Enabling Concept-Based Relevance Feedback for Information Retrieval on the WWW

Chia-Hui Chang, *Student Member*, *IEEE*, and Ching-Chi Hsu

Abstract—The World Wide Web is a world of great richness, but finding information on the Web is also a great challenge. Keyword-based querying has been an immediate and efficient way to specify and retrieve related information that the user inquires. 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 the idea of *concept-based relevance feedback* for Web information retrieval. The idea is to have users give two to three times more feedback in the same amount of time that would be required to give feedback for conventional feedback mechanisms. Under this design principle, we apply clustering techniques to the initial search results to provide concept-based browsing. We show the performances of various feedback interface designs and compare their pros and cons. We shall measure *precision* and *relative recall* to show how clustering improves performance over conventional similarity ranking and, most importantly, we shall show how the assistance of concept-based presentation reduces browsing labor.

Index Terms—Query expansion, relevance feedback, concept-based feedback, keyword extraction, document clustering, document-based browsing, cluster-based browsing.

1 Introduction

The World Wide Web is a world of great richness and diversity. It contains a complete universe of on-line information. However, with its continuing growth, finding information on the Web has become a difficult and time-consuming task because of the Web's tremendous size and various kinds of information. Traditionally, the contents that an information system provide are limited to a certain subject or domain, for example, medicine, law, such that index terms can be selected by knowledgeable persons consulting auxiliary terminology lists describing allowable vocabulary entries. In such cases, a considerable degree of indexing uniformity can be achieved, and high-quality retrieval is sometimes obtained. However, the Web is a completely open environment where everyone can publish their articles which results in synonyms and abbreviations creating more word sense ambiguity for retrieval systems.

The main issue of keyword-based query model lies in the difficulty of query formulation and the inherent word sense ambiguity in natural language. Most users find it is easier to answer (moderate) yes/no questions rather than describe what they really want. The situation is best illustrated through an information search scenario on the Web, where the queries are usually two words long [4] (which are defined here as *short queries*). In such cases, ambiguity occurs because a large number of documents may be considered to "match" the query by conventional retrieval techniques. For example, a query for "agent technology" will result in a response including Usenet news reader—Agent, real estate agents, intelligent agent technologies, etc. This is because there are various interpretations for a given query and there is too little information to clarify the ambiguity by the short query.

To overcome this problem, researchers have focused on automatic query expansion to help users formulate their queries. For example, research topics have focused on relevance feedback mechanisms which allow the user to give feedback by choosing from a list of words or documents to clarify the ambiguity in a query. However, the mechanisms of relevance feedback based on words or documents in previous research have deficiencies. As reported in [15] by Voorhees, relevant word feedback has its upper bound performance in lexical-semantic expansion and, while performance is increased as more relevant documents are given, it is sometimes too troublesome for the users.

In this paper, we propose concept-based feedback together with a joined mechanism for query expansion. This is continuing research from our previous work based on document clustering [2]. The idea is to organize 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 design philosophy, document clustering becomes our first step in concept-based feedback. The feasibility of document clustering 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.

Indeed, providing a concept-based search interface as well as an interactive feedback mechanism has attracted many researchers in the past two years. For example, the dynamic browsing paradigm from Scatter/Gather [5] that

• *C.-H. Chang and C.-C. Hsu are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan 106. E-mail: chia@ails1.csie.ntu.edu.tw, cchsu@csie.ntu.edu.tw.*

Manuscript received 10 May 1998; revised 22 Sept. 1998.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 108909.

clusters corpus documents into topical-coherent groups has been recently applied in a conventional similarity search to navigate the retrieved documents by Hearst and Pedersen [8]. Likewise, static clustering of database contents has also been exploited by Anick and Vaithyanathan [1]. The authors discuss the cognitive load required to assess cluster contents by way of their key terms. They also introduce natural language processing techniques to extract noun phrases for describing cluster contents.

The exploitation of concept-based feedback in addition to document clustering distinguishes our research from previous work in this field. Indeed, the target of our research is to implement a concept-based feedback interface in a personalized Web information search assistant by integrating existing search engines with techniques of query expansion and relevance feedback. We shall focus on the mechanisms of keyword extraction for both cluster digesting and query expansion, and compare the performances of different feedback mechanisms. Meanwhile, we shall show how such a Web assistant can be constructed and the effectiveness of this design over conventional Web search tools based on similarity ranking.

In the next section, we will present the idea of concept-based relevance feedback and compare it to past research. Section 3 details the implementation of a real IR search system on the Web. Section 4 presents the experimental results and discusses the pros and cons of different approaches for feedback. We will then discuss the benefits of the implementation and conclude the paper in Section 5.

2 Concept-Based Relevance Feedback

In this section, we will discuss some of the past research on information retrieval including the comparison of query expansion with relevance feedback and various applications of document clustering. These are the main techniques concerning concept-based relevance feedback. We will then introduce the idea of concept-based relevance feedback to solve the problem of ambiguity associated with short queries on the Web. Meanwhile, we will give an illustration of our search system to show how it is implemented.

2.1 Query Expansion vs. Relevance Feedback

As we mentioned earlier, 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 the 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 rarely are all the relevant documents retrieved [17]. From another point of view, query expansion may also be a good approach to relieving the burden the user experiments in query formulation. By introducing new words into the original query, the effect of ambiguity in words can be reduced. Therefore, query expansion can minimize low precision in current information retrieval methods.

But how can we obtain the terms for expansion? Is relevance feedback available during query expansion? Terms can be suggested by using a language specific thesaurus, say Roget's *Thesaurus* or WordNet, or by analyzing the co-occurrence patterns of terms in a particular corpus, or by locating new terms in a subset of documents retrieved by a specific query. The last approach (also called *query specific analysis*), has been proven to be quite effective when relevance feedback is available [7].

For many years, researchers have suggested relevance feedback as a mechanism for query modification. In the early 1960s, Rocchio described an elegant approach by showing how the optimal vector space query can be derived using *vector* addition and subtraction given the feedback of relevant and irrelevant documents [13]. In 1976, Robertson and Sparck Jones proposed the *probabilistic model* and showed how to adjust the individual term weight based on the distribution of these terms in relevant and irrelevant documents retrieved in response to queries [12].

In the *vector space model*, each document and query can be envisioned as an n-dimensional vector space, where n corresponds to the number of unique terms in the data set. A vector merging operation based on addition and subtraction can then be used to expand queries by automatically adding all the terms that are in the retrieved documents and then weighting terms according to document relevances. However, no new terms are actually added with negative weights; the contribution of irrelevant document terms is to modify the weighting of new terms coming from relevant documents. While in the original *probabilistic model*, the weighting scheme did not expand the query by adding new terms but only adjusted term weights. Several attempts have been made to handle this extension [7].

From the perspective of relevance feedback, both of the models are mainly based on the *document unit*, which requires the user to browse through the documents and decide which are relevant. In fact, for language specific query expansion from a manually constructed thesaurus, only small improvements are possible with longer queries. It is because many words have multiple meanings such that ambiguous results are often obtained when employing automatic query expansion. Thus, feedback from the user introduced in the form of *word suggestion* is necessary to achieve better performance. However, there is an upper bound even when words are selected by the user [15].

From this brief survey, we know that query specific analysis is the most effective way for query expansion. With relevance feedback from the user, performance is improved even more. However, the browsing process required to decide relevance is sometimes time-consuming if a large amount of feedback is required to improve the performance. Hence, the problem at hand is creating a system that can help users increase efficiency in relevance feedback.

2.2 Applications of Document Clustering

Document clustering has been extensively investigated in information retrieval as a methodology for enhancing conventional document search and retrieval (see [11] for more applications of clustering in information retrieval). The underlying reason for this enhancement is referred to

as the *cluster hypothesis*: closely associated documents tend to be relevant to the same requests. Hence, automatic clustering of related documents can improve recall by effectively matching queries against clusters and retrieving clusters with the highest matching score.

Another application, as mentioned before, is the dynamic browsing paradigm (dynamic category generation) which is used to browse large document collections [5]. In this Scatter/Gather paradigm, a corpus of data is dynamically clustered into similar groups and presented in a table of content format. This method stands out as a contrast to static directory browsing, and can be used as a good approach for data presentation.

Consider a situation in which a large amount of information is retrieved using keyword matching, but the retrieval precision is low due to the inappropriate match of keywords. Such conditions do occur because of the manifold properties of many words (e.g., "Java" as a computer language, a kind of coffee, and even a location name). A similar situation occurs when the given words are too general (e.g., words such as "computer," "information," etc.) which results in a search space that is too extensive. The same problem arises when users access other databases such as traditional library systems.

If we examine the search results more closely, we will find quite a resemblance between documents, which leads one to expect that clustering documents into conceptual subgroups can help separate the items of interest from those that matched the query for spurious or unanticipated reasons. That is, document clustering may be applied to exploit its hidden significance in the keyword-based query model. This intuitive hypothesis has received some recent experimental validation, such as in the Scatter/Gather extension and Paraphrase [8], [2], [1]. The underlying principle of document clustering is mainly for data presentation. For example, the Scatter/Gather system was redesigned by Hearst and Pederson to organize the results of a query into a small number of article groups [8]. From their experimental results, they show that users are able to interpret the cluster summary information well enough to select the cluster with the largest number of relevant documents. A similar application has also been carried out on the Web under a multiengine search architecture by Chang and Hsu [2]. Meanwhile, *Paraphrase*, by Anick and Vaithyanathan, has shown the synergy of static clustering (presearch clustering of a database) together with phrase information playing the role of context descriptors [1].

To summarize, the work by Cutting et al. [5] and Anick and Vaithyanathan [1] can be classified as static global corpus analysis; while the work by Hearst and Pedersen [8] and Chang and Hsu [2] can be classified as dynamic local analysis. As shown in Fig. 1, typical information systems compare a detection need against documents in the corpus, whereas these systems employ clustering techniques in the preprocessing of documents or after comparison of the detection need and document corpus. From another perspective, the concepts discussed in [8], [2], [1] are all oriented from the idea of giving a better presentation of search results. Thus, they can be classified as a kind of query specific approach.

2.3 Concept-Based Feedback and the Illustration

Note that in the Scatter/Gather system, cluster selection filters out documents that are not relevant, thus reducing the number of documents to be clustered without modifying the original query in later interactions [8]. While in *Paraphrase*, modification of a query occurs through explicit feedback by clicking on the phrases in each cluster. In this paper, we explore the effects of query expansion in the context of *concept-based feedback* and show an improvement in relevant document retrieval.

In other words, what we are addressing is not only filtering of documents by user feedback of interested clusters, but also expanding the original query to formulate a larger, more comprehensive retrieval that is more specific to the user's query. As opposed to individual document browsing, cluster-based feedback can draw more information from the user while spending the same amount of time required in browsing an average amount of information, say 10 or 20 documents. Thus, the feedback information can be further applied in query modification to achieve better performance.

In terms of current Web information search, such a design can be especially helpful. This is because the Web contains such a vast amount of information that many abbreviations have multiple meanings. To name a few, "UPS" stands for Uninterruptible Power Supply or United Parcel Service, whereas"ATM" stands for Automatic Teller Machine and Asynchronous Transfer Mode, etc. More unanticipated examples can be found. For example, the abbreviation "TREC" which is commonly known as Text Retrieval Conference in the area of information retrieval is also an abbreviation for Texas Real Estate Commissions,[1] the Tenderloin Reflection and Education Center,[2] and Technical Resources Environmental Conservation.[3] This motivates us to construct our concept-based Web search tool to assist users in searching the Web.

To demonstrate concept-based feedback, let us first give a result of our search tool by using "TREC conference proceeding" as an example. A portion of the search result is shown in Fig. 2 with two clusters displayed. The first is a Text Retrieval Conference publication from the National Institute of Standards and Technology (NIST), and the second is Texas Real Estate Commissions as indicated by the sample titles of the two clusters. 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 decide whether or not each cluster is relevant to the query. To see the detailed contents of each cluster, one can click on the

1. The Web site for the Texas Real Estate Commissions is http://www.trec.state.tx.us.

2. The Tenderloin Reflection and Education Center's Web site is http://www.jps.net/voices/trecidex.htm.

3. The Technical Resources Environmental Conservation's Web site is http://www.spidercop.com/.

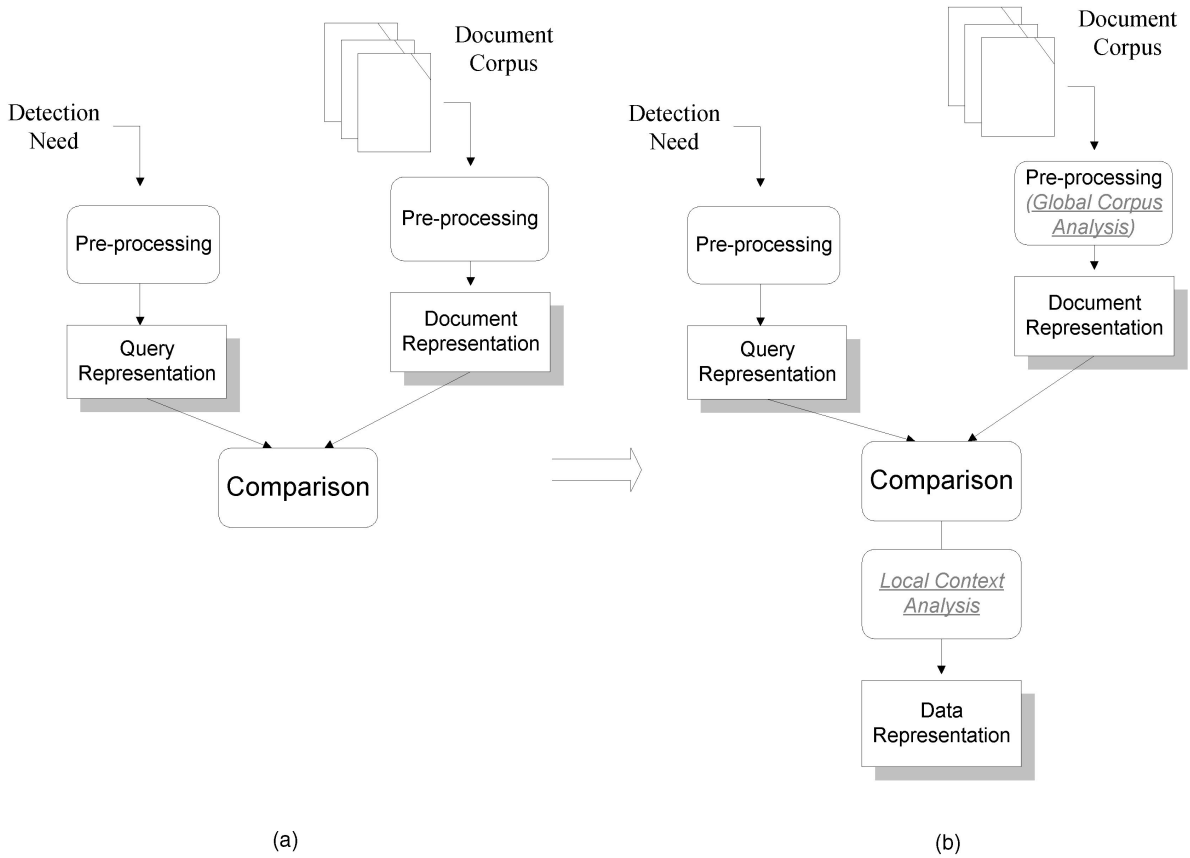

Fig. 1. (a) Typical architecture of an information retrieval system; (b) *global corpus analysis* vs. *local context analysis*.

topic number of each cluster as shown by Fig. 3. 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. Such a design is based on the consideration to offset the improper classification of documents for large clusters and will be discussed in Section 4. In the next section, we will describe the implementation details of this concept-based feedback design.

3 The Concept-Based Web Search Assistant

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 of queries are returned by a multithread search engine which forwards queries to a couple of online search engines (including AltaVista, Excite, InfoSeek, Lycos, Magellan, and WebCrawler). Multithread search has the advantages of providing a consistent user interface and increasing the limited availability and coverage of other search engines, thus has been adapted for many applications [14], [2]. The server then collates the search results and performs several procedures as discussed below to present the output. During later interactions, the user provides feedback to modify search results and the system adapts to the new search criteria. The architecture is shown in Fig. 4.

To give an overview of how the Web search assistant works, let us first examine the main procedures of the system and see how a query is processed. When a query is first commenced, it is forwarded to the multithread search engine. The Web assistant then collects the top n documents for further processing. This is the preprocessing phase for data gathering. The postprocessing phase is divided into four parts: document clustering, cluster digesting and naming, query expansion with concept feedback, and implicit query expansion. First, the Web search assistant performs document clustering to organize similar documents into groups. Second, the search assistant applies cluster digesting to extract keywords and selects a sample document as a representative for each cluster. Upon receiving document or cluster feedback from users, a method which joins the probabilistic model and the vector space model is applied by the search assistant for query expansion. Finally, implicit query expansion occurs automatically, even if there is no feedback from the user.

3.1 Document Clustering

Cluster analysis has long been applied in information systems to improve the efficiency and effectiveness of retrieval [11]. The basic principle behind clustering is to group objects that have high similarities into the same

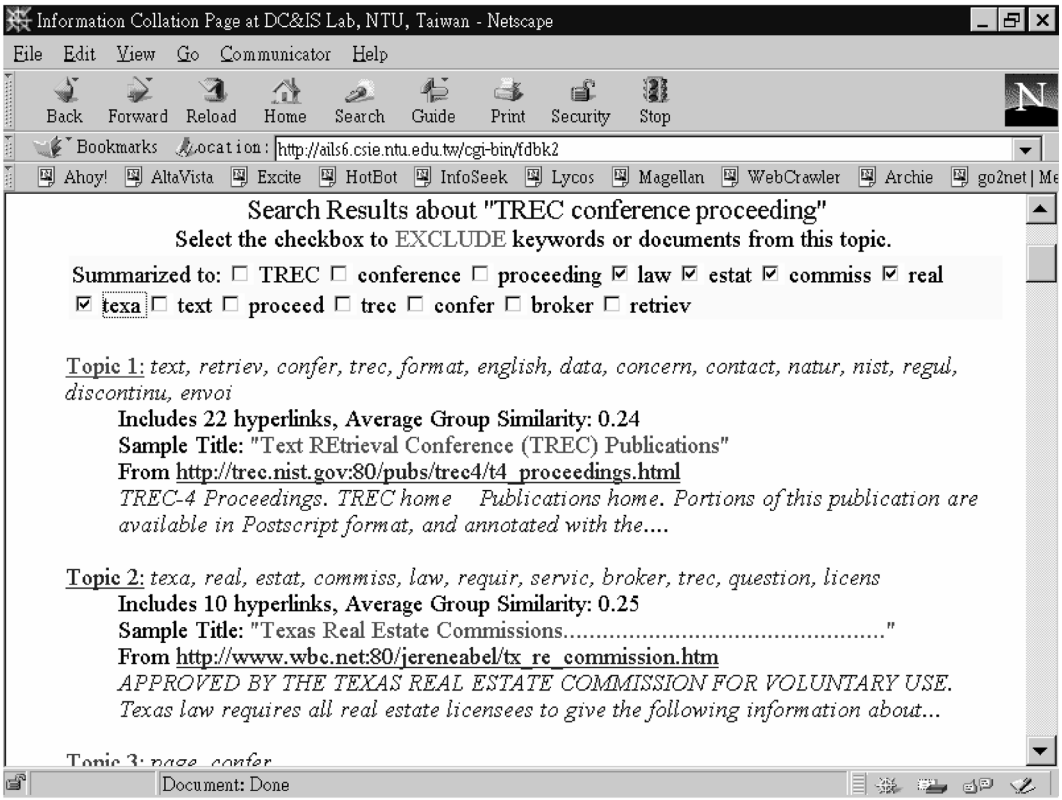

Fig. 2. The first two clusters for query "TREC conference proceeding."

cluster. Various clustering algorithms differ in measuring the similarity between clusters. For document clustering here, the measures of association between two documents d_i and d_j are determined by the inner product of the two document vectors, $v(d_i)$, $v(d_j)$:

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

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

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

where

$tf_{i,t}$	is the frequency of term t appearing in d_i,
$tfmax_i$	is the largest $tf_{i,t}$ in document d_i,
df_t	is the document frequency of term t in our collection G, where documents are sampled from the internet to give an estimation for the occurring frequency of a term, and
N	is the total number of documents in collection G.

Given the basic definition for the similarity measure between two documents, we then decide the clustering algorithm for this scenario. The clustering algorithm we adopt here is a modification of the hierarchical agglomerative clustering methods (HACM) [11], [16]. The general algorithm for the HACM is derived by identifying the two closest clusters and combining them into one cluster. The commonly used HACM includes single link, complete link, and group average link algorithms, which differ in their measures of the intercluster similarity and result in *dendrograms* with different structures (see [11], [16] for more details). To get an intermediate structure of the clustering results, we can use the *group average values* of the pairwise links within the cluster as the similarity measure (see (3)) where $|C_i|$ represents the number of documents in cluster C_i.

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

The time complexity is $O(n^2)$ computations of similarity measures for n individual documents in the stored matrix approach (with storage requirement $O(n^2)$). However, the similarity measure is not the main factor dominating computation time since documents are collected online from other search engines, thus creating a network delay during access. To make full use of time while waiting, clustering is conducted every m ($m = 30$) documents with the contents of the documents represented by a short description or summary from the search engines. The results of the $\lceil \frac{n}{m} \rceil$ clustering are then collected for final clustering. The individual clustering is controlled by the termination condition, where no two clusters have inner cluster similarity greater than θ. This method for document clustering may not result in the best hierarchical structure.

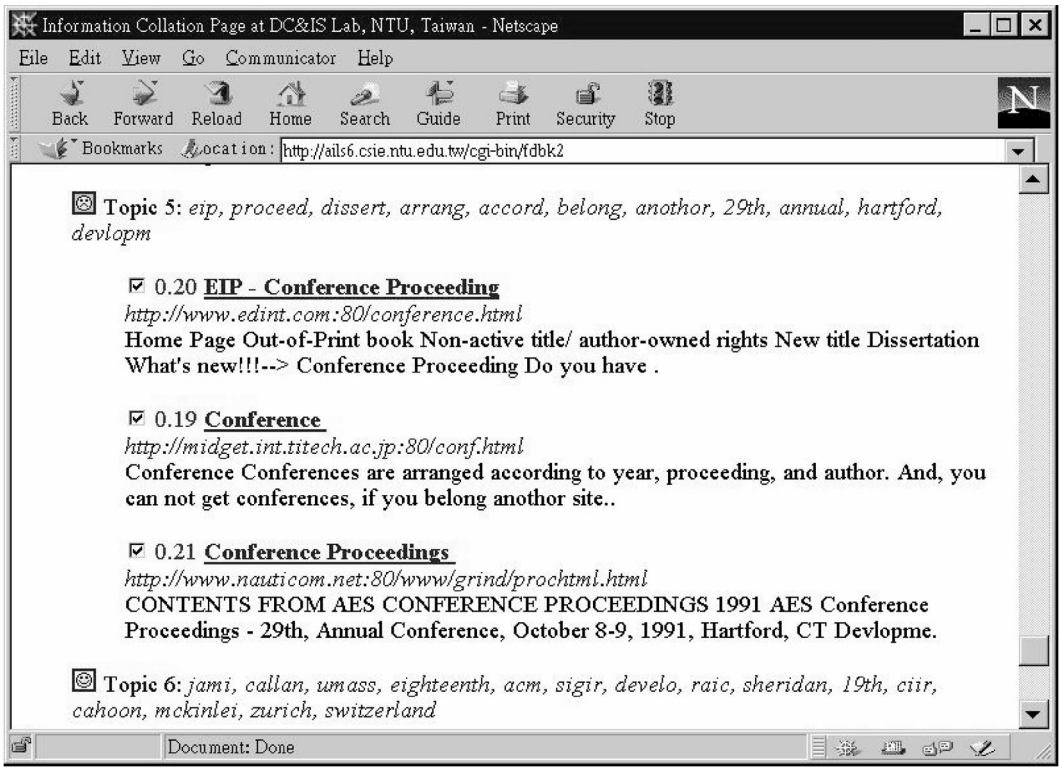

Fig. 3. Feedback signs: frowning face or smiling face.

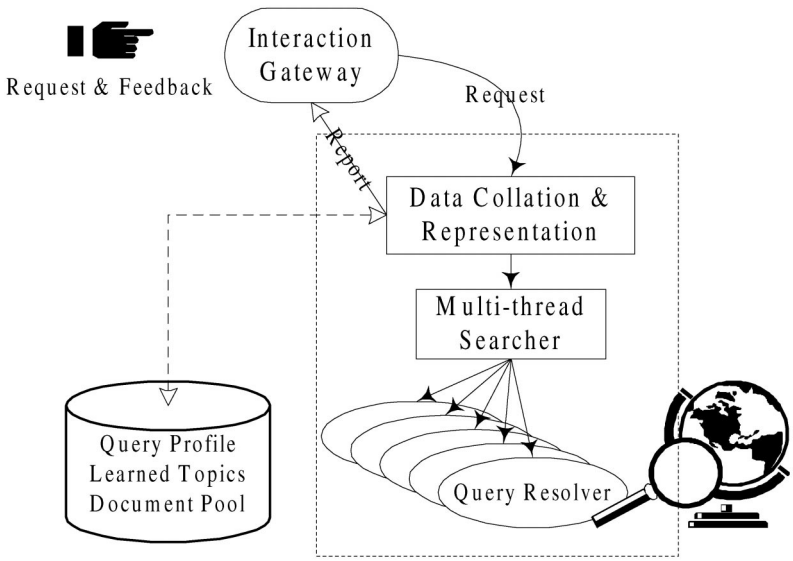

Fig. 4. System architecture.

However, with respect to partitioning documents into clusters, there is no difference between the original group average link clustering and the modified method where clustering is performed every m documents because no hierarchical presentation is actually given in the output. This modified approach, which extends the earlier iterative hierarchical clustering [2], employs the same concept as *fractionation clustering* [5] and also yields a suitable processing time for clustering.

Table 1 reports the time needed for collecting and processing (including clustering, cluster digesting, etc.) with respect to various numbers of documents. The collecting time (CT) refers to the time needed to collect a certain amount of documents from search engines, where each document is represented by the short summary or description (50-100 words) provided by search engines. The actual contents of the documents are fetched offline. It also reports the number of clusters produced and the cluster size (the number of documents in a cluster). For example, approximately 54 seconds are required to collect 60 documents from the search engines which results in an average of seven main groups (i.e., clusters with equal to or more than three documents). The last column of this table shows the percentage of singleton groups (clusters that

TABLE 1
The Time Needed for Collecting (CT) and Processing (PT), with Respect to the Number of Documents Collected

n	CT	PT	No. of Groups	1st	2nd	3rd	4th	5th	Singleton
30	26″	4″	3.6	10.4	4.6	2.7	1.9	1.3	34.9%
60	54″	9″	7.0	14.7	7.0	6.0	4.2	3.4	33.6%
90	1′22″	15″	9.3	17.6	9.6	7.1	5.8	4.5	33.8%
120	1′48″	22″	13.3	20.6	10.5	8.4	7.2	6.0	35.0%
150	2′15″	37″	16.1	24.1	11.1	9.1	7.4	6.2	35.6%

Note: The fifth column shows the number of documents per cluster in a decreasing order, and the 10st column shows the percentages of dangling documents which are difficult to classify.

have only one document) left after clustering. Usually, these documents are short and difficult to classify. Thus, in the final output, the dangling documents are not shown except for those with a very high similarity measure to the query. However, this does not imply that the user has no chance to browse these documents. With query expansion occurring at each interaction, singletons may be selected for inclusion in the following run. On average, with a similarity threshold θ set to 0.25, about 35 percent of the initial documents are left as singletons. In the following experiments and also in practical use, the retrieval is limited to 60-100 documents to keep the response time at about two minutes.

In a hypermedia environment such as the Web, hyperlinks are often viewed as a helpful tool in determining relationship between documents, since they indicate the intentional relationship between two hypertexts. However, these links are of very little use to us at all. An interesting phenomenon is that a high percentage of the links (94 percent to 97 percent) from a query result are located in the same Web server. Since 70 percent of the links from these crosslinks have a similarity value higher than the threshold $\theta = 0.25$ as shown in Table 2, clustering based on the the resemblance computation alone can result in at least 70 percent confidence even without analyzing links in search result. Thus, link analysis though provides a more accurate measure of document relationship, is not used in our clustering method.

3.2 Cluster Digesting and Naming

After preliminary clustering, the second step of the postprocessing is cluster digesting and naming. By cluster digesting, we mean the selection of representative words from a cluster and naming means the assignment of a representative document to that cluster. To select representative words for each cluster, we propose a two-phase selection algorithm which combines the majority principle and probabilistic term weighting. During the first phase, words that have *sufficient* postings (appearing frequency) in the initial document set are selected. Since approximately 75 percent of the words in the initial query results have less than four postings (see Fig. 5), only 25 percent of words are considered in the second phase. At the second phase, selected words are ranked by their *normalized cue validity*, which is computed by multiplying the *cue validity* with the inverse document frequency. The normalized cue validity of a term t with respect to a cluster C is formulated as follows:

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

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 represents the document set that the Web search assistant has collected (as discussed in (2)). The global corpus continues growing as more documents enter the database.

The idea of cue validity, a measure similar to probability weighting, is to highlight those words that have a distinguished occurring frequency in a cluster with respect to its complement cluster (documents not belonging to the cluster). 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.

TABLE 2
Analysis of Crosslinks in the Initial Query Results (Over 28 Queries)

n	No. of Crosslinks	From the Same Host	Avg. Similarity	Similarity$> \theta$
30	5.0	97.1%	0.35	79.7%
60	10.4	95.9%	0.37	73.5%
90	17.4	95.9%	0.35	70.9%
120	24.9	94.1%	0.37	67.2%
150	29.0	94.7%	0.35	65.3%

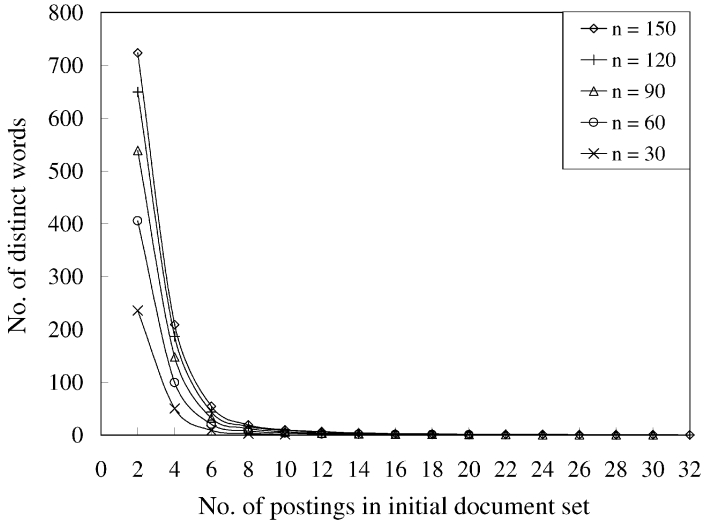

Fig. 5. Distribution of words according to number of postings in the initial query results.

From our experiments, the words selected for summary are those with sufficient postings in cluster C and less occurrence in the global corpus G (due to the inverse document frequency factor). For example, the first two clusters from the former query "TREC conference proceeding" are summarized after processing with "*text*, *retriev*, *confer*, *trec*, *format*, *english*, *data*, *concern*, *contact*, *natur*, *nist*, *regul*" and "*texa*, *real*, *estat*, *commiss*, *law*, *requir*, *servic*, *broker*, *trec*, *question*, *licens*" after digesting (Fig. 2). The words have been stemmed using suffix stripping implemented by Porter [10]. In summary, this equation shifts the emphasis from the rare words stressed by inverse document frequency to relatively significant words in cluster C.

As for the selection of a sample document, two parameters are considered. The first is the proximity of a document to the *cluster centroid* defined as the summary vector of the cluster. 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 (least number of slashes) is chosen as the sample document, since pages with a decreased URL depth often provides a more general idea of a Web site and its contents.

3.3 Query Modification Mechanisms

The third step of the postprocessing requires feedback from the user for query modification. As we discussed in Section 2.1, Rocchio's algorithm is the most employed method in the *vector space model*, which uses (document) vector addition and subtraction for query modification. Following Rocchio's principle, we design two additional approaches which employ the digesting mechanism as discussed in Section 3.2 for query modification. The first approach utilizes the clustering results and combines digested cluster vectors for query modification, while the second approach merges the digested vectors from the dichotomous, relevant and irrelevant, sets.

In the first approach, the query is modified by the cluster digests (or called summary vectors) constructed in Section 3.2. Such an approach is intuitive since less relevant words may have been eliminated at the digesting step. We call this *cluster-based query modification* as opposed to conventional *document-based query modification*, which is discussed in the following. Formally, 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:

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

where

- Φ_t and Ψ_t are the relevant and irrelevant cluster sets, respectively;
- $V(C_i)$ is the summary vector for cluster C_i; and
- β and γ are specified as the number of documents in a cluster which control the relative contributions of 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:

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

where β and γ are specified as those in (5). The idea in the first approach is to apply cluster digesting for term selection as in the probabilistic model and update the query by cluster-based vector merging, whereas the second approach exploits cluster digesting before vector merging in a dichotomous method. In order to implement both methods, in our feedback design users first decide which groups are most relevant to the query by the smiling face and frowning face, then use checkboxes to indicate document relevance for further refinement (as shown in Fig. 3). The effects of these two approaches are detailed in Section 4.2.

3.4 Implicit Query Expansion

In this subsection, we discuss implicit query expansion even when no specific feedback from the user is available. In this case, the first n documents in the initial results are considered as relevant documents to the query. Given that almost all documents in an information database are likely to be irrelevant to the specific query, the global corpus is defined as the complement set in the formulation of cue validity. Then, the digesting mechanism can be applied to select key words for expansion. However, since a small number of documents (say 60) when compared to the global corpus (say hundreds of thousands of documents) is lopsided, there is a great chance that infrequent words in the global corpus are stressed by (4) even if they do not occur frequently in the relevant set.

Hence, we scale the relative document frequency of words in the global corpus, $P(t|G)$, to lessen the inequality. We use three different scales: square root, cubic root and logarithms such that $P(t|G)$ remains in the 0 to 1 range and differences are amplified for small $P(t|G)$. The results, as one may expect, indicate two extremes. Equation (4) generates summaries with rare words, while (4) with the logarithm scale produces common words instead. From our experiments, we found that square root scaling presents the best outcome. This scaling value refers to the sizes of the two document sets. Indeed, it approximates the value p in equation $n^p = N$, where n is the number of documents in the initial query results and N is the number of documents in the global document set G. Thus, the score is computed with respect to the global set and is formulated as follows:

$$scv_{t,Q} = \frac{P(t|Q)}{P(t|Q) + \sqrt{[p]P(t|G)} + \epsilon} \tag{7}$$

Again, weighting is applied to the top 25 percent of the words that have postings higher than the remaining 75 percent. The selected words then comprise the summary vector for the query. Finally, the query is expanded by adding this summary vector to the original query by (5).

3.5 The Interface

In this subsection, we describe the clustering output in more detail. Continuing with the search results of the "TREC conference proceeding" query, the "Summarized to" section in Fig. 2 contains words that are extracted via implicit query expansion (Section 3.4) and automatically added to the query. The user can *exclude* unsolicited words by clicking the checkboxes. In this example, the query is summarized to " where the words "*texa, real, estat, commiss, law, broker*" are clicked to be excluded in the refinement 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 (Section 3.2).

As 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 relevance 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 excluded, and vice versa. Users can also indicate relevance of an 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 discussed in Section 3.3 are applied. For example, refinement of the "TREC conference proceeding" query is shown in Fig. 6, 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. In summary, Fig. 2, Fig. 3, and Fig. 6 can be seen as the results of a complete query process: first, a user commences a query "TREC conference proceeding" and gets the search results grouped in clusters (Fig. 2); the user then browses the details of each topic (Fig. 3); finally, the user refines the query to exclude irrelevant clusters and get further results (Fig. 6).

4 Evaluation of System Performance

In this section, we evaluate the performance of individual techniques and also evaluate overall system performance. From a macro analysis viewpoint, we want to know how well clustering helps an information search process and how well elimination of singleton clusters can affect retrieval performance. We would also like to evaluate the effects of query modification mechanisms under the cooperation of cluster digesting and different query modification mechanisms when explicit user feedback is given.

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. Sixteen trials (listed in Appendix A) are used in the experiments. After the multithread 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.

4.1 Evaluation of Clustering

First, to see the clustering results, the percentage of documents relevant in each cluster is depicted in Fig. 7. To obtain this result, the clusters in each group are first divided into three groups according to the number of

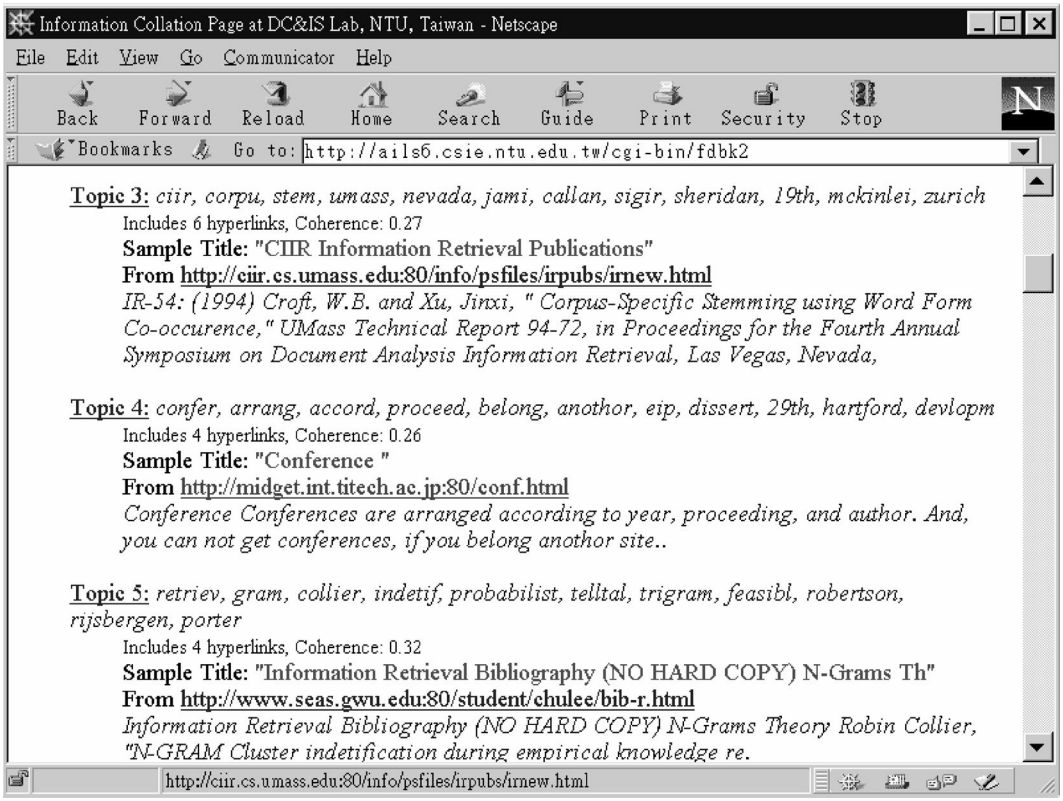

Fig. 6. The result of query "TREC conference proceedings" after refinement.

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 nonsingleton 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 is either relevant or irrelevant.

As plotted in Fig. 7, the curve of cluster size 5 to 7 has a nonlinear decrease which separates the clusters into two sets. The clusters on the left have at least 65 percent relevance and the clusters on the right always have less than 20 percent 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 3 shows the precision of documents in all clusters of a query result (column Cluster-All) and also the precision of documents in relevant clusters of a query result (column Cluster-Rel) at various cutoffs. Specifically, at a cutoff value of 60, assume that the clustering result produces seven main groups and 30 singleton clusters (Table 1). The precision for documents in all clusters (Cluster-All) are measured from the 40 documents in the seven main groups. This value is then compared to the precision of similarity ranking at a cutoff value of 60 (column Sim-Ranked), where a 11.6 percent improvement is gained as indicated in Table 3. From another viewpoint, if we consider clustering as a filtering technique that filters out singleton clusters, the 40 documents in the seven 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-Rel). 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 percent), a cluster often has at least 60 percent precision if the sample document is identified as relevant. In this measure, Cluster-Rel has a precision improvement as high as 57.1 percent at a cutoff value of 60 (see Table 3). 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 4, the penalty for a 40.7 percent increase in precision is an 6.7 percent decrease in relative recall at a cutoff value of 60. 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.

4.2 Evaluation of Digestion and Query Modification

Upon designing a query modification mechanism for concept-based feedback following the vector merging

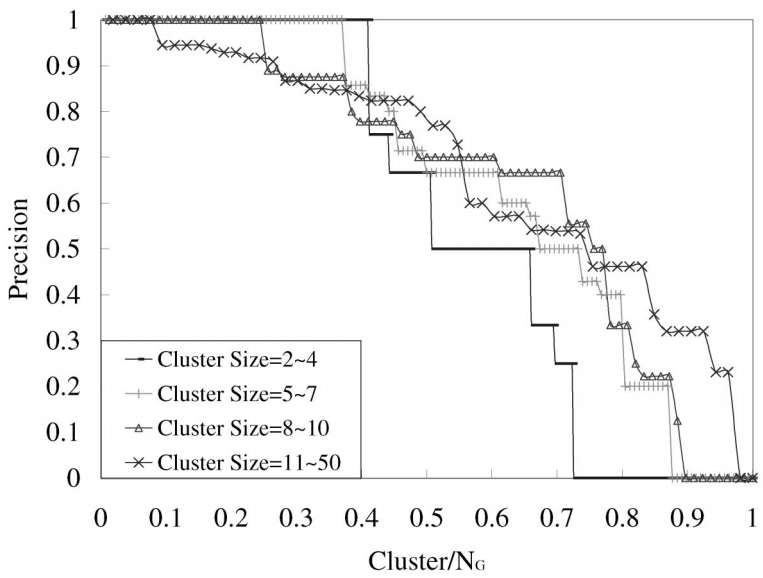

Fig. 7. The percentage of documents found relevant for each cluster ranked by decreasing precision.

TABLE 3
Comparison of Precision for Similarity Ranking and Clustering

	Precision Increase For Clustering				
n	Sim-Ranked	Cluster-All	Increase	Cluster-Rel	Increase
30	0.575	0.677	17.8%	0.867	50.8%
60	0.533	0.595	11.6%	0.838	57.1%
90	0.490	0.638	33.2%	0.831	69.8%
120	0.464	0.618	30.1%	0.791	70.6%
150	0.436	0.567	39.4%	0.786	80.4%

Note: Cluster-All refers to percentage of relevant documents in all clusters of a query result (without singleton clusters). Cluster-Rel refers to percentage of relevant documents in relevant clusters. The Increase column refers to comparison with similarity ranking.

TABLE 4
Comparison of Cluster-All and Cluster-Rel

	Relative Recall Degradation vs. Precision Increase For Clustering					
n	Precision-A	Precision-R	Increase	Recall-A	Recall-R	Decrease
30	0.677	0.867	28.0%	0.692	0.630	-9.0%
60	0.595	0.838	40.7%	0.750	0.700	-6.7%
90	0.638	0.831	30.3%	0.635	0.604	-5.0%
120	0.618	0.791	28.0%	0.690	0.637	-7.6%
150	0.567	0.786	38.6%	0.584	0.526	-10.0%

Note: 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.

principle, two control parameters concerning us most. The first is whether digested vectors are better than original vectors in query modification. The second is whether the clustered results are better than unclustered results when taken as vector merging units. While evaluating the effects of these two parameters, we also show how much degradation in performance occurs when singleton clusters are excluded as in real situations. In the following experiments, we use 60 documents from similarity ranking as training examples in relevance feedback. The modified query vector is then used to give weights to all 150 documents. Four experiment sets are used to measure the effects of digesting and clustering at different combinations.

First, to evaluate the effect of digesting, the training examples are divided into relevant documents and irrelevant documents. In one experiment set, these two groups are first digested as summary vectors by (4) and then merged with the original query vector by (6). In the second experiment set, the individual document vectors, without undergoing digestion, are merged with the original query vector using (8), which results in the original Rocchio algorithm.

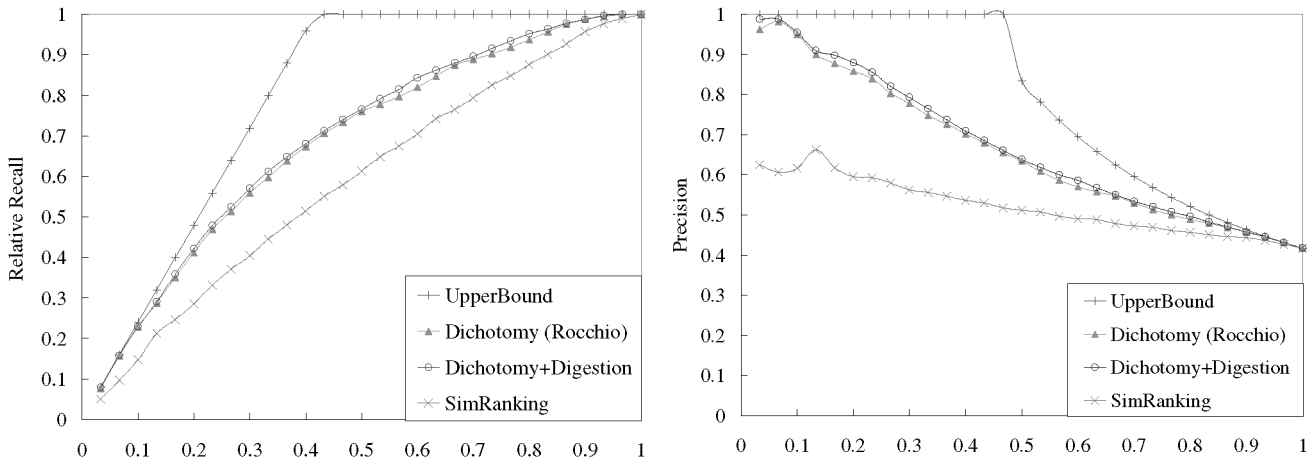

Fig. 8. Comparison of relative-recall and precision for the digestion approach and Rocchio's algorithm. The plots show *relative recall* and precision as a function of the fraction of documents retrieved out of a total of 150 documents. The performance obtained by retrieving document order and the upper bound are also plotted for comparison.

$$Q_{t+1} = Q_t + \sum_{d_i \in R_t} v(d_i) - \sum_{d_j \in S_t} v(d_j) \tag{8}$$

Both Rocchio's algorithm and the digesting approach are optimized to get the best performance. For Rocchio's algorithm, since the basic operation is to merge document vectors and original query vectors, queries are automatically expanded by adding all the terms not in the original query that are in the relevant documents and irrelevant documents. However, after merging, only 50 or so terms with the highest weights are retained in the query vector to obtain the best performance. As for the digestion approach, weighting of terms by (4) extracted an average of 50 terms for the relevant cluster and irrelevant cluster respectively. The summary vectors are then merged with the original queries by (6). Finally, 60 terms are selected for each query to yield the best performance.

The performance is measured by *relative recall*, defined as the fraction of relevant documents from the 150 documents, and *precision*, defined as the fraction of documents that are relevant. Fig. 8 shows the results for these two experiment sets. From this figure, it is shown that the digestion result has slightly better performance (1.3 percent average) than the result without undergoing digestion.

The next two experiment sets are designed to show the influence of digestion in a scenario when clustering is applied. This time, 60 documents are first clustered into approximately 5 to 7 groups. We assign relevance to each cluster according to the relevance of the sample title for each cluster. Then, in one experiment, the clusters are digested to merge with the original query by (5). In the other experiment, the document vectors are added to merged with the original query as follows:

$$Q_{t+1} = Q_t + \sum_{C_i \in \Phi} \sum_{d_j \in C_i} v(d_j) - \sum_{C_i \in \Psi} \sum_{d_j \in C_i} v(d_j) \tag{9}$$

These two experiment sets simulate a cluster-based feedback scenario where the user browses only the sample titles and assigns relevance to clusters. Note that since singleton clusters are excluded from the training examples, the actual training set is only two-thirds of 60 documents. The results of the experiments are displayed in Fig. 9 with digestion as the control parameter. This time, the approach with digestion performs slightly worse (–1.4 percent) than the approach without digestion. In fact, the difference that digestion can make is very small.

In summary, with digesting and clustering as control parameters, four combinations are plotted on a precision-recall coordinate axes (Fig. 10). The results show an average of 5 to 10 percent performance degradation for the cluster-based modification compared to document-based modification (the ones without clustering). Indeed, because of elimination of singleton clusters after clustering, there are only two-thirds the number of training examples in the second two experiment sets, resulting in inferior results because of fewer chance to learn the real meaning of the user's query. Furthermore, since only sample document is given relevance information, there is chance that irrelevant documents are assumed relevant when they mix in a relevant cluster. Hence, the cluster-based modification is inferior to the dichotomy approach using document-based modification.

In fact, what users really browse are those main clusters presented in the output, since singleton clusters are removed intentionally. The system is designed such that users can just browse sample documents of clusters to give feedback for these clusters. If extra time is available, users can also browse individual documents to give detail feedback for those document. To facilitate both kinds of relevance feedback, our implementation adopts a document-based vector merging mechanism, where we can save the efforts without maintaining cluster states. To show the practical performance we can get when singleton clusters are removed, further experiments are conducted. The first experiment simulates the scenario when the assignment of relevance to each document is available, while the second experiment describes the case when only the sample documents are given relevance information such that other documents in the cluster are assigned the assumed relevancy accordingly. Both these experiments show better performance than the cluster-based approach as shown in Fig. 11.

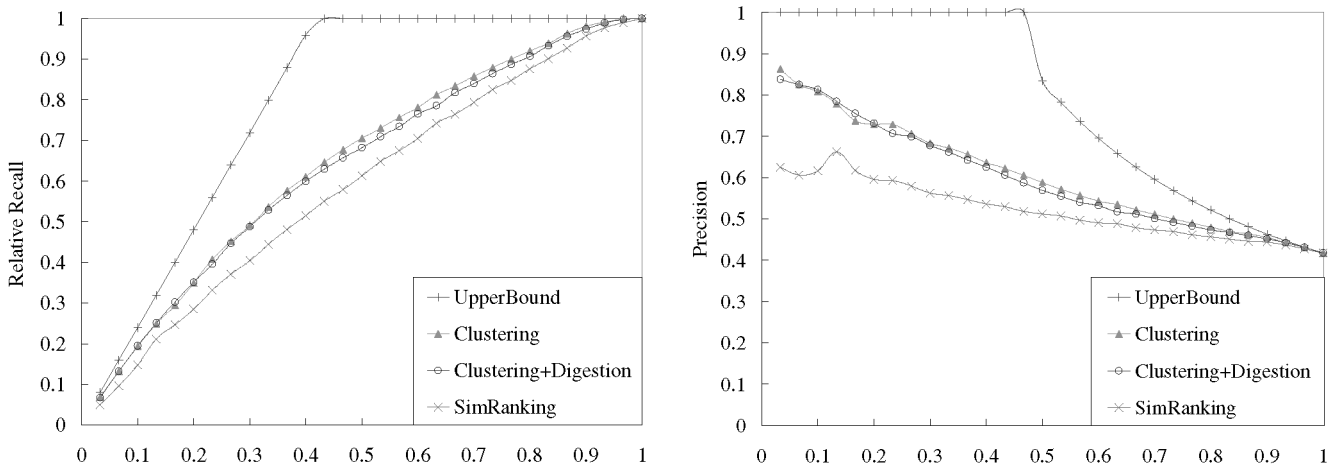

Fig. 9. Performance comparison showing digestion effects under clustering-based feedback scenario.

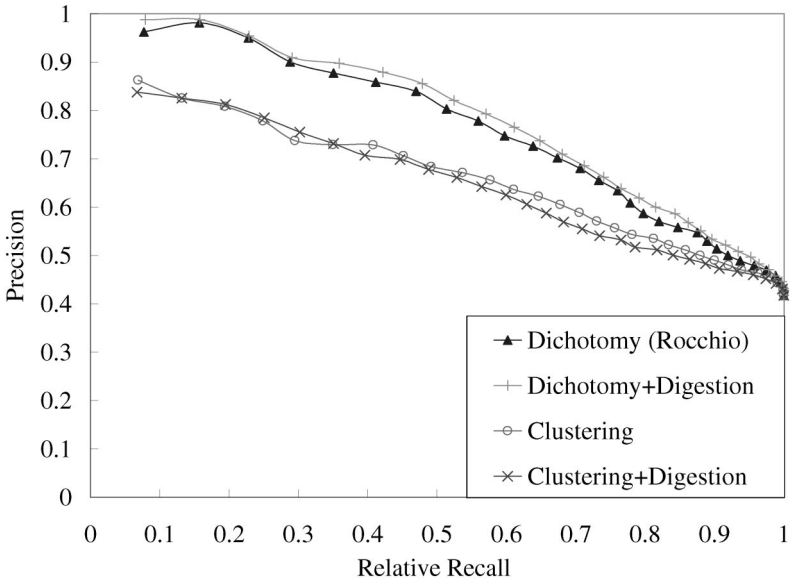

Fig. 10. Comparison of various combinations of two control parameters, digestion and clustering, plotted on relative recall vs. precision scales.

4.3 Discussion

Performance evaluation for information search tools on the Web is tough work. The experiments we reported here are made by real users giving relevance to retrieved documents. We evaluate the actual performance for individual techniques used in this section in terms of precision and relative recall. The main purposes of these two subsections are to show how much performance we can obtain from cluster-based browsing and to show what query modification mechanisms are suitable for this scenario. From Section 4.1, we learned that proper clustering can improve system performance with less browsing effort from users. With a better presentation of the search results, users can access targets more quickly. The modified query vectors (using the top 60 documents) can highly (above 50 percent) improve retrieval precision for later retrieval.

Also, in Section 4.2, the performance degradation for cluster-based modification compared to document-based modification is an important clue in the design of a relevance feedback interface. On one hand, we can design cluster-based feedback which requires more clustering states to be maintained for following interactions, but with some performance degradation. On the other hand, we can design document-based feedback to simplify the feedback design without maintaining clustering states and to obtain better performance. Hence, a wise decision for a Web search assistant will be to employ document-based query modification (6) instead of cluster-based modification (5). Such an approach can offset inadequate clustering for large clusters, and at the same time, maintains the convenience of cluster-based browsing.

Note that in terms of Web information retrieval, techniques such as global document clustering by Cutting et al. [5] and Anick and Vaithyanathan [1], whether dynamic or static, are not practical due to the large volume and volatility of the Web. As we can predict, the bigger the database, the less easy it is to well define a cluster. Hence, applying document clustering to the search result of a query can give a more specific idea of what the query can retrieve. What differs here from the Scatter/Gather by Hearst and Pedersen [8] is that the feedback of relevant clusters and documents is not only gathered for clustering

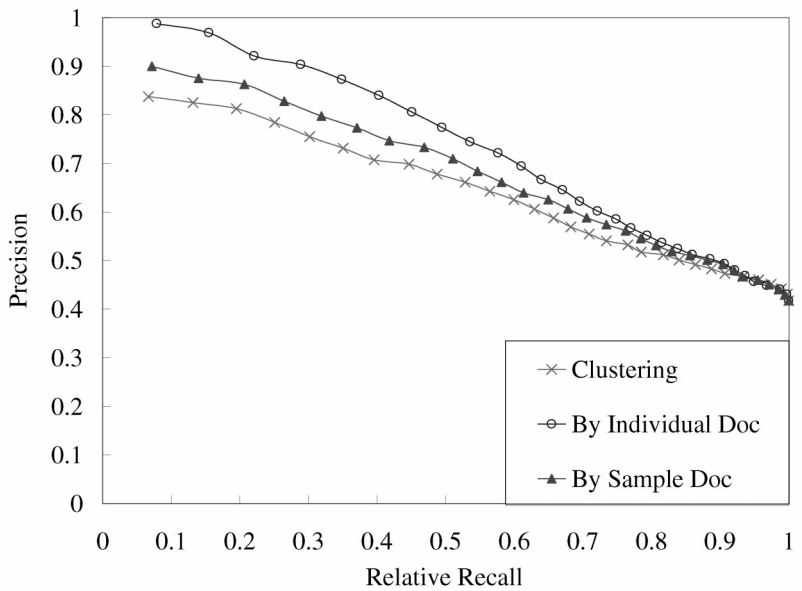

Fig. 11. Actual performance of document-based query modification with singleton clusters eliminated. The performance is better than the former cluster-based approach.

a second time but rather the query is modified to better formulate the information required. In some way, the Scatter/Gather paradigm is more like a filtering tool that narrows down the search area from documents retrieved by the original query as reported in their user study. No modification is made to the original query. In fact, selecting the promising documents is a kind of feedback that we can take advantage of to interpret and to learn what a query stands for. In this paper, we not only verify the increase of precision in relevant clusters (with moderate decrease in relative recall), but also employ the application of the document-based query modification mechanism to obtain better interpretation of a query. Such interpretations for queries can be used for weighting or retrieving documents in later interactions. The experiments in this section explain the improvement in precision for the cluster-based query modification approach.

5 Conclusions

Searching for desired data on the Web is one of the most common ways the Web is used. A lot of efforts are put into the research of information retrieval on the Web. In this paper, we propose document clustering and query expansion as the main technology in concept-based relevance feedback. This idea is inspired from the observation that most users give very little information as query input in standard search methods. Thus, we try to apply the clustering technique to summarize related topics from similarity-ranking search results for better data presentation, and explore new techniques for query expansion via 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 are confined to a limited number of topics and relevance is easily decided when giving feedback. Indeed, this is the basic motivation behind *local* document clustering. 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 cluster-based principle is especially useful for short queries that encompass a wide range of topics because of word sense 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 special topics, other techniques should be considered since categorization does not simply depend on similarity measures but rather on arbitrary categorization.

Our system is based on a multithread search architecture where multiple search engines are incorporated to overcome deficiency such as limited availability and coverage as occurs with individual search engine. This common architecture is adopted for many multiengine searchers such as Savvy Search [6] and MetaCrawler [14], etc., to integrate the results of multiple heterogeneous databases, where different approaches of data collation are applied.

This system can also be implemented for real Web information search applications. We consider this system as a personal or proprietary search assistant. Given all the queries the user has submitted, the constructed summary of each query could be viewed as a detailed profile of the user's information need. By organizing the queries into proper interest groups, the Web search assistant can suggest related queries and give hints for query formulation at a later time. Furthermore, other automatic discovery mechanisms can be implemented based on this system, such as *hyperlink-based traversing* that reviews information on behalf of users from promising links [3].

APPENDIX A
LISTS OF QUERIES EXPERIMENTED

1. OCR
2. TARGON
3. ActiveX
4. flamenco
5. data mining
6. emulator rom
7. key recovery
8. Michael Klim
9. Delphi TImage
10. computer virus
11. australia hotel
12. information agent
13. Age of empires II
14. musicals and opera
15. distributed systems
16. TREC conference proceeding

REFERENCES

[1] P.G. Anick and S. Vaithyanathan, "Exploiting Clustering and Phrases for Context-Based Information Retrieval," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* pp. 314–323, 1997.

[2] C.H. Chang and C.C. Hsu, "A Customizable Multiengine Search Tool with Clustering," *Computer Networks and ISDN Systems,* vol. 29, nos. 8-13, pp. 1,217–1,224, Sept. 1997; also appeared in *Proc. Sixth Int'l WWW Conf.,* Apr. 1997.

[3] C.H. Chang and C.C. Hsu, "Exploiting Hyperlinks for Automatic Information Discovery on the WWW," *Proc. 10th IEEE Int'l Conf. Tools with Artificial Intelligence,* Chien Tan Youth Activity Center, Taipei, Taiwan, Nov. 1998.

[4] W.B. Croft, R. Cook, and D. Wilder, "Providing Government Information on the Internet: Experiences with Thomas," *Proc. Digital Libraries Conf.,* pp. 19–25, 1995.

[5] D.R. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey, "Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* pp. 318-329, 1992.

[6] D. Dreilinger and A. Howe, "An Information Gathering Agent for Querying Web Search Engines," Technical Report CS-96-111, Colorado State Univ., 1996.

[7] D. Harman, "Relevance Feedback Revisited," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* pp. 1-10, 1992.

[8] M.A. Hearst and J.O. Pedersen, "Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* 1996.

[9] B. Pinkerton, "Finding What People Want: Experiences with the WebCrawler," *Proc. Second Int'l WWW Conf.,* Chicago, 1994.

[10] M.F. Porter, "An Algorithm for Suffix Stripping," *Program,* vol. 14, no. 3, pp. 130–137, July 1980.

[11] E. Rasmussen, "Clustering Algorithms," W. Frakes and R. Baeza-Yates, eds., *Information Retrieval: Data Structures and Algorithms,* chapter 16, Prentice Hall, 1992.

[12] S.E. Robertson and K. Sparck Jones, "Relevance Weighting of Search Terms," *J. Am. Soc. Information Science,* vol. 27, no. 3, pp. 129–146, 1976.

[13] J.J. Rocchio, "Relevance Feedback in Information Retrieval," G. Salton, ed., *SMART Retrieval System,* Prentice Hall, pp. 313–323, 1971.

[14] E. Selberg and O. Etzioni, "Multi-Engine Search and Comparison Using the MetaCrawler," *Proc. Fourth Int'l WWW Conf.,* Boston, 1995.

[15] E. Voorhees, "Query Expansion Using Lexical-Semantic Relations," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* pp. 61-69, 1994.

[16] P. Willett, "Recent Trends in Hierarchic Document Clustering: A Critical Review," *Information Processing and Management,* vol. 24, no. 5, pp. 577–597, 1988.

[17] J. Xu and W.B. Croft, "Query Expansion Using Local and Global Document Analysis," *Proc. ACM SIGIR Int'l Conf. Research and Development in Information Retrieval,* pp. 4-11, 1996.

Chia-Hui Chang received her BS degree in computer science and information engineering in 1993 from National Taiwan University, and is now a PhD candidate in the same department and institute. Her research interests include distributed computing, intelligent agents, and digital libraries, with a focus on information retrieval on the World Wide Web. She is a student member of the IEEE and the IEEE Computer Society.

Ching-Chi Hsu received the BS degree in physics from National Tsing Hwa University in Hsinchu, Taiwan, in 1971; and the MS and PhD degrees in computer engineering from National Taiwan University, Taipei, Taiwan, in 1975 and 1982, respectively. He joined the faculty of the Department of Computer Science and Information Engineering at National Taiwan University in 1977, and became an associate professor there in 1982. Currently, he is a professor and head of the department. His research interests include distributed systems, distributed processing of data and knowledge, information retrieval on the Internet, constraint programming, genetic algorithms, simulated annealing, and scheduling theory.