluster size 5 to 7 has a non-linear decrease which separates the clusters into two sets. The clusters on the left have at least 65% relevance and the clusters on the right always have less than 20% relevance. Note that as cluster size grows, the sharp contrast blurs accordingly, which explains the difficulty of sharply defining a topic of interest for large clusters.

Having an approximate idea of the clustering results, we will now evaluate how clustering can affect system performance. Table 1 shows the precision of documents in all clusters of a query result (column Cluster-A) and also the precision of documents in relevant clusters of a query result (column Cluster-R) at various cutoffs. Specifically, at a cutoff value of 90, assume that the clustering result produces 7 main groups and 30 singleton clusters. The precision for documents in all clusters (Cluster-A) are measured from the 60 documents in the 7 main groups. This value is then compared to the precision of similarity ranking at a cutoff value of 90 (column Sim-Ranked), where a 12.5% improvement is gained as indicated in Table 1. From another viewpoint, if we consider clustering as a filtering technique that filters out singleton clusters, the 60 documents in the 7 main clusters also contain more relevant documents than are obtained with similarity ranking with a cutoff value of 60.

In fact, the improvement is even better when we remove irrelevant clusters and measure the precision of documents in relevant clusters (column Cluster-R). By relevant clusters, we do not mean those with high precision, but rather those with their sample documents identified relevant by the user. In most cases (92%), a cluster often has at least 60% precision if the sample document is identified as relevant. In this measure, Cluster-R has a precision improvement as high as 45.9% at a cutoff value of 90 (see Table 1). Of course, such improvement in precision has resulted in some degradation in *relative recall*, i.e. the fraction of relevant documents retrieved from the original documents using similarity ranking. As shown in Table 2, the penalty for a 29.7% increase in precision is an 11.1% decrease in relative recall at a cutoff value of 90. Nonetheless, the most exciting result clustering has brought us is the large decrease in effort the user spends in browsing. Instead of browsing through tens of documents, the user needs only to review the sample documents in each cluster, which are far less than the number of individual documents.

5 Summary

In this paper, we propose document clustering and query expansion as the main technology in concept-based relevance feedback. The idea is inspired from the observation that most users give very little information as query input. Thus, we try to apply the clustering technique to summarize related topics from similarity-ranking search results and explore new techniques for query expansion via keyword extraction and query modification.

To some extent, the idea of concept-based information retrieval is to integrate the query-oriented search model with the browsing-oriented search model by way of topic/subject categorization such that the choices

n	Precision Increase For Clustering				
	Sim-Ranked	Cluster-A	Increase	Cluster-R	Increase
30	0.643	0.711	10.6%	0.828	28.8%
60	0.606	0.677	11.5%	0.845	39.4%
90	0.550	0.620	12.5%	0.803	45.9%
120	0.528	0.664	25.5%	0.774	46.5%

Table 1: **Comparison of precision for similarity ranking and clustering. Cluster-A refers to percentage of relevant documents in all clusters of a query result. Cluster-R refers to percentage of relevant documents in relevant clusters. The increase refers to comparison with similarity ranking.**

n	Relative Recall Degradation vs. Precision Increase For Clustering					
	Precision-A	Precision-R	Increase	Recall-A	Recall-R	Decrease
30	0.712	0.829	16.4%	0.678	0.611	-9.9%
60	0.675	0.845	20.2%	0.744	0.596	-19.9%
90	0.620	0.803	29.7%	0.724	0.643	-11.1%
120	0.663	0.774	16.7%	0.730	0.678	-7.1%

Table 2: **Comparison of Cluster-A and Cluster-R. For relative recall at cutoff n, Recall-A refers to the fraction of relevant documents in all main clusters from n documents, whereas Recall-R refers to the fraction of relevant documents in relevant clusters.**

are confined to a limited number of topics and relevance is easily decided when giving feedback. The obvious advantage of concept-based information retrieval is that it accelerates the browsing speed by dividing the initial results into several document groups such that relevance feedback can be given by a dichotomy of relevant and irrelevant clusters.

The designing principle is especially useful for short queries that encompass a wide range of topics because of word ambiguity in natural language. Thus, document clustering can serve as the basic mechanism for concept-based information retrieval. However, for long queries that focus on some special topics, other techniques should be used since categorization does not simply depend on similarity measures but rather depends on arbitrary categorization methods.

References

[1] C.H. Chang and C.C. Hsu. Customizable multi-engine search tool with clustering. *Computer Network and ISDN Systems*, 29(8-13):1217–1224, Aug 1997.

[2] W.B. Croft, R. Cook, and D. Wilder. Providing government information on the internet: Experiences with thomas. In *Proc. of Digital Libraries Conference*, pages 19–24, 1995.

[3] D. Harman. Relevance feedback revisited. In *Proc. of ACM SIGIR Intl. Conf. on Research and Development in Information Retrieval*, pages 1–10, 1992.

[4] K. Sparck Jones. *Automatic Keyword Classification for Information Retrieval.* Butterworth, London, 1971.

[5] E. Rasmussen. Clustering algorithms. In W. Frakes and Baeza-Yates R., editors, *Information Retrieval: Data Structures and Algorithms*, chapter 16. Prentice-Hall, 1992.

[6] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[7] S.E. Robertson and S. Walker. On relevance weights with little relevance information. In *Proc. of ACM SIGIR Intl. Conf. on Research and Development in Information Retrieval*, pages 16–23, 1997.

[8] J.J. Rocchio. Relevance feedback in information retrieval. *The SMART Retrieval System*, pages 313–323, 1971.

[9] E. Voorhees. Query expansion using lexical-semantic relations. In *Proc. of ACM SIGIR Intl. Conf. on Research and Development in Information Retrieval*, pages 61–69, 1994.

[10] J. Xu and W.B. Croft. Query expansion using local and global document analysis. In *Proc. of ACM SIGIR Intl. Conf. on Research and Development in Information Retrieval*, pages 4–11, 1996.