

An Interface for Navigating Clustered Document Sets Returned by Queries

Robert B. Allen, Pascal Obry, and Michael Littman
Bellcore
MRE 2A367
445 South Street
Morristown, NJ 07960 USA
{rba, obry, mlittman}@bellcore.com

ABSTRACT

An interface has been implemented for exploring the structure of document sets returned in response to a query. The interface allows a user to find subsets of documents that are especially relevant to the query through interaction with an *interactive dendrogram* which displays a hierarchical clustering of the documents. Dynamic lists of document titles are interlocked with the dendrogram to provide detail of the clusters being viewed by the user. For efficiency, the interface has been implemented in several parallel and distributed computation environments and has been applied to retrieval of encyclopedia articles and news stories.

KEYWORDS

Clustering, information retrieval, interfaces.

1. INTRODUCTION

Classical information retrieval systems are based on the idea that each document in a collection can be compared to a user's query, scored as to its similarity, and displayed in similarity-sorted order. However, this linear sorting may obscure structure among the documents. For instance, the titles of the top 15 encyclopedia [1] articles returned by a latent semantic indexing search (LSI) [5] for the query "Tell about scientific studies of air pressure" were:

bellows
front
air conditioning
whirlwind
air force
compressor
air mass
fog
pneumatic systems
Billy Mitchell
Strategic Air Command
United States Air Force Academy
inversion
mirage (image)
foehn.

Several of the articles such as *bellows* and *compressor* focus on devices which produce air pressure. Other articles seem to relate to weather, while still others reflect the semantic confusion between *air pressure* and *air force*. Finally, some articles such as *foehn* are simply obscure and their appearance in the sorted list carries no information for most people. However, seeing *foehn* grouped with *whirlwind* and *fog* might lead us to suspect that it weather related (in fact it is a type of mountain wind). Clearly, some type of simple categorization could help separate potentially relevant from clearly irrelevant articles.

With the availability of powerful workstations and parallel computers, it is possible to use a variety of methods to find and visualize structure in similarities between the documents of a return set. We have explored classical Torgerson-Young multidimensional scaling, with inconsistent results, and Kohonen nets [8], which have given some good results but are still computationally expensive and may not scale well. This paper documents the use of hierarchical clustering for presenting the structure of a retrieved set of documents. Hierarchical clustering has the advantage that it is less computationally expensive and lends itself to relatively easy

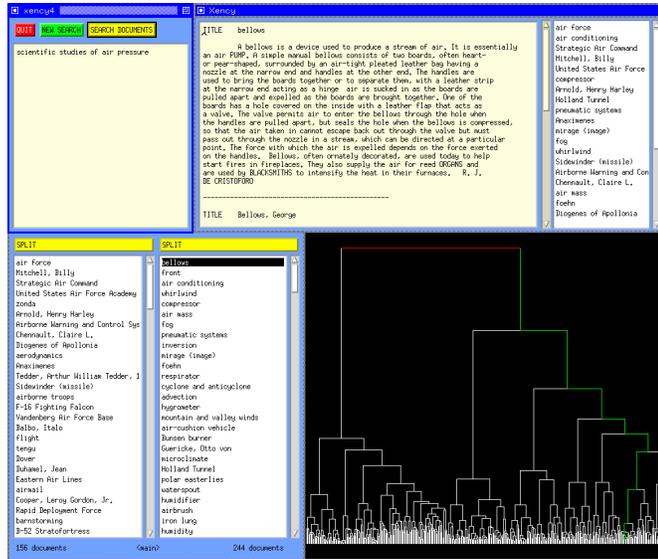


Figure 1: Cluster Interface for Encyclopedia.

presentation in an interactive interface. Clustering is a fairly common tool in information retrieval and has been used frequently to organize entire document collections for retrieval [12], as well as to organize return sets to queries about software modules [9].

Interfaces for displaying hierarchical structure have been developed in a variety of contexts such as descending menus and hypertext displays. Menu-based interfaces are often not suitable for trees resulting from statistical clustering because meaningful labels are not available (see Section 3.4). Crouch et al. [3] describe a Macintosh interface for browsing a static document clustering, but it seems to be useful for only a small corpus. Many class-hierarchy browsers (e.g., [2]) provide interfaces for navigating simple hierarchies. However, we do not feel that existing hypertext browsers are well suited for rapidly sorting through a large number of documents.

This paper introduces an interface for browsing relations among documents as derived by hierarchical clustering of returned document sets. The Motif X-windows interface, as depicted in Figure 1, consists of four major windows: the query window (upper left, where the user types free text queries for the system to process), the interactive dendrogram, the subtree document lists, and the text window and lists of proximal documents (all of which are described in more detail in the following sections).

The main concepts demonstrated by this interface are: (a) The interlocking functionality of graphics and text windows. (b) The use of the a dendrogram as a highly structured map of the document collection (the lack of which contributes to disorientation in hypertext systems). (c) Logical zooming [11] within subtrees of the dendrogram. This provides detail without a loss of global context.

2. INTERFACE COMPONENTS

In a typical interaction, a user enters a query into the system and the similarity between that query and all documents is computed. A fraction of the documents, those with the highest similarity to the query, become the return set and are compared with one another to identify structure *between* the documents. The system computes a hierarchical clustering of the returned documents and presents the resulting tree (dendrogram) to the user. Briefly, hierarchical clustering takes a matrix of distances between objects (in this case, similarity between returned documents), and creates a binary tree [6] in which similar documents are joined low in the tree and less related documents are not joined until higher in the tree. In this application, the leaves of the tree are individual documents and the root of the tree is the coarsest level of structure identified in the return set. The set of returned documents and the clustering tree itself are the main focus of the interface.

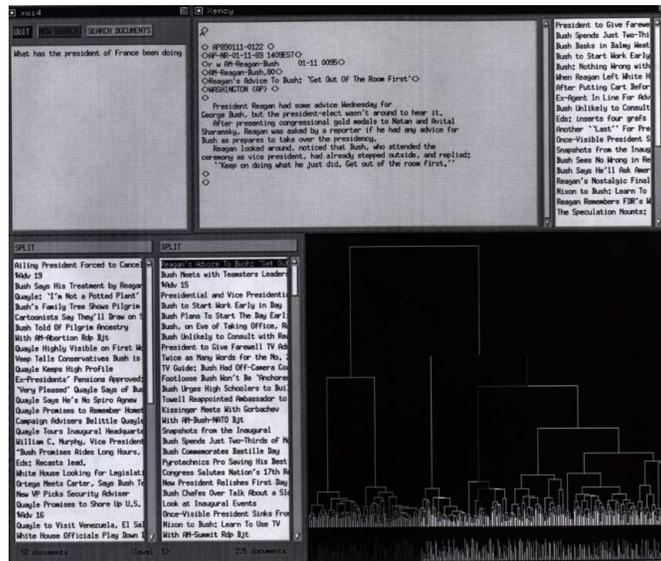


Figure 2: Interface for AP News Stories.

Two concepts are fundamental to the presentation of the returned documents. At any given time there is a notion of a “selected document” and a “selected subtree.” These represent a particular document and a particular portion of the tree that the user would like to focus on. Initially, the selected subtree is the entire return set and the selected document is the one with the highest similarity to the query, but this is quickly changed by the user in the act of browsing.

2.1. Interactive Dendrogram

In one window in the interface, the clustering tree itself is presented to give the user a sense of the overall structure of the return set. In the context of a given query, the display of the tree is static, but colors are used to indicate the selected subtree (a horizontal red bar across the root of that subtree) and the selected document (a red path from the root of the tree to the leaf representing that document). In addition, the highest similarity document within the selected subtree is highlighted by a green path from the root of the subtree to the leaf representing that document. At any time, the user, by clicking on a junction (i.e., the root of a subtree), can make a different subtree the selected subtree causing the display to be updated accordingly.

Figure 2 gives an example of the interface applied to viewing a collection of Associated Press News Stories. Here indicators at the bottom of the dendrogram show the similarity of each article to the query. The length of these indicators is proportional to the article's similar-

ity to the query (longer bars show articles that are close to the query). These bars may indicate a concentration of relevant articles and they effectively add a third dimension to the dendrogram.

2.2. Subtree Document Lists

The window on the bottom left displays two lists of article titles, one for each subtree of the selected subtree. In each list, the articles are ordered by their proximity to the query with the article with the highest similarity highlighted initially. At the top of each list is a “split” button, which makes that subtree the selected subtree and therefore splits it further. This function is similar to clicking on the dendrogram and provides the user with more information about a part of the tree he/she deems relevant. Of course clicking on dendrogram at the junction above the selected subtree will “un-split” the lists. Clicking on the title of a document makes that document the selected document, once again giving the user an opportunity to get more detail about a piece of relevant information.

2.3. Text Window and Lists of Proximal Documents

The window on the top right provides a “logical zoom” of the selected document. That is, the selected document, presumably because it has been judged relevant by the user, deserves extra screen real estate in the form of (a) its text and (b) the return set sorted by similarity to the selected article. The term “logical zoom” is used here because a “physical zoom” (that is, writing the selected title in a larger font) is not at all desirable.

Zooming in this context involves changing the form of the information (from title to text) and not just its scale, but it does provide similar functionality. At first glance, the list of proximal documents is redundant since the clustering tree also gives proximity information between the selected document and the others in the return set. However, the clustering tree is not a perfect model of the similarity structure and instead represents a “best fit” or compromise over the entire return set. As a result, documents that are far apart in the tree might actually be quite similar and dissimilar documents might be quite close in the tree. By listing the return documents with respect to proximity to the selected document, the distance information is depicted more accurately and in a manner particularly relevant to the user.

2.4. Usage

Now that the fundamental structure of the interface has been discussed, we can step through an example of how it can be used to respond to queries. Returning to the “scientific studies of air pressure” example, inspection of Figure 1 shows that the articles were clustered into two main groups: *air force* articles on the left and all others on the right. A user could descend the more promising right subtree by clicking on the right split button. Since this pulls out a cluster of articles on meteorology, the user might descend further down the right side of the tree to see clusters on heat, fluid dynamics, and sound. While articles on studies of air pressure are not concentrated in any one of the sub-clusters, they generally appear near the top of the lists for the major clusters and the clusters themselves give structure to the set of relevant articles.

3. IMPLEMENTATION

3.1. Corpus and Preprocessing

A set of 25,629 articles (those of more than 50 words in length) from the *Academic American Encyclopedia* [1] was employed because it was the largest collection that had been preprocessed using the latent semantic indexing tools. The documents were indexed by the 56,530 terms that occurred in more than two documents. The encyclopedia articles included cross-reference “See Also” links which were not used in the structuring procedure partly because these are used inconsistently (e.g., many asymmetrical links) and incompletely (e.g., many reasonable links left out) but also because we hoped that the interface described here would be applicable to cor-

pora that have no explicit cross-references.

LSI, a statistical technique that locates documents and terms in a high-dimensional space, was used to analyze the collection. This is done by generating a large, sparse, term by document matrix where each cell of the matrix contains the number of occurrences of the given term in the given document. This matrix is then analyzed using singular value decomposition (SVD). This finds a lower-dimensional vector representation for each term and each document such that the dot product of the vectors reapproximates the corresponding cell of the original matrix. To compute the similarity between a query and a document, we use the vector representations of the terms in the query. The cosine (normalized dot product) between the resulting vector and the vector representing the document is the similarity score used in this interface.

In the particular analysis performed here, the raw term by document matrix was transformed using entropy term-weighting with no normalization for the length of the articles. SVD generated a 322-dimensional vector for each term and document. Although the present interface uses an LSI retrieval engine, interactive clustering should be useful for any retrieval algorithm that returns an ordered list of documents.

3.2. Clustering

User-specified queries generate return sets consisting of the top 1.6% of the articles (the top 400 out of 25,000 in the case of the encyclopedia) and these are used to compute a clustering tree. Proximity in the resulting clustering is not a simple function of document to document similarity but is based on overall patterns of similarity in the 400 returned documents. The clustering is accomplished using Ward’s algorithm [6] which we chose after informally comparing several standard clustering algorithms on the basis of interpretability of the derived clusters.

Considerable computation is required to search the document vectors and generate the clustering. Willet [10] describes studies of parallel clustering algorithms; however, he appears not to have reported parametric tests of performance for hierarchical clustering. We compared a hierarchical clustering algorithm implemented on a 16K processor MasPar parallel computer to performance on a workstation (DEC 5000). The results are summarized

# of vectors	workstation	parallel
256	12	4
512	61	14
1024	343	36

Table 1: Seconds of processing time for clustering.

in Table 1. For return sets of 256 LSI vectors (each consisting of 322 real numbers), the parallel machine was 3 times faster than the workstation, while for sets of 1024 vectors the parallel machine was about 10 times faster. We estimate that an improved algorithm on a dedicated parallel computer for search and clustering could complete the entire operation in about 8 seconds per query (over 40 times faster than the sequential machine).

Because the MasPar computer was not consistently available, a version of the interface was developed in which the computation was distributed across a network of workstations.

Several methods, including global clustering and a massively parallel implementation, were tested to reduce the time for calculation of the distances between the query and the documents. The most practical method involved dividing the database into equal parts and loading these through the local network to 45 workstations. The vector calculated from a user's query was sent to each of these workstations which then computed its top matches and returned them for merging. Finally, the vectors for the top 400 articles were clustered and the interface updated, resulting in an overall response time of approximately one minute.

3.3. Performance

In spite of our use of advanced interface concepts such as giving windows interlocking functionality (that is, selections in one window affect the displays in other windows in a task-relevant way), we do not view this as an interface for a casual user. Nonetheless, our experience suggests that about 20 minutes of training is sufficient for technically-oriented users to become comfortable with the interface and capable of carrying out simple navigation tasks.

In our informal experiments with the encyclopedia, queries varied a great deal as to whether or not hierarchical clustering simplified the retrieval process. Clustering appears to be useful for retrieval when questions are composed of several parts or include terms with multiple meanings. This is because many spurious entries are

retrieved and the interface allows entire categories to be quickly discarded. For example, the following queries appeared to be facilitated by the interface:

future of electronic libraries
 history of rock and roll
 french explorers of north america
 extinction of the dinosaurs.

However, there are many situations in which the clustering did not seem to improve retrieval. One class of failures was due to classical information retrieval not being good enough to return the relevant articles for clustering. Another class involved classical information retrieval being too good, that is in which a single conceptual category fills the top of the return list. Furthermore, the interface is of little benefit when no appropriate answer is present in the database. However, there are also cases in which the clustering does not match the categories of interest to the user. Informally, however, it seemed that clustering was successful in most cases when the returned documents fell naturally into distinct categories.

3.4. Labeling of Cluster Nodes

A limitation of this approach is that collections with entries that have no concise description or title are difficult to search. The terse and informative titles in the encyclopedia are very helpful in guiding the user to fruitful material. Taking this a step further, in the hierarchical interface, there is much to be gained from somehow getting concise descriptions or titles of higher-level clusters.

We considered several techniques (cf. [4]) to achieve this, including presenting a small set of terms that are closest to the query centroid. However, none seemed more effective than the chosen approach of simply presenting the ranked document lists in similarity order. This way, when the user focuses on a given subtree a "description" of that subtree appears in the form of the list of documents that comprise it. Of course, this solution does not work in the case of collections of untitled documents but that must be considered as a separate problem.

We found that clustering an output list may show the scope of topics relevant to a particular query. For instance, the response to the general query "Tell me about telephones", included clusters that could be characterized as "electronics", "electrical devices", "information theory", and "finance." In this case, an encyclopedia

article on *telephones* was available (and was retrieved first in response to the query) but the clustering gave a useful overview of the topic.

An interesting observation is that often the strongest clustering occurs when a query contains a polysemous word. In these cases, the high level clusters correspond to the individual word senses (e.g., “lung” as a Chinese name vs. “lung” as a body part). The fact that the clustering separated the senses cleanly lead us to consider applying this technique to automatically generate senses for nouns (e.g., [7]) Investigation of this possibility suggested, unsurprisingly, that the clustering was only effective at distinguishing senses which were well represented in the corpus (the encyclopedia in this case which was unfortunately fairly limited in its coverage of some of the more interesting polysemous words).

4. DISCUSSION

An interface built around an interactive dendrogram has been developed for allowing a user to rapidly explore the structure of documents returned to an LSI query. Of course, the utility of this technique depends on several factors such as the needs and background of the user and the nature of the corpus and queries. Therefore, it seems clustering should not be the only technique available.

Clustering is likely to be most useful in large, rich corpora. One barrier to scaling the technique up to larger collections (apart from the computational costs) is that as of yet, no systematic rule has been found for determining the optimal number of articles to cluster given the corpus size. In particular, small sets of articles do not provide enough structure for clustering to be effective while large sets are costly and tend to blur important local structure. This issue would need to be addressed in later empirical studies.

ACKNOWLEDGMENTS

We thank Sue Dumais for assistance with the corpus and scaling. Pascal Obry is an employee of Electricité De France (EDF) who participated in this project as a visitor at Bellcore. Michael Littman is now at the Department of Computer Science, Brown University.

REFERENCES

1. *Academic American Encyclopedia*. Princeton: Arete Publishing Company, 1981. Electronic copy obtained from Grolier, Inc.

2. CREECH ET AL., M. L. Using hypertext in selecting reusable software components. In *Proceedings ACM Hypertext* (Dec. 1991), pp. 25–38.
3. CROUCH ET AL., D. B. The use of cluster hierarchies in hypertext information retrieval. In *Proceedings ACM Hypertext* (Nov. 1989), pp. 225–237.
4. DUMAIS, S. T., AND LANDAUER, T. K. Describing categories of objects for menu retrieval systems. *Behavior Research Methods, Instrumentation, and Computers* (1984), 242–248.
5. DUMAIS ET AL., S. T. Using latent semantic analysis to improve access to textual information. In *Proceedings ACM SICHI: Human Factors in Computing Systems* (May 1988), pp. 281–285.
6. EVERITT, B. *Cluster Analysis*. London: Heinemann Educational Books, 1977.
7. KROVETZ, R., AND CROFT, W. B. Lexical ambiguity in information retrieval. *ACM Transactions on Information Systems* 10 (1992), 115–141.
8. LIN ET AL., X. A self-organizing semantic map for information visualization. In *Proceedings of the ACM SIGIR Conference* (July 1991), pp. 262–269.
9. MAAREK ET AL., Y. S. An information retrieval approach for automatically constructing software libraries. *IEEE Transactions on Software Engineering T-SE* 16 (1991), 800–813.
10. RASMUSSEN ET AL., E. M. Automatic classification of chemical structure databases using a highly parallel array processor. *Journal of Computational Chemistry* (1988), 378–386.
11. STUETZLE, ET AL., W. Visualization of quantitative data. Tech. rep., Department of Statistics, U. of Washington, Seattle and Bellcore, 1990. A 27 min video tape.
12. VANRIJSBERGEN, C. J. *Information Retrieval*. London: Butterworths, 1979.