Enhancing Performance in Latent Semantic Indexing (LSI) Retrieval

Susan T. Dumais
Bellcore, 21236

ABSTRACT

We have previously described an extension of the vector retrieval method called "Latent Semantic Indexing" (LSI) (Deerwester, et al., 1990; Dumais, et al., 1988; Furnas, et al., 1988). The LSI approach partially overcomes the problem of variability in human word choice by automatically organizing objects into a "semantic" structure more appropriate for information retrieval. This is done by modeling the implicit higher-order structure in the association of terms with objects. Initial tests find this completely automatic method to be a promising way to improve users' access to many kinds of textual materials or to objects for which textual descriptions are available.

This paper describes some enhancements to the basic LSI method, including differential term weighting and relevance feedback. Appropriate term weighting improves performance by an average of 40%, and feedback based on 3 relevant documents improves performance by an average of 67%.

1. Introduction

We have developed a method called Latent Semantic Indexing (LSI) that can improve people's access to electronically available textual information (Deerwester, et al., 1990; Dumais, et al., 1988; Furnas, et al., 1988). Most approaches to retrieving textual materials depend on a lexical match between words in users' requests and those in database objects. Typically only text objects that contain one or more words in common with those in the users' query are returned as relevant. Word-based retrieval systems like this are, however, far from ideal - many objects relevant to a users' query are missed, and many unrelated or irrelevant materials are retrieved. We believe that fundamental characteristics of human verbal behavior underly these retrieval difficulties (Bates, 1986; Fidel, 1985; Furnas et al., 1987). Because of the tremendous diversity in the words people use to describe the same object or concept (*synonymy*), requesters will often use different words from the author or indexer of the information, and relevant materials will be missed. Conversely, since the same word often has more than one meaning (*polysemy*), irrelevant materials will be retrieved.

The LSI approach tries to overcome these problems by automatically organizing objects into a semantic structure more appropriate for information retrieval. This is done by modeling the implicit higher-order structure in the association of terms with objects. We assume that there is some underlying or "latent" structure in the pattern of word usage that is partially obscured by the variability of word choice. We use statistical techniques to estimate this latent structure and get rid of the obscuring "noise". A description of terms, objects and user queries based on the underlying latent semantic structure (rather than surface level word choice) is used for representing and retrieving information. LSI has several appealing properties for retrieval, both in a standard retrieval scenario, where a set of text objects is retrieved in response to a user's query, and in more sophisticated retrieval operations.

2. Overview of Latent Semantic Indexing

2.1 Theory.

The particular latent semantic indexing analysis that we have tried uses singular-value decomposition (SVD), a technique closely related to eigenvector decomposition and factor analysis (Forsythe, Malcolm and Moler, 1977; Cullum and Willoughby, 1985). We take a large matrix of term to text-object association data and decompose it into a set of, typically 50 to 150, orthogonal factors from which the original matrix can be approximated by linear combination. More formally, any rectangular matrix, X, for example a $t \times o$ matrix of terms and objects, can be decomposed into the product of three other matrices:

$$\underset{t\times o}{X} = \underset{t\times r}{T_0} \cdot \underset{r\times r}{S_0} \cdot \underset{r\times o}{O_0}' ,$$

such that T_0 and O_0 have orthonormal columns, S_0 is diagonal, and r is the rank of X. This is so-called *singular value decomposition* of X and it is unique up to certain row, column and sign permutations.

If only the k largest singular values of S_0 are kept along with their corresponding columns in the T_0 and O_0 matrices, and the rest deleted (yielding matrices S, T and O), the resulting matrix, $\hat{X}$, is the unique matrix of rank k that is closest in the least squares sense to X:

$$\underset{t\times o}{X} \approx \underset{t\times o}{\hat{X}} = \underset{t\times k}{T} \cdot \underset{k\times k}{S} \cdot \underset{k\times o}{O'} .$$

The idea is that the $\hat{X}$ matrix, by containing only the first k independent linear components of X, captures the major associational structure in the matrix and throws out noise. It is this reduced model, usually with k=100, that we use to approximate the term to text-object association data in X. Since the number of dimensions in the reduced model (k) is much smaller than the number of unique terms (t), minor differences in terminology are ignored. In this reduced model, the closeness of objects is determined by the overall pattern of term usage, so objects can be near each other regardless of the precise words that are used to describe them, and their description depends on a kind of consensus of their term meanings, thus dampening the effects of polysemy. In particular, this means that text objects which share no words with a user’s query may still be near it if that is consistent with the major patterns of word usage. We use the term "semantic" indexing to describe our method because the reduced SVD representation captures the major associative relationships between terms and text objects.

One can also interpret the analysis performed by SVD geometrically. The location of the terms and objects in k-space is given by the row vectors from the T and O matrices, respectively. In this space the cosine or dot product between vectors corresponds to their estimated similarity. Since both term and object vectors are represented in the same space, similarities between any combination of terms and objects can be easily obtained. Retrieval proceeds by using the terms in a query to identify a point in the space, and all text objects are then ranked by their similarity to the query. We make no attempt to interpret the underlying dimensions or factors, nor to rotate them to some intuitively meaningful orientation. Our analysis does not require us to be able to describe the factors verbally but merely to be able to represent terms, text objects and queries in a way that escapes the unreliability, ambiguity and redundancy of individual terms as descriptors.

This geometric analysis makes the relationship between LSI and the standard vector retrieval methods clear. In the standard vector model, terms are assumed to be independent, and form the orthogonal basis vectors of the vector space (Salton and McGill, 1983; van Rijsbergen, 1979). The independence assumption is quite unreasonable, but makes parameter estimation more tractable. Several researchers have suggested ways in which term dependencies might be added to the vector model (e.g. van Rijsbergen, 1977; Yu et al., 1983), but parameter estimation remains a severe difficulty especially when higher-order associations are included. In LSI, the reduced dimensional SVD gives us the orthogonal basis vectors, and term vectors are positioned in this space so as to reflect the correlations in their usage across documents. Since both terms and objects are represented in the same space, queries can be formed using any combination of terms and objects, and any combination of terms and objects can be returned in response to a query. Like the standard vector method, the LSI representation has many desirable properties: it does not require Boolean queries; it ranks retrieval output in decreasing order of document to query similarity; it easily accommodates term weighting; and it is easy to modify the location of individual vectors for use in dynamic retrieval environments.

The idea of aiding information retrieval by discovering latent proximity structure has several lines of precedence in the information science literature. Hierarchical classification analyses have sometimes been used for term and document clustering (Sparck Jones, 1971; Jardin and van Rijsbergen, 1971). Factor analysis has also been explored previously for automatic indexing and retrieval (Baker, 1962; Atherton and Borko, 1965; Borko and Bernick, 1963; Ossorio, 1966). Koll (1979) has discussed many of the same ideas we describe above regarding concept-based information retrieval, but his system lacks the formal mathematical underpinnings provided by the singular value decomposition model. Our latent structure method differs from these approaches in several important ways: (1) we use a high-dimensional representation which allows us to better represent a wide range of semantic relations; (2) both terms and text objects are explicitly represented in the same space; and (3) objects can be retrieved directly from query terms.

2.2 Practice.

We have applied LSI to several standard information science test collections, for which queries and relevance assessments were available. The text objects in these collections are bibliographic citations (consisting of the full text of document titles, authors and abstracts), or the full text of short articles. For each user query, all documents have been ordered by their estimated relevance to the query, making systematic comparison of alternative retrieval systems straightforward. Results were obtained for LSI and compared against published or computed results for other retrieval techniques, notably the vector method in SMART (Salton, 1968). Each document is indexed completely automatically, and each word occurring in more than one document and not on SMART's stop list is included in the LSI analysis. The LSI analysis begins with a large term by document matrix in which cell entries are a function of the frequency with which a given term occurred in a given document. A singular value decomposition (SVD) is performed on this matrix, and the k largest singular values and their corresponding left and right singular vectors are used for retrieval. Queries are automatically indexed using the same preprocessing as was used for indexing the original documents, and the query vector is placed at the weighted sum of its constituent term vectors. The cosine between the resulting query vector and each document vector is calculated. Documents are returned in decreasing order of cosine, and performance evaluated on the basis of this list.

Performance of information retrieval systems is often summarized in terms of two parameters - precision and recall. *Recall* is the proportion of all relevant documents in the collection that are retrieved by the system; and *precision* is the proportion of relevant documents in the set returned to the user. Average precision across several levels of recall can then be used as a summary measure of performance. For several information science test collections, we found that average precision using LSI ranged from comparable to to 30% better than that obtained using standard vector methods. (See Deerwester, et al., 1990; Dumais, et al., 1988; for details of these evaluations.) The LSI method performs best relative to standard vector methods when the queries and relevant documents do not share many words, and at high levels of recall.

Another test of the LSI method was done in connection with the Bellcore Advisor, an on-line service which provides information about Bellcore experts on topics entered by a user (Lochbaum and Streeter, 1988; Streeter and Lochbaum, 1987). The objects in this case were research groups which were characterized by abstracts of technical memoranda and descriptions of research projects written for administrative purposes. A singular value decomposition of a 728 (work group descriptions) by 7100 (term) matrix was performed and a 100-dimensional representation was used for retrieval. Evaluations were based on a new set of 263 technical abstracts not used in the original scaling, and descriptions of research interests solicited from 40 people. Each new abstract or research description served as a query and its similarity to each organization in the space was calculated. The result of interest is the rank of the correct organization in the returned list. For the research descriptions, LSI outperformed standard vector methods (median rank 2 vs. 4); for technical abstracts, performance was the same with the two methods (median rank 2). Interestingly, a combination of the two methods (maximum of normalized values on each) provided the best performance, returning the appropriate organization with a median rank of 1 for both kinds of test queries.

3. Improving Performance

There are several well-known techniques for improving performance in standard vector retrieval systems. One of the most important and robust methods involves differential *term weighting* (Sparck Jones, 1972). Another method of improvement involves an iterative retrieval process based on users’ judgments of relevant items - often referred to as *relevance feedback* (Salton, 1971). The LSI approach also involves choosing the *number of dimensions* for the reduced space. We do not present detailed results of other methods that sometimes improve retrieval performance like stemming and phrases since preliminary investigations showed at most modest (1%-5%) improvements with these methods. The effects of stemming were especially variable, sometimes resulting in small improvements, sometimes in small decrements in performance.

Table 1 gives a brief description and summarizes some characteristics of the datasets and queries used in our experiments. As noted above, the "documents" in these collections consisted of the full text of document abstracts or short articles.

MED: document abstracts in biomedicine received from the National Library of Medicine
CISI: document abstracts in library science and related areas published between 1969 and 1977 and extracted from Social Science Citation Index by the Institute for Scientific Information
CRAN: document abstracts in aeronautics and related areas originally used for tests at the Cranfield Institute of Technology in Bedford, England
TIME: articles from Time magazine’s world news section in 1963
ADI: small test collection of document abstracts from library science and related areas

	MED	**CISI**	**CRAN**	**TIME**	**ADI**
number of documents	1033	1460	1400	425	82
number of terms (occurring in more than one document)	5831	5743	4486	10337	374
number of queries	30	35	225	83	35
average number of documents relevant to a query	23	50	8	4	5
average number of terms per doc	50	45	56	190	16
average number of docs per term	9	13	16	8	4
average number of terms per query	10	8	9	8	5
percent non-zero entries	0.86	0.88	1.10	1.80	4.38

Table 1. Characteristics of Datasets

3.1 Choosing the number of dimensions.

Choosing the number of dimensions for the reduced dimensional representation is an interesting problem. Our choice thus far has been determined simply by what works best. We believe that the dimension reduction analysis removes much of the noise, but that keeping too few dimensions would loose important information. We suspect that the use of too few dimensions has been a deficiency of previous experiments that have employed techniques similar to SVD (Atherton and Borko, 1965; Borko and Bernick, 1963; Ossorio, 1966; Koll, 1979). Koll, for example, used only seven dimensions. We have evaluated retrieval performance using a range of dimensions. We expect the performance to increase only while the added dimensions continue to account for meaningful, as opposed to chance, co-occurrence. That LSI works well with a relatively small (compared to the number of unique terms) number of dimensions shows that these dimensions are, in fact, capturing a major portion of the meaningful structure.

Figure 1 shows performance for the MED database using different numbers of dimensions in the reduced LSI representation. Performance is average precision over recall levels of 0.25, 0.50 and 0.75.

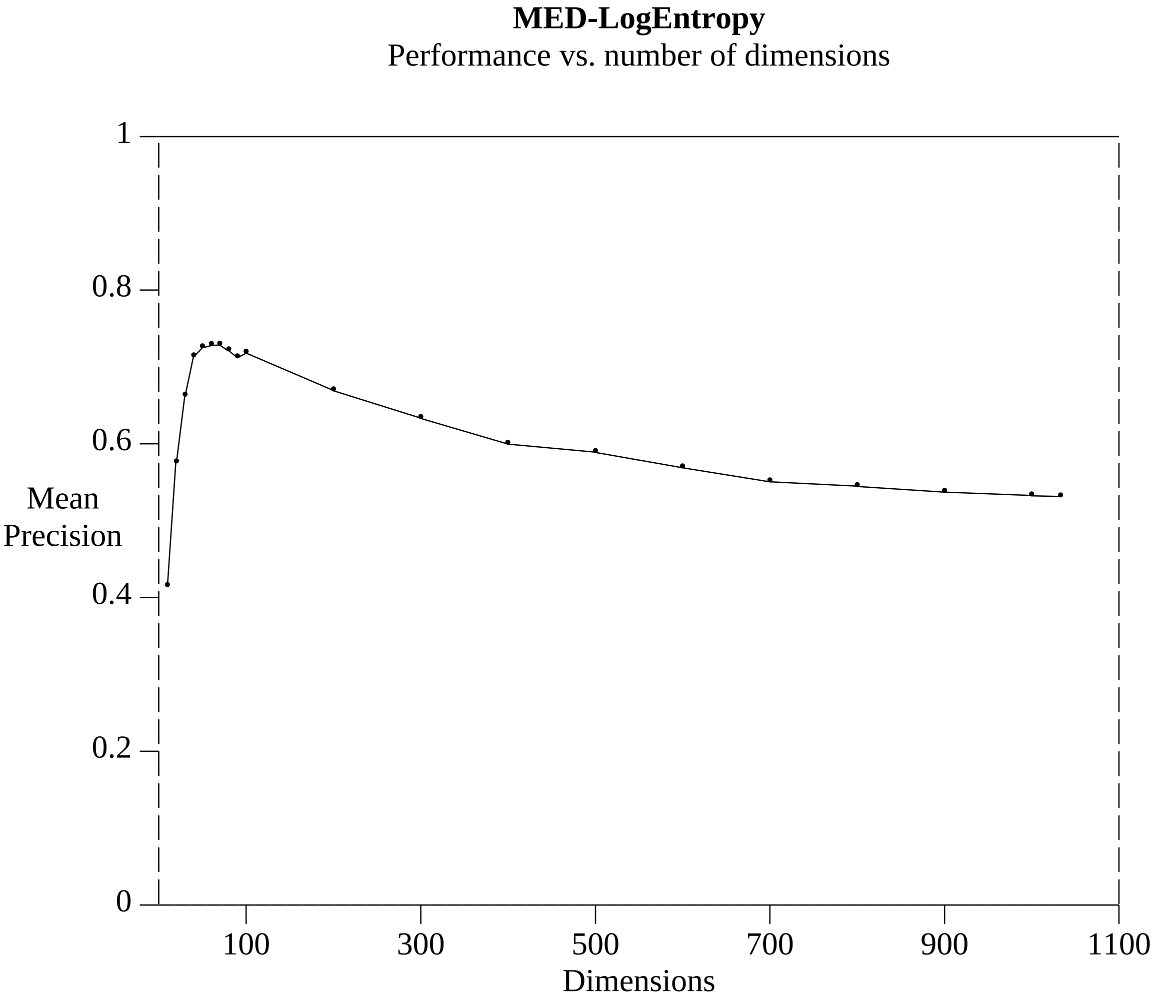

Figure 1. MED number of dimensions

It is clear from this figure that performance improves considerably after 10 or 20 dimensions, peaks between 70 and 100 dimensions, and then begins to diminish slowly. This pattern of performance (initial large increase and slow decrease to word-based performance) is observed with other datasets as well. As noted above, eventually performance must approach (typically by decreasing) the level of performance attained by standard vector methods, because with sufficient parameters SVD will exactly reconstruct the original term by document matrix.

We have found that 100-dimensional representations work well for these test collections. However, the number of dimensions needed to adequately capture the structure in other collections will probably depend on their breadth. Most of the test collections are relatively homogeneous, and 100 dimensions appears to be adequate to capture the major patterns of word usage across documents. In choosing the number of dimensions, we have been guided by the operational criterion of "what works best". That is, we examine performance for several different dimensions, and select the dimensionality that maximizes performance. In practice, the use of statistical heuristics for determining the dimensionality of an optimal representation will be important.

3.2 Term weighting.

One of the common and usually effective methods for improving retrieval performance in vector methods is to transform the raw frequency of occurrence of a term in a document (i.e. the value of a cell in the raw term-document matrix) by some function. Such transformations normally have two components. Each term is assigned a *global weight*, indicating its overall importance in the collection as an indexing term. The same global weighting is applied to an entire row (term) of the term-document matrix. It is also possible to transform the term's frequency in the document; such a transformation is called a *local weighting*, and is applied to each cell in the matrix. In the general case, the value for a term t in a document d is $L(t,d) \times G(t)$, where $L(t,d)$ is the local weighting for term t in document d and $G(t)$ is the term's global weighting.

Some popular **local weightings** include:

- Term Frequency
- Binary
- log(Term Frequency + 1)

Term Frequency is simply the frequency with which a given term appears in a given document. Binary weighting replaces any term frequency which is greater than or equal to 1 with 1. Log (Term Frequency + 1) takes the log of the raw term frequency, thus dampening effects of large differences in frequencies.

Four well-known **global weightings** are: Normal, GfIdf, Idf, and Entropy. Each is defined in terms of the term frequency (tf_{ij}), which is the frequency of term i in document j, the document frequency (df_i), which is the number of documents in which term i occurs, and the global frequency (gf_i), which is the total number of times term i occurs in the whole collection.

- Normal: $\frac{1}{\sqrt{\sum_j tf_{ij}^2}}$
- GfIdf: $\frac{gf_i}{df_i}$
- Idf: $\log_2\left(\frac{ndocs}{df_i}\right)+1$, where $ndocs$ is the number of documents in the collection
- 1 - Entropy or Noise: $1-\sum_j \frac{p_{ij}\, log(p_{ij})}{log(ndocs)}$ where $p_{ij}=\frac{tf_{ij}}{gf_i}$, and $ndocs$ is the number of documents in the collection

All of the global weighting schemes basically give less weight to terms that occur frequently or in many documents. The ways in which this is done involve interesting variations in the relative importance of local frequency, global frequency, and document frequency. Normal normalizes the length of each row (term) to 1. This has the effect of giving high weight to infrequent terms. Normal, however, depends only on the sum of the squared frequencies and not the distribution of those frequencies per se - e.g. a term that occurs once in each of 100 different documents is given the same weight as a term that occurs 10 times in only one document. GfIdf and Idf are closely related, both weighting terms inversely by the number of different documents in which they appear. GfIdf also increases the weight of frequently occurring terms. Neither method depends on the distribution of terms in documents, only the number of different documents in which a term occurs - e.g. a term occurring 10 times in each of two documents is given the same weight as a term occurring 1 time in one document and 19 times in a second document. Entropy is based on information theoretic ideas and is the most sophisticated weighting scheme. The average uncertainty or entropy of a term is given by $\sum_j \frac{p_{ij} \log(p_{ij})}{\log(ndocs)}$ where $p_{ij} = \frac{tf_{ij}}{gf_i}$. Subtracting this quantity from a constant assigns minimum weight to terms that are equally distributed over documents (i.e. where $p_{ij} = 1/ndocs$), and maximum weight to terms which are concentrated in a few documents. Entropy takes into account the distribution of terms over documents.

We explored the effects of six different term weighting schemes in each of the test collections. We performed analyses using: no global weighting (i.e. raw term frequency, tf_{ij}), combinations of the local weight tf_{ij} and each of the four global weights discussed above (GfIdf, Idf, Entropy, and Normal), and one combination of a local log weight (log(tf_{ij} + 1)) and a global entropy weight (LogEntropy). The original term by document matrix was transformed according to the relevant weighting scheme, and a reduced dimensional SVD was calculated and used for the analysis. Sixty dimensions were used for the ADI collection, and 100 dimensions were used for the remaining collections. In all cases, query vectors were composed using the same weight used to transform the original matrix.

Average precision (over the queries) at 10 levels of recall is shown in Figure 2 for the Cranfield test collection (CRAN). For this evaluation, we used a subset of CRAN, made up of the 924 documents that were relevant to at least one query, and a set of 100 test queries.

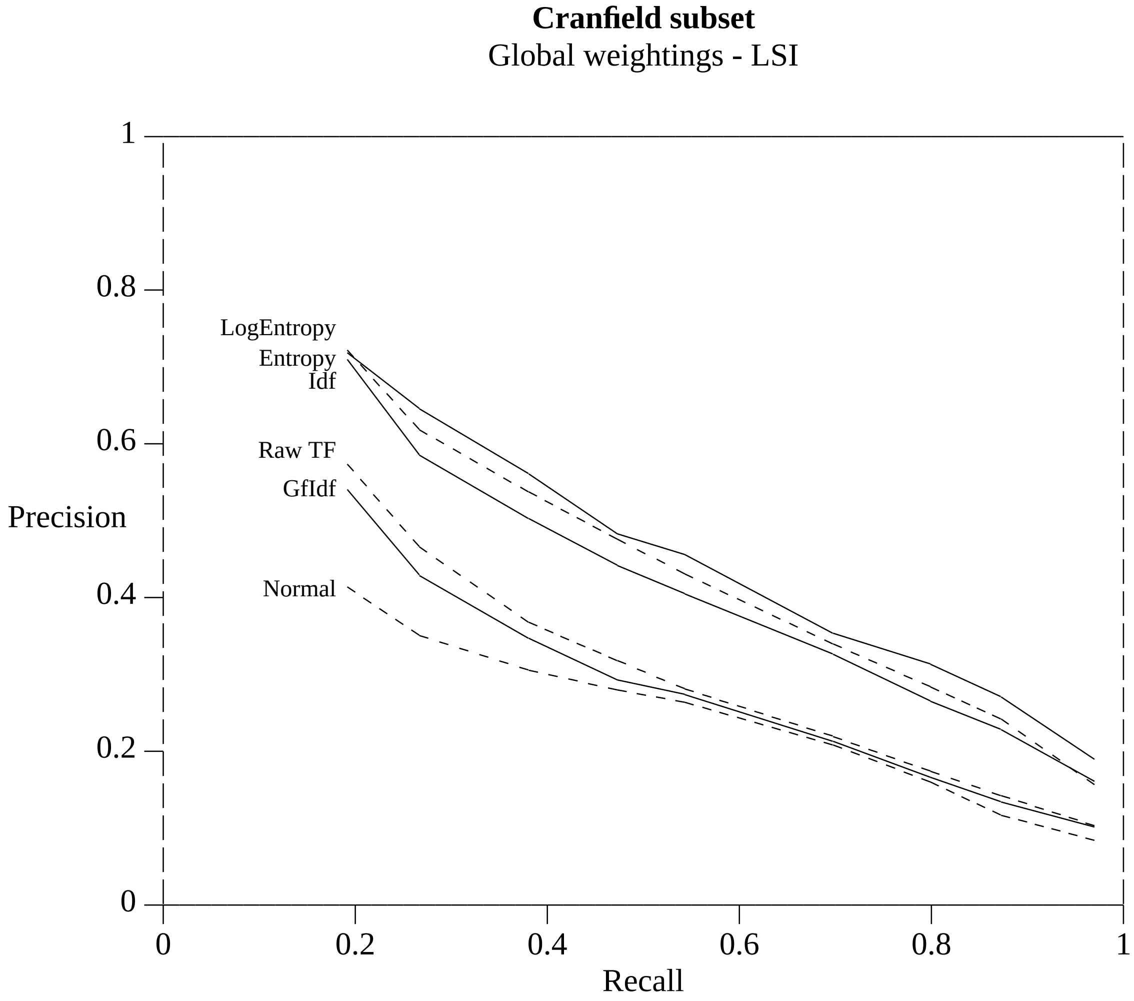

Figure 2. CRAN global term weightings

As can be seen, Normal and GfIdf are worse than no global weighting, and Idf, Entropy and LogEntropy result in substantial performance improvements. These results are generally confirmed on the other data sets as well. Table 2 presents a summary of term weighting experiments for five different datasets. The entries in the table are average precision over three levels of recall (.25, .50 and .75) and over all queries. The numbers in parenthesis indicate the percentage improvement over performance in the Raw Term Frequency case for that dataset.

	TF Raw	TF Normal	TF GfIdf	TF Idf	TF Entropy	Log(TF) Entropy
ADI	.28	.30 (+7%)	.24 (-14%)	.34 (+22%)	.36 (+30%)	.36 (+30%)
MED	.52	.48 (-6%)	.55 (+6%)	.67 (+30%)	.66 (+27%)	.72 (+39%)
CISI	.11	.10 (-6%)	.10 (-7%)	.15 (+36%)	.16 (+46%)	.17 (+53%)
CRAN	.29	.25 (-15%)	.28 (-5%)	.40 (+37%)	.40 (+38%)	.46 (+57%)
TIME	.49	.32 (-35%)	.42 (-14%)	.55 (+12%)	.54 (+10%)	.59 (+20%)
mean		-11%	-7%	+27%	+30%	+40%

Table 2. Summary term weighting

Normalization and GfIdf have mixed effects, depending on the dataset, but generally result in decreases in performance compared with raw term frequency. Idf, Entropy and LogEntropy all result in consistently large improvements in performance, with LogEntropy being somewhat better than the other two weightings.

These results are quite consistent with those reported by Harman (1986) using a standard vector retrieval method. In her study using the IRX (Information Retrieval Experiment) at the National Library of Medicine, Harman found that both Idf and Entropy weightings produced large performance advantages. Harmon found the following advantages over her baseline term overlap measure: Idf 20%; Entropy 24%; LogEntropy 35%; LogEntropy/normalized-document-lengths 44%. Given the differences in test document sets and the baseline method, the agreements in our results indicate that positive effects of the Idf and Entropy global weightings, and the local Log weighting are quite robust. We did not run any tests that involved normalizations of document length. We hope to encorporate this into our tools soon, but suspect that the differences will be relatively small for our test sets since documents within a set are roughly comparable in length (except for the TIME collection).

Somewhat surprisingly, term weighting experiments with the Bellcore Advisor which used an LSI representation give different results (Lochbaum and Streeter, 1988). When both the initial matrix transformation and query processing used the same weighting scheme, Lochbaum and Streeter found the best results for a Normalization weighting (median rank 3, 75th percentile rank 19), compared with (median rank 4, 75th percentile rank 23) for the raw matrix. Best performance was found when Normalization was used for the initial matrix transformation, and Entropy was used for query processing (median rank 2, 75th percentile rank 14). It is not clear how these differences can be explained. The Lochbaum and Streeter study differs from the others in an number of ways (e.g. performance measure, length of queries, and variations in document length, use of phrases) and some combination of these factors may be responsible.

3.3 Relevance feedback.

The idea behind relevance feedback is quite simple. Users are very unlikely to be able to specify their information needs adequately, especially given only one chance. With increases in computer speed, interactive or iterative searches are common, and users can reformulate queries in light of the system's response to previous queries (e.g. Oddy, 1977; Williams, 1984). Users claim to use information about the number of documents retrieved, related terms and information in relevant citations to reformulate their search requests. There is, however, little experimental evidence to back up this claim or to assess the success of such reformulations. Another approach to query reformulation is to have the system *automatically* alter the query based on user feedback about which documents are relevant to the initial request (Salton, 1971). This automatic system adaptation is what is usually meant by *relevance feedback*, and can be summarized as follows:

```
1. initial user query -> ranked list of documents
2. user selects some documents as relevant
      system automatically reformulates query using these docs ->
             new ranked list of docs
iterate step 2.
```

Simulations have shown that relevance feedback can be very effective (Ide and Salton, 1971; Rocchio, 1971; Ide, 1971; Salton, 1971; Stanfel, 1971; Stanfill and Kahle, 1986; Salton and Buckley, 1990). Systems can use information about which documents are relevant in many ways. Typically what is done is to increase the weight given to terms occurring in relevant documents and to decrease the weight of terms occurring in non-relevant documents. Most of our tests using LSI have involved a method in which the initial query is *replaced* with the vector sum of the documents the users has selected as relevant. We do not currently make use of negative information; for example, by moving the query away from documents which the user has indicated are irrelevant.

Table 3 on the next page shows an example of the feedback process using Query 34 from the CISI dataset. The top half of the table shows the system's initial response to the query. The first column indicates whether the document is relevant to the query or not; the second column is the document number; the third column is the cosine between the query and the document (in the 100-dimensional LSI solution); the fourth column is the rank of the document; and the fifth column is an abbreviated document title. We see that 4 of the top 15 documents are relevant - there are 38 documents relevant to this query. The precision for this query (averaged over recall levels .25, .50 and .75) is .09.

Next we show what happens when the first relevant document, document 673 in this case, replaces the initial query. This can be viewed as a simulation of a retrieval scenario in which a user looks down a list of candidate documents and stops when the first relevant document is encountered. The full text of this document abstract is then used as the query. Now 7 of the top 15 documents, including the first 4, are relevant. The precision for this query improves to .14; far from 1.00, but a noticeable improvement none the less. (Note that document 673 has similarity 1.00 with itself. When more than one document is used to form the new query, the similarities will not be 1.00, and there is no guarantee that the documents selected as relevant will be at the top of the revised list.)

Original Query: CISI, Query #34
Methods of coding used in computerized index systems.

Original Ranked List:

rel	docn	cos	rank	title
	531	0.598	1	Index Term Weighting
	146	0.592	2	Research on Users' Needs: Where is it Getting Us?
R	673	0.568	3	Rapid Structure Searches via Permuted Chemical ...
R	34	0.558	4	Keyword-In-Context Index for Technical Literature (KWIC Index)
R	53	0.554	5	The KWIC Index Concept: A Retrospective View
	688	0.549	6	The Multiterm Index: A New Concept in Info Retrieval
	149	0.523	7	Current Awareness Searches on CT, CBAS and ASCA
R	44	0.501	8	The Distribution of Term Usage in Manipulative Indexes
	86	0.494	9	A Core Nursing Library for Practitioners
	1283	0.481	10	Science Citation Index - A New Dimension in Indexing
	390	0.475	11	Factors Determining the Performance of Indexing Systems
	827	0.460	12	The Evaluation of Information Retrieval Systems
	194	0.451	13	World Biomedical Journals, 1951-60:
	790	0.447	14	Computer Indexing of Medical Articles - Project Medico
	180	0.446	15	Medical School Library Statistics

NEW Ranked List - Using Doc 673 as Query

rel	docn	cos	rank	title
R	673	1.000	1	Rapid Structure Searches via Permuted Chemical ...
R	669	0.623	2	Rapid Structure Searches via Permuted Chemical Line-Notations
R	53	0.601	3	The KWIC Index Concept: A Retrospective View
R	687	0.595	4	Index Chemicus Registry System: Pragmatic Approach to
	1026	0.584	5	Line - Formula Chemical Notation
	675	0.575	6	Atom-by-Atom Typewriter Input for Computerized
	688	0.568	7	The Multiterm Index: A New Concept in Info Retrieval
	715	0.532	8	Articulation in the Generation of Subject Indexes by computer
	1452	0.528	9	The Wiswesser Line-Formula Chemical Notation (WLN)
R	44	0.527	10	The Distribution of Term Usage in Manipulative Indexes
R	34	0.519	11	Keyword-In-Context Index for Technical Literature (KWIC Index)
	738	0.518	12	Substructure Searching of Computer-Readable Chemical Abstracts
	684	0.517	13	Operation of Du Pont's Central Patent Index
R	704	0.509	14	Use of the IUPAC Notation in Computer Processing
	86	0.509	15	A Core Nursing Library for Practitioners

Table 3. Relevance Feedback example

We have conducted several more systematic studies of relevance feedback. The document sets and user queries described in Table 1 were used in these experiments. We compared performance with the original query against two simulated cases of "feedback" and against a "centroid" query. Retrieval is first performed using the original query. Then this query is replaced by: the first relevant document, the weighted average or centroid of the first three relevant documents, or the centroid of all relevant documents. The "centroid" condition represents the best performance that can be realized with a single-point query. While this cannot be achieved in practice (except through many iterations), it serves as a useful upper bound on the performance that can be expected given the LSI representation.

Table 4 presents a summary of the relevance feedback simulation experiments. In all cases performance is the mean precision averaged over three levels of recall (.25, .50 and .75) and over all queries.

Orig: performance with original query
Nrel: average number of relevant documents for queries
Feedbk n=1: first relevant document is new query
Feedbk n=3: average of first three relevance documents is new query
Nview*n*: median number of documents viewed to find first *n* relevant ones
i.e. median rank of *n*th relevant document
Centroid: = average or centroid of all relevant documents is query

Dataset	Orig	Nrel	Feedbk n=1	Nview1	Feedbk n=3	Nview3	Centroid
ADI	.36	5	.59 (63%)	3	.87 (141%)	13	.98 (172%)
CISI	.16	50	.21 (31%)	2	.26 (62%)	9	.47 (193%)
MED	.67	23	.68 (1%)	1	.72 (7%)	3	.80 (19%)
CRAN	.42	8	.51 (20%)	1	.74 (76%)	6	.86 (95%)
TIME	.54	4	.80 (48%)	1	.85 (53%)	4	.85 (57%)
mean			(33%)	1	(67%)	7	(107%)

Table 4. Relevance Feedback summary

Large performance improvements are found for all but the MED dataset, where initial performance was already quite high. Performance improvements average 67% when the first three relevant documents are used as a query and 33% when the first document is used. These substantial performance improvements can be obtained with little cost to the user. A median of 7 documents have to be viewed in order to find the first three relevant documents, and a median of 1 document has to be seen in order to find the first relevant document. Relevance feedback using documents as queries imposes no added costs in terms of system computations. Since both terms and documents are represented as vectors in the same *k*-dimensional space, forming queries as combinations of terms (normal query), documents (relevance feedback) or both is straightforward.

The "centroid" query results in performance that is quite good in all but one collection. This suggests that the LSI representation is generally adequate to describe the interrelations among terms and documents, but that users have difficulty in stating their requests in a way that leads to appropriate query placement. In the case of the CISI collection (where there are an average of 50 relevant documents) a single query does not seem to capture the appropriate regions of relevance. We are now exploring a method for representing queries in a way that allows for multiple disjoint regions to be equally relevant to a query (Kane-Esrig et al., 1989). A related method for "query splitting" has also been described by (Borodin, Kerr and Lewis, 1968).

The LSI relevance feedback results are comparable to those reported by Salton and his colleagues using the vector method (Rocchio, 1971; Ide, 1971; Salton, 1971). Even though they used somewhat different datasets and the details of their feedback method was quite different, they too found large (average 60%) and reliable improvements in performance with feedback. Their feedback method involved finding all relevant documents in the top 15, and modifying the query by adding terms from relevant documents and subtracting terms from non-relevant documents. If there are on average 3 relevant documents in the top 15 then these results are very similar to ours. Stanfill and Kahle (1986) also report large improvements with feedback, but their results involve only two queries and a small subset of documents. We have also performed a few more direct comparisons using a standard word-based vector method on our datasets. Performance increases were comparable to those found with the reduced dimensional LSI representation.

The simulation experiments described above examined performance improvements obtained using a single feedback cycle. We are currently exploring what happens when several retrieval iterations are used. For example, how do three sequential one-document feedback queries compare with the three-document feedback query described above?

It should be noted that the extent of the effectiveness of relevance feedback is tricky to gauge. There will be some improvement in performance simply because the documents used as the feedback query will typically move higher in the ranked retrieval list. It is difficult to separate the contribution to increased retrieval effectiveness produced when these documents move to higher ranks from the contributions produced when the rank of previous unselected relevant documents increases. (It is probably the latter which users care about.) Methods for dealing with this include: freezing documents identified as relevant in their original ranks, or splitting the document collection into halves with similar relevance properties, but we have not encorporated such corrections into our calculations. However, even if the magnitude of the improvement is somewhat smaller than noted above, there is room for significant improvement as can be seen by the "centroid" results and the relevance feedback methods are a step in this direction.

How well relevance feedback (or other iterative methods) will work in practice is an empirical issue. We are now in the process of conducting an experiment in which users modify initial queries by: rephrasing the original query; relevance feedback based on user-selected relevant documents; or a combination of both methods. (See Dumais, Littman and Arrowsmith, 1989 for a description of the interface and evaluation method.)

4. Conclusions

LSI is a modification of the vector retrieval method that explicitly models the correlation of term usage across documents using a reduced dimensional SVD. The technique's tested performance ranged from roughly comparable to 30% better than standard vector methods, apparently depending on the associative properties of the document set and the quality of the queries. These results demonstrate that there is useful information in the correlation of terms across of documents, contrary to the assumptions behind many vector-model information retrieval approaches which treat terms as uncorrelated.

Performance in LSI-based retrieval can be improved by many of the same techniques that have been useful in standard vector retrieval methods. In addition, varying the *number of dimensions* in the reduced space influences performance. Performance increases dramatically over the first 100 dimensions, reaching a maximum and falling off slowly to reach the typically lower word-based level of performance. Idf and Entropy global *term weighting* improved performance by an average of 30%, and improvements with the combination of a local Log and a global Entropy weighting (LogEntropy) were 40%. In simulation experiments, *relevance feedback* using the first 3 relevant documents improved performance by an average of 67%, and feedback using only the first relevant document improved performance by an average of 33%. Since the first three relevant document are found after searching through only a median of 7 documents, this method offers the possibility of dramatic performance improvements with relatively little user effort. An experiment is now underway to evaluate the feedback method in practice, and to compare it with other methods of query reformulation.

REFERENCES

1. Atherton, P. and Borko, H. A test of factor-analytically derived automated classification methods. AIP rept AIP-DRP 65-1, Jan. 1965.

2. Baker, F.B. Information retrieval based on latent class analysis. *Journal of the ACM*, 1962, *9*, 512-521.

3. Bates, M.J. Subject access in online catalogs: A design model. *JASIS*, 1986, *37 (6)*, 357-376.

4. Borko, H and Bernick, M.D. Automatic document classification. *Journal of the ACM,* April 1963, *10(3)*, 151-162.

5. Borodin, A., Kerr, L., and Lewis, F. Query splitting in relevance feedback systems. *Scientific Report No. ISR-14,* Department of Computer Science, Cornell University, Oct. 1968.

6. Cullum, J.K. and Willoughby, R.A. *Lanczos algorithms for large symmetric eigenvalue computations - Vol 1 Theory* (Chapter 5: Real rectangular matrices). Brikhauser, Boston, 1985.

7. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman R. A. Indexing by latent semantic analysis. *JASIS,* 1990, *41(6)*, 391-407.

8. Dumais, S. T., Furnas, G. W., Landauer, T. K., and Deerwester, S.. Using latent semantic analysis to improve information retrieval. In *CHI'88 Proceedings*, pp. 281-285.

9. Dumais, S.T., Littman, M.L. and Arrowsmith, E. InfoSearch: A program for iterative information retrieval using LSI. Bellcore TM, 1989.

10. Fidel, R. Individual variability in online searching behavior. In C.A. Parkhurst (Ed.). *ASIS'85: Proceedings of the ASIS 48th Annual Meeting, Vol. 22*, October 20-24, 1985, 69-72.

11. Forsythe, G.E., Malcolm, M.A., and Moler, C.B. *Computer Methods for Mathematical Computations* (Chapter 9: Least squares and the singular value decomposition). Englewood Cliffs, NJ: Prentice Hall, 1977.

12. Furnas, G. W., Deerwester, S., Dumais, S. T., Landauer, T. K., and Harshman, R. A. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proceedings of SIGIR*, 1988, 36-40.

13. Furnas, G.W., Landauer, T.K., Gomez, L.M., and Dumais, S.T. The vocabulary problem in human-system communication. *Communications of the ACM*, 1987, *30(11)*, 964-971.

15. Harman, D. An experimental study of factors important in document ranking. In *Proceedings of ACM SIGIR*, Pisa, Italy, 1986.

15. Ide, E. New experiments in relevance feedback. Chapter 16. In G. Salton (Ed.), *The SMART retrieval system: Experiments in automatic document processing,* Prentice Hall, 1971.

15. Ide, E. and Salton, G. Interactive search strategies and dynamic file organization in information retrieval. In G. Salton (Ed.), *The SMART retrieval system: Experiments in automatic document processing*, Prentice Hall, 1971.

16. Jardin, N. and van Rijsbergen, C.J. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 1971, *7*, 217-240.

17. Kane-Esrig, Y., Casella, G., Streeter, L. A., and Dumais, S. T. Ranking documents for retrieval by Bayesian modeling of a relevance distribution. In *Proceedings of the 12th IRIS (Information System Research In Scandinavia) Conference*, Aug 1989.

18. Koll, M. An approach to concept-based information retrieval. *ACM SIGIR Forum, XIII32-50*, 1979.

20. Lochbaum, K. E. and Streeter, L. A. Comparing and combining the effectiveness of Latent Semantic Indexing and the ordinary Vector Space Model for information retrieval. Bellcore TM-ARH-012395, August 1988.

21. Oddy, R. N. Information retrieval through man-machine dialogue. *Journal of Documentation*, 1977, *33*, 1-14.

22. Ossorio, P.G. Classification space: A multivariate procedure for automatic document indexing and retrieval. *Multivariate Behavioral Research*, October 1966, 479-524.

24. Rocchio, J.J.Jr. Relevance feedback in information retrieval. Chapter 14. In G. Salton (Ed.), *The SMART retrieval system: Experiments in automatic document processing*. Prentice Hall, 1971.

25. Salton, G. *Automatic Information Organization and Retrieval*. McGraw Hill, 1968.

26. Salton G. (Ed.), *The SMART retrieval system: Experiments in automatic document processing.* Prentice Hall, 1971.

27. Salton, G. Relevance feedback and the optimization of retrieval effectiveness. Chapter 15. In G. Salton (Ed.), *The SMART retrieval system: Experiments in automatic document processing.* Prentice Hall, 1971.

28. Salton, G. and McGill, M.J. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

29. Salton, G. and Buckley, C. Improving retrieval performance by relevance feedback. *JASIS*, 1990, *41(4)*, 288-297.

30. Sparck Jones, K. *Automatic Keyword Classification for Information Retrieval,* Buttersworth, London, 1971.

31. Sparck Jones, K. A statistical interpretation of term specificity and its applications in retrieval. *Journal of Documentation*, 1972, *28(1)*, 11-21.

32. Stanfel, L. E. Sequential adaptation of retrieval systems based on user inputs. *Information Storage and Retrieval*, 1971, *7*, 69-78.

33. Stanfill, C. and Kahle, B. Parallel free-text search on the connection machine system. *Communications of the ACM*, 1986, *29(12)*, 1229-1239.

34. Streeter, L. A. and Lochbaum, K. E. An expert/expert-locating system based on automatic representation of semantic structure. In *Proceedings of the fourth conference on artificial intelligence applications*, March 14-18, 1987, San Diego, CA, 345-350.

35. van Rijsbergen, C.J. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, 1977, *33(2)*, 106-119.

36. van Rijsbergen, C.J. *Information retrieval.*, Buttersworth, London, 1979.

37. Williams, M. D. What make RABBIT run? *International Journal of Man-Machine Studies*, 1984, *21*, 333-352.

38. Yu, C. T., Buckley, C., Lam, K., Salton, G. A generalized term dependence model in information retrieval. *Information Research and Technology*, 1983, *2*, 129-154.