# Automatic Query Formulations in Information Retrieval

G. Salton, C. Buckley and E.A. Fox

October 1982

TR 82-524

Department of Computer Science
Cornell University
Ithaca, New York 14853

# Automatic Query Formulations in Information Retrieval

G. Salton, C. Buckley and E.A. Fox*

## Abstract

Modern information retrieval systems are designed to supply relevant information in response to requests received from the user population. In most retrieval environments the search requests consist of keywords, or index terms, interrelated by appropriate Boolean operators. Since it is difficult for untrained users to generate effective Boolean search requests, trained search intermediaries are normally used to translate original statements of user need into useful Boolean search formulations.

Methods are introduced in this study which reduce the role of the search intermediaries by making it possible to generate Boolean search formulations completely automatically from natural language statements provided by the system patrons. Frequency considerations are used automatically to generate appropriate term combinations as well as Boolean connectives relating the terms. Methods are covered to produce automatic query formulations both in a standard Boolean logic system, as well as in an extended Boolean system in which the strict interpretation of the connectives is relaxed. Experimental results are supplied to evaluate the effectiveness of the automatic query formulation process, and methods are described for applying the automatic query formulation process in practice.

---

*Department of Computer Science, Cornell University, Ithaca, New York 14853.

This study was supported in part by the National Science Foundation under Grant No. IST-80-17589.

## 1. The Conventional Retrieval Environment

In operational information retrieval, Boolean query formulations are used to express the customers' information needs. The standard rules of Boolean logic may not, however, provide an ideal environment for the formulation of effective search requests. For one thing, the conventional Boolean logic assigns a strict interpretation to the connectives and, or, and not, which produces unwanted retrieval results in certain circumstances. Thus, in response to an or-query such as "A or B or ... or Z", a document containing all of the terms specified in the query will not be rated more highly than a document containing only one of the query terms. Analogously, in response to an and-query such as "A and B and ... and Z", a document containing all but one of the query terms is deemed to be just as useless as a document containing none of the query terms.

Such a situation would not necessarily be injurious if the average retrieval system user were familiar with the rules of Boolean logic and could evaluate the occurrence characteristics of the query terms in the document collections under consideration. In that case, it would be possible to assess the likelihood of co-occurrence of two or more terms in the documents of a collection and to reach a decision whether such terms ought to be treated as synonymous by including them in an or-clause, or as members of a phrase by using an and connective to relate them. Unfortunately, there exists much evidence to show that ordinary users are unable to master the complications of Boolean logic, and even professional indexers and searchers find it difficult to construct consistently effective index representations and search statements. [1,2,3]

One possible solution consists in giving up the Boolean logic entirely

of previously retrieved items judged to be relevant to a given query could be used to reformulate or improve a given available Boolean query statement. This problem is considered in the remainder of this study.

## 2. Automatic Formulation of Conventional Boolean Queries

The basic problem in Boolean query formulation consists in first choosing an appropriate set of query terms, and in then using the Boolean operators to generate a formulation which is not so broad as to retrieve an unreasonable amount of extraneous matter thereby causing a loss in search precision, nor so narrow as to reject a large number of relevant items thereby causing a loss in search recall. A term that is considered to be too broad can be rendered more specific by including it in an and-clause, such as (A and B); analogously, a term considered to be too specific can be broadened by using it in an or-clause together with other related, specific terms.

These considerations suggest that it may be possible to generate automatic Boolean query formulation in "disjunctive normal form" by using Boolean or connectives to relate clauses of terms consisting of single terms of the correct specificity, or anded combinations of broader terms that need to be rendered more specific. A typical query could then be formulated as

$$S_i \text{ or } S_j \text{ or} \ldots . \text{or } (P_{k_1} \text{ and } P_{k_2}) \text{ or} (P_{m_1} \text{ and } P_{m_2}) \text{ or} \ldots .$$

$$\text{or } (T_{n_1} \text{ and } T_{n_2} \text{ and } T_{n_3}) \text{ or } (T_{p_1} \text{ and } T_{p_2} \text{ and } T_{p_3}) \text{ or} \ldots$$

When $S_i$ designates a single term, $P_{k_i}$ is a component of a term pair and $T_{n_i}$ is a component of a term triple. To determine the productive terms or term combinations, it is convenient to relate the specificity of a term, or term combination, to the frequency with which the term is assigned to the documents of

document frequency for a term combination consists in first taking the document frequency of each single term, that is, the postings frequency $n_i$ for each term i. If one assumes that the terms occur independently of one another in the N documents of a collection, the estimated document frequency of a pair of terms ($T_i$ and $T_j$) may be specified as

$$n_{ij} = \frac{n_i \cdot n_j}{N} \tag{1}$$

For an anded triple such as ($T_i$ and $T_j$ and $T_k$) one similarly obtains

$$n_{ijk} = \frac{n_i \cdot n_j \cdot n_k}{N^2}. \tag{2}$$

In general for a conjunction of p terms ($T_1$ and $T_2$ and ...and $T_p$) the estimated document frequency in a collection of N documents is set at $(n_1 \cdot n_2 \cdot \cdot \cdot \cdot \cdot n_p)/N^{p-1}$.

Earlier work in the theory of term importance has shown that in the absence of specific information about the actual importance of query or document terms, the inverse document frequency (idf) represents a useful indication of term importance. [6,7] This implies that the usefulness of a term, or a term combination may be determined as an inverse function of the corresponding document frequency. The idf weight of a term, or term combination, of document frequency $n_i$ may conveniently be computed as

$$w_i = \log \frac{N}{n_i} \quad \text{or} \quad w_i = 1 - \frac{n_i}{N.} \tag{3}$$

The document frequency computations ((1)(2)(3)) for terms and term combinations provide a simple way of determining the order in which the terms, or term combinations are incorporated into a given query formulation: the best terms--those with the highest inverse document frequency weight should be used

order and subtracting the corresponding term pairs subsumed by the added singles. (For example, if term A is added to a query formulation, the pairs A and B, A and C, A and D, etc. become redundant.) Following each addition and subtraction operation, the estimated number of retrieved items must be recomputed.

c) If on the other hand, the estimated number of retrieved items is larger than the preestablished retrieval threshold, it becomes necessary to generate a narrower query formulation. This is done by successively eliminating the single terms in increasing idf weight order while adding the missing term pairs which include the deleted singles as components. If the estimated number of retrieved items is still too large, the term pairs are eventually eliminated in increasing idf weight order and the missing term triples are added. The deletion and addition operations stop when the desired number of retrieved items is eventually obtained.

A summary of the automatic Boolean query construction process appears in Table 1.

The automatic query construction method may be illustrated by using as an example a query submitted to the Medlars search system which is operated by the National Library of Medicine in Washington. The original natural language query is listed at the top of Table 2 together with the Boolean search statement generated manually by a trained Medlars searcher. The analyzed natural language statement is shown at the bottom of Table 2. Following deletion of common words and suffix removal, the original statement is reduced to seven terms. The term "pyrophosphate" could not be found in the stored automatic term dictionary and does therefore not appear in the analyzed term list. The term "effect" which does appear is deleted because of the excessive number of

(parathyroid) which is replaced by five pairs containing the Pa term; this reduces the expected number of retrieved items to 50.6. The corresponding query statement is shown in step 3 of Table 4. Since no further single terms remain, it is necessary next to remove pairs of terms in order to obtain narrower query statements. In steps 4 and 5 of Table 4 the term pairs with highest estimated document frequency (lowest idf weight) are successively removed to reduce the estimated number of retrieved items to 38.4. In step 6 one additional term pairs (Ur and Ki) is removed, but since the three pairs (Ki and Ho), (Ur and Ho) and (Ur and Ki) are now absent, the triple (Ur and Ki and Ho) is no longer redundant. This triple is therefore added in step 6 following the removal of (Ur and Ki). Successive deletions of three more term pairs followed the addition of nonredundant term triples finally produce a query statement consisting of the 9 pairs and 4 triples shown at the bottom of Table 4. This query may be expected to retrieve 22 items which is close enough to the stated number of wanted items. It is clear that all addition and deletion operations are controlled only by the estimated document frequencies and idf weights of the terms, and can therefore be carried out automatically without difficulty.

The automatic Boolean query generation process using term singles, pairs, and triples in disjunctive normal form may be evaluated by comparing the retrieval results obtained for the automatically formulated queries with the results obtained with manually formulated queries. To evaluate the effectiveness of a retrieval system it is customary to compute values of the search recall and search precision following the retrieval of some fixed number of documents. The recall represents the proportion of relevant items retrieved out of the total number of relevant in the whole collection, and the precision represents the proportion of relevant items retrieved out of the total number

queries at recall levels exceeding 0.2. The recall-precision curve for the automatic query formulation process is correspondingly closer to the upper right-hand corner of the graph where both recall and precision values are equal to 1. For the Medlars collection the manual query formulations were evidently not optimal, and the larger number of anded term pairs and triples included in the automatic query forms produced better output than the original manual queries.

It is awkward to compare complete tables or graphs with each other when many comparisons are performed. In the remainder of this study, the evaluation results are therefore stated in terms of a single precision value, representing the average precision obtained at three typical recall levels, including a low recall level of 0.25, a medium recall of 0.50, and a high recall of 0.75. Percentage improvement or deterioration values may then be given for each of these composite precision measures. A complete recall-precision comparison using the composite precision values is shown in Table 5 for the Medlars 1033 collection. The base run listed at the top of Table 5 is the conventional retrieval output obtained with manually formulated Boolean queries. A secondary base run also included in Table 5 is the vector processing run using a cosine coefficient to compare query and document vectors composed of weighted terms automatically derived from natural language statements. [4,5] For the Medlars collection under consideration, the vector processing system using term weights and a large number of natural language terms proves superior to the standard Boolean query system.

The output for the automatic Boolean query generation process is included in the center section of Table 5. Various cases are shown for singles, pairs and triples, including retrieval thresholds of 20, 30, 50 and 100 documents.

computer science, known as Inspec 12684. It may be noted that in this case, the conventional Boolean search with manually prepared queries is closer to the optimal retrospective run than it was for the Medlars collection previously used. This indicates that the conventional Boolean queries were more carefully constructed for Inspec than for Medlars. Correspondingly, the automatically built Boolean queries using singles, pairs and triples do not now perform as well as the conventional manual queries. Evidently, the natural language statements which were used as a basis for the automatic query construction method were not well suited to the heterogeneous Inspec collection. In that case one cannot expect to construct good Boolean statements without using a substantial amount of intellect which is of course lacking in the query construction method under discussion.

The results of Tables 5 and 6 show that when good natural language statements of user need are available which match the language of the document collection, the simple automatic query construction process outlined in Table 1 is competitive with conventional manual Boolean query formulations. Indeed for the Medlars collection, the automatic Boolean queries proves superior to the manual ones. When the natural language query formulations do not match the document terms additional procedures must be incorporated in the automatic query formulation process. This possibility is further considered in the remainder of this study.

## 3. The Application of Extended Boolean Logic

The simple query construction process described in the previous section is surprisingly powerful, given that the anded term pairs and term triples which are used as a basis are constructed by purely formal, as opposed to semantic, criteria. The results obtained with the Inspec collection (Table 6)

the strict rules of Boolean logic. Thus when an and connective is used to relate two phrase components, the respective terms are treated as essential regardless of semantic appropriateness. Similar considerations apply to terms related by or connectives.

Since the formal query construction process creates some questionable phrases, improved retrieval results may be obtainable by using a fuzzy type of query document matching in which the interpretation of the Boolean connectives is less unforgiving than in the conventional Boolean logic. Such as extended Boolean system has recently been proposed, using an auxiliary parameter p, $1 \leq p \leq \infty$, attached to the connectives and and or to control the query-document matching process, and providing thereby a variety of more or less stringent interpretations for the query statements. [11-12]

Consider, as an example, a document D with assigned terms A and B and let $d_A$ and $d_B$ represent the weights or importance of the two terms in the document, $0 \leq d_A, d_B \leq 1$. A term weight of 0 indicates that the corresponding term is not assigned to an item; a weight of 1 represent a fully weighted term, and weights between 0 and 1 are partial term assignments. Given queries (A and B) and (A or B), it is possible to define the following query-document similarity functions between these queries and a document $D = (d_A, d_B)$.

$$sim(Q_{(A \text{ and } B)}, D) = 1 - \left[ \frac{(1-d_A)^p + (1-d_B)^p}{2} \right]^{1/p} \tag{4a}$$

$$sim(Q_{(A \text{ or } B)}, D) = \left[ \frac{d_A^p + d_B^p}{2} \right]^{1/p} \tag{4b}$$

It is easy to verify that when p is large, the similarity function (4a) produces a value of 1 whenever $d_A = d_B = 1$, and values of 0 whenever one or both of the term weights $d_A$ or $d_B$ is equal to 0. In the same way, the func-

similarity value of 1 with respect to queries (A and B) and (A or B). When $d_A = d_B = 0$ one again obtains a query-document similarity of 0. When one of the two document terms has a value of 1 and the other a value of 0, retrieval values of $1-1/\sqrt{2}$ are obtained with respect to query (A and B), and $1/\sqrt{2}$ with respect to (A or B). In other words, intermediate values between 1 and 0 are obtained when some of the document terms match the query terms and others do not. [11-12]

The extended Boolean retrieval system is easily applied to the case where term weights are also attached to the query terms in accordance with the presumed importance of each term in the query. Such query weights can then be used in addition to the weights attached to the document terms. In that case the queries may be formulated as [(A,a) or (B,b)] and [(A,a) and (B,b)] respectively, with a and b, $0 \leq a, b \leq 1$, representing the weights of terms A and B in the query. The similarity functions with document $D = (d_A, d_B)$ now become

$$sim\,(Q_{[(A,a)\ \underline{and}\ (B,b)]}, D) = 1 - \left[\frac{a^p(1-d_A)^p + b^p(1-d_B)^p}{a^p + b^p}\right]^{1/p} \qquad (5a)$$

and

$$sim\,(Q_{[(A,a)\ \underline{or}\ (B,b)]}, D) = \left[\frac{a^p d_A^p + b^p d_B^p}{a^p + b^p}\right]^{1/p} \qquad (5b)$$

Once again when the query and document term weights are binary (restricted to 0 and 1), a standard Boolean retrieval system is obtained for large p values ($p=\infty$). A standard vector processing system which does not distinguish between and and or connectives is obtained for p=1, and intermediate systems with loose phrase and synonym interpretations are obtained for intermediate p values. The interpretation of the p-value variations is summarized in the

chosen as the averages of the weights of the terms in the respective clauses. For the normal Boolean formulation of Table 8(b), the clause weights are of course also equal to 1. When the matching functions of expressions 4 and 5 are used with the query formulation of Table 8(b), the results are identical with those obtained by a conventional Boolean retrieval system using the formulation of Table 8(a).

The relaxed p-norm formulation for the sample query is shown in Table 8(c). The outer and and inner or operators are assigned p-values designated as $p_1$ and $p_2$ respectively. Each term is weighted using the automatically determined inverse document frequency (idf) value computed as $\log N/n_i$ (see expression (3)). The clause weight is set as the average of the idf values of the component terms. The last part of Table 9 contains a standard vector formulation for the same query, consisting of a set of weighted terms used without Boolean operators.

The operations of the extended Boolean retrieval system are evaluated in the next section.

## 4. Automatic Query Formulation in the Extended System Using Single Terms, Term Pairs and Term Triples

The document collections already used earlier to evaluate the automatic query formulations in the conventional Boolean system are used again to carry out experiments in the extended model. The experimental results cover retrieval thresholds which vary between 20 and 500 documents, respectively; p-values equal to 1, 2, 5, and infinity are used in the experiments. A common p-value was assigned to all Boolean query operators within a given experimental run.

preted in the strict conventional way.

For the Medlars collection the automatic p-norm queries are comparable in effectiveness also with the automatically constructed vector queries used in a vector processing system. The advantage of the p-norm queries compared with the vector queries is the compatibility with the conventional Boolean processing system which is lacking in the vector processing environment. For the Inspec collection, the automatic query formulations produce results which are 20 to 30 percent better than the manual Boolean queries, but 30 to 40 percent worse than the vector processing queries. The automatic queries in the extended system appear good enough to be used as initial queries to be improved by user-system interaction for subsequent searches.

The experiments of Tables 9 and 10 lead to a practical method for the construction of effective quasi-Boolean queries:

1) conventional Boolean queries are first constructed in disjunctive normal form, using single terms, as well as anded term pairs and term triples;

2) p-values are assigned to the Boolean connectives; if no special information is available about the importance of particular terms, uniformly low values of p=1 or p=2 are preferred;

3) the query terms are weighted using actual inverse document frequency values for single terms and estimated idf values for term pairs and triples; the clause weights are determined as the average idf values of the component terms;

4) the matching functions of expressions (5) are used to compare queries and documents, and the documents are brought to the users' attention in

a) very low document frequency terms exhibit low discrimination values;

b) as the document frequency of a term rises the discrimination value of the term improves up to a maximum point reached for medium frequency terms;

c) as the document frequency increases still further the discrimination value of a term worsens rapidly;

d) very high frequency terms exhibit the worst discrimination values.

The term discrimination analysis shows that very low and very high frequency terms do not produce satisfactory content identifiers. This suggests that frequency transformation operators be used to lower the frequency of terms on the high side of frequency spectrum, while increasing the frequency of the terms on the low side of the spectrum. A broad high-frequency term receives a narrower scope and hence a lower assignment frequency when it is incorporated into a term phrase; for example, instead of using "computer" one could use "computer science", "computer theory", or "computer program". Analogously, a narrow low frequency term receives a broader interpretation by using it together with other low-frequency synonyms; instead of using the single identifier "microcomputer", one would say "microcomputer or minicomputer or hand-held calculator".

The term discrimination theory can be used as a basis for a new query formulation system which is compatible with the earlier system based on single terms, term pairs and term triples, but also provides special treatment for the low frequency terms in addition to the high frequency terms:

a) low frequency terms should be broadened by incorporating several related low-frequency terms into a single or-clause, that is, a clause related by

instances, and the outer operator ($\underline{or}(p_0)$) can be replaced by $\underline{and}(p_0)$.

It remains to specify a procedure for choosing an appropriate number of frequency ranges and for assigning the p-values to the outer and inner Boolean operators. Since the terms falling into the more extreme frequency ranges (either very low or very high frequency groups) are most in need of being related to other terms, the corresponding p-values should be interpreted more strictly than the p-values for the less extreme frequency ranges. For the example previously used a value of 2 might be used for $p_1$ and $p_4$, covering the very low and very high frequency terms; correspondingly smaller values of 1.5 or 1.2 should be appropriate for $p_2$ and $p_3$. Finally a very low value of 1 or 1.1 might be suitable for the outer operator $p_0$. When the outer operator $p_0$ equals 1, the results are of course identical for and and or operators.

The frequency ranges can be chosen automatically in such a way that the number of terms in each range is approximately the same. Consider as an example the distribution of document frequencies of Table 11 for the terms used with the Medlars and Inspec collections. (To reduce the term set used for indexing to a manageable size, terms of document frequency 1 have been excluded from consideration for the Inspec collection.) Over 53 percent of the terms occur with frequency 1 in the Medlars collection; another 12 percent occur in two documents, and about seven percent in three documents. It is not possible to break down the large number of very low frequency terms into several frequency classes. However, the higher frequency terms can be grouped in such a way that each class covers approximately five percent of the terms for Medlars and eight percent of the terms for Inspec as shown in column 2 of Table 11. It is obvious that the grouping operation can be performed automatically, given approximate guidelines about group size.

results are obtained when all p-values are set to 1, and when the p-values are varied. A comparison between the results of Tables 13 and 14 covering the frequency range method and those of Tables 9 and 10 for the method using singles, pairs, and triples, respectively, shows that the frequency range process produces results comparable to the best output obtained with the earlier SPT procedure. All the procedures for automatic query formulation produce results which substantially outperform the manually constructed conventional Boolean queries.

## 6. Utilization of Automatic Query Formulation Process

A few words are in order about the methods that could be used to incorporate the foregoing automatic query construction methods in a practical retrieval environment based on Boolean queries and inverted file methodologies. The following sample process requires no alterations of the basic operational bibliographic retrieval system except for the addition of "back-end" programs to carry out the query-document comparisons in the extended p-norm model using the formulas of expressions (4) and (5):

1) Given a user search statement in natural language form, a broad conventional Boolean query formulation is prepared similar to the first statement in the example of Table 4. Typically such a statement consists of single terms and possibly a few anded phrases all connected by or-operators. The initial Boolean statement must be sufficiently broad to retrieve most potentially relevant items.*

*Methods for constructing broad query statements, or for "loosening" narrow Boolean statements have been described in the literature. [14]

## References

[ 1] F.H. Barker, D.C. Veal, and B.K. Wyatt, Towards Automatic Prefile Construction, Journal of Documentation, Vol. 28, No. 1, March 1972, p. 44-55.

[ 2] B.M. Preschel, Indexer Consistency in Perception of Concepts and Choice of Terminology, Final Report, School of Library Science, Columbia University, New York, June 1972.

[ 3] R.C. Lufkin, Determination and Analysis of Some Parameters Affecting the Subject Indexing Process, Master's Thesis, Department of Electrical Engineering, M.I.T., Cambridge, Massachusetts, September 1968.

[ 4] G. Salton, Automatic Information Organization and Retrieval, McGraw Hill Book Company, New York, 1968.

[ 5] G. Salton, editor, The Smart Retrieval System-Experiments in Automatic Document Processing, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1971.

[ 6] K. Sparck Jones, A Statistical Interpretation of Term Specificity and its Application in Retrieval, Journal of Documentation, Vol. 28, No. 1, January 1972, p. 11-21.

[ 7] G. Salton, H.Wu and C.T. Yu, The Measurement of Term Importance in Automatic Indexing, Journal of the ASIS, Vol. 32, NO. 3, May 1981, p. 175-186.

[ 8] S.E. Robertson and K. Sparck Jones, Relevance Weighting of Search Terms, Journal of the ASIS, Vol. 27, No. 3, 1976, p. 129-146.

[ 9] K. Sparck Jones, Experiments in Relevance Weighting of Search Terms, Information Processing and Management, Vol. 15, 1979, p. 133-144.

[10] G. Salton, A Theory of Indexing, Regional Conference Series in Applied Mathematics No. 18, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, February 1975.

[11] H. Wu, On Query Formulation in Information Retrieval, Doctoral Disseration, Cornell University, Ithaca, New York, January 1981.

[12] G. Salton, E.A. Fox, and H. Wu, Extended Boolean Information Retrieval, Technical Report 82-511, Department of Computer Science, Cornell University, Ithaca, New York, 1982.

[13] G. Salton, C.S. Yang and C.T. Yu, A Theory of Term Importance in Automatic Text Analysis, Journal of the ASIS, Vol. 26, No. 1, January-February 1975, p. 33-44.

1. Take the set of terms included in a natural language query formulation; use a stop list to eliminate unwanted function words.

2. Look up the document frequency (number of postings) for all remaining single terms, and eliminate terms with excessive document frequencies.

3. For the remaining terms generate all term pairs and term triples and compute the corresponding expected document frequencies and inverse document frequency weights.

4. Construct an initial query formulation using a few singles with the highest idf weights and term pairs which do not include the chosen singles. Compute the expected number of retrieved items as $\Sigma n_i + \Sigma n_{jk}$.

5. If the expected number of retrieved items is too small, broaden the query by adding singles in decreasing idf weight order and removing redundant pairs. Recompute the expected number of retrieved items following each query alteration.

6. If the expected number of retrieved items is too large, render the query more specific by eliminating singles and adding pairs, or eliminating pairs and adding the corresponding triples. Recompute the expected number of retrieved items and stop the deletion and addition operations when the desired number of items is obtained.

Summary of Boolean Query Construction Process in Disjunctive Normal Form

Table 1

| Pair Formation | Estimated Frequency $n_{ij} = \frac{n_i \cdot n_j}{N+1}$ | Term Pair Weight $1 - \frac{n_{ij}}{N+1}$ | Pairs in Order of "Goodness" | Estimated Frequency |
|---|---|---|---|---|
| 2-1 ho-ex | 4.1 | .9961 | 1. 5-4 ph-pa | 1.1 |
| 3-1 ki-ex | 3.9 | .9962 | 2. 4-1 pa-ex | 1.4 |
| 3-2 ki-ho | 6.1 | .9941 | 3. 6-4 ur-pa | 2.0 |
| 4-1 pa-ex | 1.4 | .9987 | 4. 4-3 pa-ki | 2.0 |
| 4-2 pa-ho | 2.1 | .9980 | 5. 4-2 pa-ho | 2.1 |
| 4-3 pa-hi | 2.0 | .9980 | 6. 5-1 ph-ex | 2.2 |
| 5-1 ph-ex | 2.2 | .9979 | 7. 5-3 ph-ki | 3.2 |
| 5-2 ph-ho | 3.4 | .9967 | 8. 6-5 ur-ph | 3.2 |
| 5-3 ph-ki | 3.2 | .9969 | 9. 5-2 ph-ho | 3.4 |
| 5-4 ph-pa | 1.1 | .9989 | 10. 3-1 ki-ex | 3.9 |
| 6-1 ur-ex | 3.9 | .9962 | 11. 6-1 ur-ex | 4.1 |
| 6-2 ur-ho | 6.1 | .9941 | 12. 2-1 ho-ex | 4.1 |
| 6-3 ur-ki | 5.9 | .9943 | 13. 6-3 ur-ki | 5.9 |
| 6-4 ur-pa | 2.0 | .9980 | 14. 6-2 ur-ho | 6.1 |
| 6-5 ur-ph | 3.2 | .9969 | 15. 3-2 ki-ho | 6.1 |
| | | | | 50.6 |

a) Term Pair Formation

| Term Triple | Frequency $n_{ijk} = \frac{n_{ij} \cdot n_k}{N+1}$ | Weight $1 - \frac{n_{ijk}}{N+1}$ | Term Triple | Frequency $n_{ijk} = \frac{n_{ij} \cdot n_k}{N+1}$ | Weight $1 - \frac{n_{ijk}}{N+1}$ |
|---|---|---|---|---|---|
| 1,3-2 Ex, Ki-Ho | .31 | .9997 | 2,4-3 Ho, Pa-Ki | .16 | .9998 |
| 1,4-2 Ex, Pa-Ho | .11 | .9999 | 2,5-3 Ho, Ph-Ki | .25 | .9998 |
| 1,4-3 Ex, Pa-Ki | .10 | .9999 | 2,5-4 Ho, Ph-Pa | .09 | .9999 |
| 1,5-2 Ex, Ph-Ho | .10 | .9998 | 2,6-3 Ho, Ur-Ki | .46 | .9996 |
| 1,5-3 Ex, Ph-Ki | .16 | .9998 | 2,6-4 Ho, Ur-Pa | .16 | .9998 |
| 1,5-4 Ex, Ph-Pa | .06 | .9999 | 2,6-5 Ho, Ur-Ph | .25 | .9998 |
| 1,6-2 Ex, Ur-Ho | .31 | .9997 | 3,5-4 Ki, Ph-Pa | .08 | .9999 |
| 1,6-3 Ex, Ur-Ki | .30 | .9997 | 3,6-4 Ki, Ur-Pa | .15 | .9998 |
| 1,6-4 Ex, Ur-Pa | .10 | .9999 | 3,6-5 Ki, Ur-Ph | .24 | .9998 |
| 1,6-5 Ex, Ur-Ph | .16 | .9998 | 4,6-5 Ta, Ur-Ph | .08 | .9998 |

b) Term Triple Formation

Formation of Term Pairs and Triples

Table 3

Medlars 1033

30 Queries

| Type of Run | Average Precision at three Recall Points | Percent Difference from Boolean | Percent Difference from Cosine |
|---|---|---|---|
| Basic Boolean Retrieval with Manually Formed Queries (p=∞) | .2065 | - | -62% |
| Vector Processing using Cosine Match and Natural language Queries | .5473 | +165% | - |
| Automatically Formed Conventional Boolean Queries from Original Natural Language Statements Using Singles, Pairs, Triples (S,P,T) unweighted terms | | | |
| estimated retrieved 20 | .2057 | - 0.4% | -62.4% |
| estimated retrieved 30 | .2540 | +23.0% | -53.6% |
| estimated retrieved 50 | .2899 | +40.4% | -47.0% |
| estimated retrieved 100 | .2735 | +32.5% | -50.0% |
| estimated retreived 200 | .1918 | - 7.1% | -64.9% |
| Automatically Formed Conventional Boolean Queries from Original Natural Language Using Single Terms (S) Only | | | |
| estimated retrieved 30 | .2310 | +11.9% | -57.8% |
| estimated retrieved 100 | .3009 | +45.7% | -45.0% |
| estimated retrieved 500 | .1356 | -34.3% | -75.2% |
| Optimal Case Using Retrospective Term Relevance Weights | .7074 | +243% | +29% |

Automatic Generation of Conventional Boolean Queries from Natural Language Text Using Singles, Pairs, Triples

(Medlars 1033 Collection)

Table 5

Assume $D = (d_A, d_B)$ $\quad 0 \le d_A, d_B \le 1$

$Q = (A \text{ and } B)$ or $Q = (A \text{ or } B)$ (unweighted query terms)

$$Sim(D, Q_{A \text{ and } B}) = 1 - \left[ \frac{(1-d_A)^p + (1-d_B)^p}{2} \right]^{1/p} \qquad Sim(D, Q_{A \text{ or } B}) = \left[ \frac{d_A^p + d_B^p}{2} \right]^{1/p}$$

| | | | | | |
|---|---|---|---|---|---|
| $p = 1$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ | $p = 1$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ |
| | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 1/2$ | | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 1/2$ |
| | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ | | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ |
| $p = 2$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ | $p = 2$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ |
| | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 1-1/\sqrt{2}$ | | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 1/\sqrt{2}$ |
| | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ | | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ |
| $p = \infty$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ | $p = \infty$ | $d_A=1, d_B=1$ | $Sim(D,Q) = 1$ |
| | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 0$ | | $d_A=1, d_B=0$<br>$d_A=0, d_B=1$ | $Sim(D,Q) = 1$ |
| | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ | | $d_A=0, d_B=0$ | $Sim(D,Q) = 0$ |

Query-Document Similarity for Unweighted Queries

$D = (d_A, d_B)$; $Q = (A \text{ and } B)$, $Q = (A \text{ or } B)$

Table 7

30 Queries

| Type of Run | | Average Precision at three Recall Points | Percent Difference from Boolean | Percent Difference from Cosine |
|---|---|---|---|---|
| Basic Boolean Retrieval with Manually Formed Queries (p=∞) | | .2065 | - | -62% |
| Vector Processing using Cosine Match and Natural Language Queries (weighted terms) | | .5473 | +165% | - |
| Automatically Formed P-Norm Queries from Original Natural Language Statements Using Singles, Pairs, Triples (weighted document and query terms) | | | | |
| estimated retrieved 20 | p=1 | .5588 | +170.7% | + 2.1% |
| | p=2 | .5447 | +163.8% | - 0.5% |
| | p=5 | .5179 | +150.8% | - 5.4% |
| | p=∞ | .4039 | + 95.6% | -26.2% |
| estimated retrieved 50 | p=1 | .5367 | +159.9% | - 1.9% |
| | p=2 | .5284 | +155.9% | - 3.5% |
| | p=5 | .5062 | +145.2% | - 7.5% |
| | p=∞ | .4595 | +122.6% | -16.0% |
| estimated retrieved 100 | p=1 | .5597 | +171.1% | + 2.3% |
| | p=2 | .5440 | +163.5% | - 0.6% |
| | p=5 | .5028 | +143.5% | - 8.1% |
| | p=∞ | .4594 | +122.5% | -16.1% |
| estimated retreived 200 | p=1 | .5331 | +158.3% | - 2.6% |
| | p=2 | .5366 | +159.9% | - 2.0% |
| | p=5 | .4911 | +137.9% | -10.3% |
| | p=∞ | .4500 | +118.0% | -17.8% |
| Automatically Formed P-Norm Queries from Original Natural Language Statements Using Single Terms Only (weighted document and query terms) | | | | |
| estimated retrieved 30 | p=1 | .2542 | + 23.1% | -53.6% |
| | p=2 | .2544 | + 23.2% | -53.5% |
| | p=5 | .2552 | + 23.6% | -53.4% |
| | p=∞ | .2553 | + 23.7% | -53.3% |
| estimated retrieved 100 | p=1 | .4706 | +127.9% | -14.0% |
| | p=2 | .4675 | +126.4% | -14.6% |
| | p=5 | .4530 | +119.4% | -17.2% |
| | p=∞ | .4394 | +112.8% | 19.7% |
| estimated retrieved 500 | p=1 | .5550 | +168.8% | + 1.4% |
| | p=2 | .5397 | +161.4% | - 1.4% |
| | p=5 | .4943 | +139.4% | - 9.7% |
| | p=∞ | .4594 | +122.5% | -16.1% |
| Optimal Case Using Retrospective Term Relevance Weights | | .7074 | +243% | +29% |

Automatic Generation of p-Norm Queries from Natural Language Statements Using Singles, Pairs and Triples (Medlars 1033 Collection)

Table 9

| Frequency Range | Number of Terms | Percentage of Terms (8 classes) | Percentage of Terms (5 classes) | Percentage of Terms (3 classes) | Percentage of Terms (2 classes) |
|---|---|---|---|---|---|
| 1 | 4681 | 53.5% | 53.5% | 53.5% | 53.5% |
| 2 | 1099 | 12.6% | 12.6% | 26.2% | |
| 3 | 577 | 6.6% | | | |
| 4-5 | 613 | 7.0% | 13.6% | | 46.5% |
| 6-8 | 492 | 5.6% | 10.5% | 20.3% | |
| 9-14 | 429 | 4.9% | | | |
| 15-29 | 434 | 5.0% | 9.8% | | |
| 30-357 | 425 | 4.8% | | | |

a) Frequency Range Subdivision for Medlars 1033 Collection

| Frequency Range | Number of Terms | Percentage of Terms (8 classes) | Percentage of Terms (5 classes) | Percentage of Terms (3 classes) |
|---|---|---|---|---|
| 2 | 5209 | 35.5% | 35.5% | 35.5% |
| 3 | 2103 | 14.3% | 14.3% | 31.0% |
| 4 | 1156 | 7.8% | 16.7% | |
| 5-6 | 1306 | 8.9% | | |
| 7-10 | 1232 | 8.4% | 16.9% | |
| 11-20 | 1245 | 8.5% | | |
| 21-58 | 1199 | 8.2% | 16.6% | 33.5% |
| 59+ | 1233 | 8.4% | | |

b) Frequency Range Subdivision for Inspec 12684 Collection

Frequency Range Determination

Table 11

Medlars 1033
30 Queries

| Type of Run | Average Precision | Percent Difference from Boolean | Percent Difference from Cosi |
|---|---|---|---|
| Basic Boolean Retrieval with Manually Formed | .2065 | - | -62% |
| Vector Processing Using Cosine Match and Natural Language Queries | .5473 | +165% | - |
| 8 ranges starting at frequency of 1,2,3,4,6,9,15,30-357 p=2,1.5,1.3,1,1,1.3,1.5,2; weighted | .5042 | +144.2 | - 7.9 |
| 8 ranges all p=1 weighted | .5022 | +143.3 | - 8.2 |
| 8 ranges all p=∞ unweighted | .0371 | - 82.0 | -93.2 |
| 8 ranges all p=∞ weighted | .0872 | - 57.7 | -84.1 |
| 5 ranges starting at 1,2,3,6,15-357 p=1.5,1.3,1,1.3,1.5; weighted | .4953 | +139.9 | - 9.5 |
| 5 ranges all p=1 weighted | .5005 | +142.4 | - 8.6 |
| 5 ranges all p=∞ unweighted | .0371 | - 82.0 | -93.2 |
| 5 ranges all p=∞ weighted | .1016 | - 50.8 | -81.4 |
| 3 ranges starting at 1,2,5-357 p=1.5,1,1.5; weighted | .5229 | +153.3 | - 4.5 |
| 3 ranges all p=1 weighted | .5329 | +158.1 | - 2.6 |
| 3 ranges all p=∞ unweighted | .0371 | - 82.0 | -93.2 |
| 3 ranges all p=∞ weighted | .1027 | - 50.3 | -81.2 |
| Optimal Retrospective Term Relevance Weights | .7074 | +243 | +20 |

Automatic Generation of p-Norm Queries from Natural Language Statements

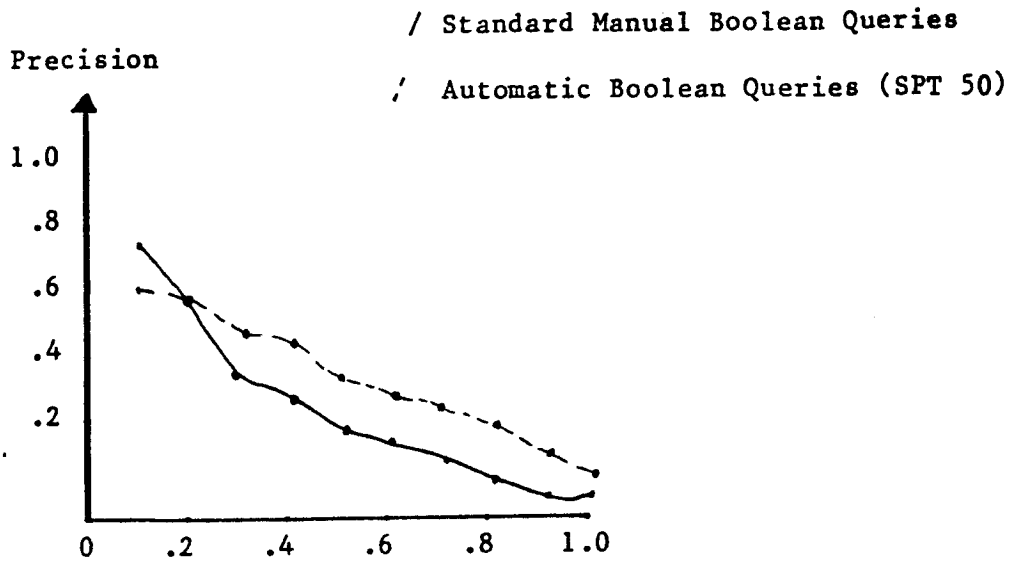
Using Frequency Range Method (outer p-operator equal to 1)

(Medlars 1033, 30 Queries)

Table 13

| Recall Level | Average Precision for 30 Queries | |
|---|---|---|
| | Manual Queries Standard Boolean | Automatic Boolean Queries (S,P,T) Set to Retrieve 50 Items |
| 0.1 | 0.5606 | 0.4731 |
| 0.2 | 0.4364 | 0.4261 |
| 0.3 | 0.2917 | 0.3990 |
| 0.4 | 0.2387 | 0.3421 |
| 0.5 | 0.1871 | 0.2890 |
| 0.6 | 0.1562 | 0.2508 |
| 0.7 | 0.1073 | 0.1867 |
| 0.8 | 0.0774 | 0.1656 |
| 0.9 | 0.0385 | 0.0873 |
| 1.0 | 0.0324 | 0.0282 |
| | Average Improvement | +40.4% |

a) Typical Recall-Precision Table




b) Recall Precision Graph

Comparison of Manual Boolean Queries with Automatic Boolean Formulations (SPT 50) Constructed from Natural Language Statements (Medlars 1033, 30 queries)

Fig. 1