Concept Based Query Expansion

Yonggang Qiu
Department of Computer Science
Swiss Federal Institute of Technology
Zurich
CH-8092 Zurich, Switzerland

H.P. Frei
UBILAB
Union Bank of Switzerland
CH-8021 Zurich, Switzerland

Abstract

Query expansion methods have been studied for a long time - with debatable success in many instances. In this paper we present a probabilistic query expansion model based on a similarity thesaurus which was constructed automatically. A similarity thesaurus reflects domain knowledge about the particular collection from which it is constructed. We address the two important issues with query expansion: the selection and the weighting of additional search terms. In contrast to earlier methods, our queries are expanded by adding those terms that are most similar to the concept of the query, rather than selecting terms that are similar to the query terms. Our experiments show that this kind of query expansion results in a notable improvement in the retrieval effectiveness when measured using both recall-precision and usefulness.

1. Introduction

In weighted Information Retrieval (IR) the number of retrieved documents is related to the number of appropriate search terms. The more search terms, the more documents delivered by the IR system. This is why thesaurus browsers are integrated into modern IR systems. They help to find additional search terms [Qiu 92]. However, the aim of the retrieval activity is not to retrieve a large number of documents. Rather, users are interested in a high usefulness of the retrieved documents. The purpose of this paper is to disclose how a higher usefulness can be achieved when a query is expanded by choosing carefully additional search terms on the basis of statistical co-occurrence data.

Research on automatic query expansion (or modification) was already under way before 1960 when initial requests were enlarged on the grounds of statistical evidence [Spa 91]. The idea was to obtain additional relevant documents through expanded queries based on the co-occurrence of terms. At that time, the co-occurrence of index terms was usually the only criterion in the absence of relevance feedback. However, this kind of automatic query expansion has not been very successful. The retrieval effectiveness of the expanded queries was often no greater than, or even less than, the effectiveness of the original queries [Min 72, Pea 91, Sme 83].

We assume that documents and queries are represented by a relatively small number of weighted index and search terms. It is to be noted that the probability of a term representing the concept of a document is not identical to the probability of the document representing the meaning of the term. Therefore, a relationship between terms can be based on the probabilities of the documents representing the terms.

In this paper, we first give a brief introduction into previous work. We then present a construction method that allows to get a similarity thesaurus [Sch 92] from a given document collection. In section 4, we describe a probabilistic query expansion and weighting model using the similarity thesaurus. After describing our test setting, some results of experiments carried out with three standard test collections are presented in section 5. We then point out two reasons why many of the early automatic query expansion methods failed. Finally, we conclude with the main findings and point out further research and possible applications of the methods presented.

2. Automatic Query Expansion

The automatic query expansion or modification based on term co-occurrence data has been studied for nearly three decades. The various methods proposed in the literature can be classified into the following four groups:

1) Simple use of co-occurrence data. The similarities between terms are first calculated based on the association hypothesis and then used to classify terms by setting a similarity threshold value [Les 69, Min 72, Spa 71]. In this way, the set of index terms is subdivided into classes of similar terms. A query is then expanded by adding all the terms of the classes that contain query terms. It turns out that the idea of classifying terms into classes and treating the members of the same class as equivalent is too naive an approach to be useful [Min 72, Pea 91, Spa 91].

2) Use of document classification. Documents are first classified using a document classification algorithm. Infrequent terms found in a document class are considered similar and clustered in the same term class (thesaurus class) [Cro 90]. The indexing of documents and queries is enhanced either by replacing a term by a thesaurus class or by adding a thesaurus class to the index data. However, the retrieval effectiveness depends strongly on some parameters that are hard to determine [Cro 92]. Furthermore, commercial databases contain millions of documents and are highly dynamic. The number of documents is much larger than the number of terms in the database. Consequently, document classification is much more expensive and has to be done more often than the simple term classification mentioned in 1).

3) Use of syntactic context. The term relations are generated on the basis of linguistic knowledge and co-occurrence statistics [Gre 92, Rug 92]. The method uses a grammar and a dictionary to extract for each term t a list of terms. This list consists of all the terms that modify t. The similarities between terms are then calculated by using these modifiers from the list. Subsequently, a query is expanded by adding those terms most similar to any of the query terms. This produces only slightly better results than using the original queries [Gre 92].

4) Use of relevance information. Relevance information is used to construct a global information structure, such as a pseudothesaurus [Sal 71, Sal 80] or a minimum spanning tree [Sme 83]. A query is expanded by means of this global information structure. The retrieval effectiveness of this method depends heavily on the user relevance information. Moreover, the experiments in [Sme 83] did not yield a consistent performance improvement. On the other hand, the direct use of relevance information, by simply extracting terms from relevant documents, is proved to be effective in interactive information retrieval [Har 92, Sal 90]. However, this approach does not provide any help for queries without relevance information.

In addition to automatic query expansion, semi-automatic query expansion has also been studied [Ekm 92, Han 92, Wad 88]. In contrast to the fully automated methods, the user is involved in the selection of additional search terms during the semi-automatic expansion process. In other words, a list of candidate terms is computed by means of one of the methods mentioned above and presented to the user who makes the final decision. Experiments with semi-automatic query expansion, however, do not result in significant improvement of the retrieval effectiveness [Ekm 92].

Among the various approaches, automatic query expansion by using plain co-occurrence data is the simplest method. In contrast to the approaches presented, we use a similarity thesaurus [Sch 92] as the basis of our query expansion. First we show how such a similarity thesaurus is constructed and then we present a query expansion model in order to overcome the drawbacks of using plain co-occurrence data.

3. Constructing a Similarity Thesaurus

A similarity thesaurus [Sch 92] is a matrix that consists of term-term similarities. In contrast to a co-occurrence matrix, a similarity thesaurus is based on how the terms of the collection "are indexed" by the documents. We show that a similarity thesaurus can be constructed automatically by using an arbitrary retrieval method with the roles of documents and terms interchanged. In other words, the terms play the role of the retrievable items and the documents constitute the "indexing features" of the terms.

With this arrangement a term t_i is represented by a vector $\vec{t}_i = (d_{i1}, d_{i2}, ..., d_{in})^T$ in the document vector space (DVS) defined by all the documents of the collection. The d_{ik}'s signify feature weights of the indexing features (documents) d_k with respect to the item (term) t_i and n is the number of features (documents) in the collection. We adopt the normalized tf • idf weighting scheme [Sal 88] and define the feature weights d_{ik} by the feature frequency (ff), the inverse item frequency (iif), and the maximum feature frequency (maxff) as follows.

$$d_{ik} = \frac{(0.5+0.5\,\frac{ff(d_k,t_i)}{maxff(t_i)})\cdot iif(d_k)}{\sqrt{\sum_{j=1}^{n}((0.5+0.5\,\frac{ff(d_j,t_i)}{maxff(t_i)})\cdot iif(d_j))^2}} \qquad (1)$$

where

$ff(d_k,t_i)$ is the within-item frequency of feature d_k in item t_i.

$iif(d_k) = \log(\frac{m}{|d_k|})$ is the inverse item frequency of feature d_k; m is the number of items in the collection and $|d_k|$ is the number of different items indexed by the feature d_k. In other words, $|d_k|$ is the number of terms appearing in document d_k.

$maxff(t_i)$ is the maximum within-item frequency of all features in item t_i.

The feature frequency ff(d_k,t_i) specifies the number of occurrences of the indexing feature d_k in item t_i. It is analogous to the term frequency tf(t_i,d_k) when the documents are indexed by terms. The definition of the inverse item frequency shows that a short document plays a more important role than a long document. If two terms co-occur in a long document, the probability that the two terms are similar is smaller than if they would co-occur in a short document. From formula (1), we can derive that

$$|\vec{t}_i| = \sqrt{\sum_{k=1}^{n} d_{ik}^2} = 1 \qquad (2)$$

This means that $\vec{t}_i$ is a unit vector representing the term in the document vector space DVS.

With these definitions, we define the similarity between two terms t_i and t_j by using a similarity measure such as the simple scalar vector product:

$$\mathrm{SIM}(t_i,t_j) = \vec{t}_i^{\,T} \cdot \vec{t}_j = \sum_{k=1}^{n} d_{ik} \cdot d_{jk} \qquad (3)$$

The similarity thesaurus is constructed by determining the similarities of all the term pairs (t_i,t_j). The result is a symmetric matrix whose values are in the following range:

$$0 \leq \mathrm{SIM}(t_i, t_j) \leq 1 \qquad (4)$$

Earlier studies [Min 72, Spa 71] employed the probabilities of the terms representing the documents to build a co-occurrence matrix. In contrast, our similarity thesaurus is based on the probabilities of the documents representing the meanings of the terms. In other words, we use the weights of the documents in the terms.

The construction of such a similarity thesaurus for a large database is computationally expensive. However, it is a single expense. Adding a few documents to a database with millions of documents hardly changes the relationships between terms. Furthermore, an update of the similarity thesaurus can be achieved by modifying only those entries corresponding to terms contained in the new documents. More precisely, we can evaluate the similarities between the newly arrived terms and then update corresponding entries in the similarity thesaurus. How much of this can be done without rescaling is an open research issue and is likely to depend on the kind of domain knowledge.

4. A Probabilistic Query Expansion Model

As already mentioned, most attempts at automatically expanding queries failed to improve the retrieval effectiveness. The opposite case was often true: Expanded queries were less effective than the original queries. Therefore, it was often concluded that automatic query expansion based on statistical data was unable to bring a substantial improvement in the retrieval effectiveness [Pea 91]. However, our belief is that two of the basic problems were not solved when expanding queries automatically:

1) the selection of suitable terms;
2) the weighting of the selected additional search terms.

We pointed out in section 2 that with most methods, terms are selected that are strongly related to one of the query terms. The methods differ in the kind of relationships used. The entire query - in other words, the query concept - is seldom taken into account. This may be compared to translating from a natural language text into another: A dictionary look-up for a word does not give the final answer in many cases. Rather, the translator who knows the meaning of the text has to choose the suitable word from an entire list of possible translations. Likewise, we should consider a term that is similar to the query concept rather than one that is only similar to a single term in the query.

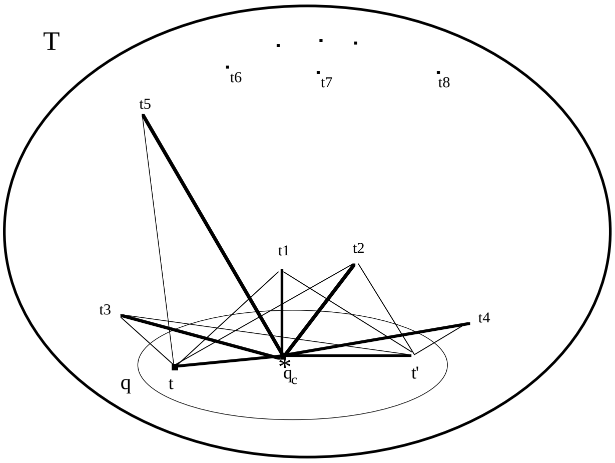

Fig. 1: Relationships between terms and query in the DVS

Let T be a set of indexing terms and q be the user query containing two terms, t and t', as shown in Fig. 1. The similarity thesaurus of the collection contains the pair-wise similarities of all the terms with respect to this particular collection. In Fig. 1 we have represented these pair-wise similarities with fine lines. The closer two linked terms are to each other, the more similar they are. t_3 is more similar to t than all the others, t_4 is more similar to t' than all the other terms. In addition, we show the virtual term q_c that is supposed to represent the general concept of the query q. This concept may be obtained by simply calculating the centroid of q or by using an appropriate information structure. If the number of terms to be added to the query is 2, should we choose t_3 and t_4 or some other terms? The answer to this question is pretty obvious when considering Fig. 1: Since t_1 and t_2 are the terms most similar to the query concept q_c, they will be chosen as additional search terms instead of t_3 and t_4. In what follows we explain how such terms are determined by using the similarity thesaurus.

A query q is represented by a vector $\vec{q} = (q_1, q_2, ..., q_m)^T$ in the term vector space (TVS) defined by all the terms of the collection. Here, the q_i's are the weights of the search terms t_i contained in the query q; m is the total number of terms in the collection.

The probability that a term t is similar to the concept of query q is P(S|q,t). In order to estimate the probability, we first apply Bayes' theorem and get:

$$P(S|q,t) = P(S|t) \cdot \frac{P(q|S,t)}{P(q|t)} = \frac{P(S|t)}{P(q|t)} \cdot P(q|S,t)$$

We assume that the distribution of terms in all the queries to which a term is similar is independent:

$$\begin{aligned} P(S|q,t) &= \frac{P(S|t)}{P(q|t)} \cdot \prod_{i=1}^{m} P(q_i|S,t) \\ &= \frac{P(S|t)}{P(q|t)} \cdot \prod_{i=1}^{m} \frac{P(S|q_i,t)}{P(S|t)} \cdot P(q_i|t) \\ &= \frac{1}{P(q|t) \cdot P(S|t)^{m-1}} \cdot \prod_{i=1}^{m} P(S|q_i,t) \cdot P(q_i|t) \end{aligned}$$

An additional assumption is that the similarity between a term and the concept of a query depends only on the terms contained in the query and not on other terms. Hence,

$$P(S|q,t) = \frac{1}{P(q|t) \cdot P(S|t)^{m-1}} \cdot \prod_{t_i \in q} P(S|t_i,t) \cdot P(t_i|t) \qquad (5)$$

Here, $P(S|t_i,t)$ is the probability that the query term t_i is similar to the term t. $P(t_i|t)$ is the probability that the query term t_i represents the query q. P(q|t) is the probability that the query q will be submitted to the IR system. P(S|t) is the probability that the term t is similar to an arbitrary query.

Formula (5) elucidates that the probability of a term to be similar to a query depends on the following factors:

- the similarities between the term and all the query terms;
- the weights of the query terms.

As mentioned above, the objective of our query expansion scheme is to find suitable additional query terms. They should have the property of being similar to the entire query rather than to individual query terms. We showed that such terms can only be found when an overall similarity scheme is taken into account. Since the similarity thesaurus expresses the similarity between the terms of the collection in the DVS (defined by the documents of the collection), we map the vector $\vec{q}$ from the TVS (defined by the terms of the collection) into a vector in space DVS. This way, the overall similarity between a term and the query can be estimated. Each query term t_i is defined by the unit vector $\vec{t}_i$ which itself is defined by a number of documents as was pointed out in section 3. q_i is the weight of term t_i in the query. In other words, the concept expressed by the term t_i in the query has an importance of $q_i \cdot \vec{t}_i$ for the query. We assume that the concept expressed by the entire query depends only on the terms in the query. Therefore, the vector $\vec{q}_c$ representing the query concept in space DVS is the virtual term vector:

$$\vec{q}_c = \sum_{t_i \in q} q_i \cdot \vec{t}_i \qquad (6)$$

The similarity between a term and the query q is denoted by Simqt(q,t). The scalar vector product is used as similarity measure:

$$\begin{aligned} Simqt(q,t) &= \vec{q}_c^{\,T} \cdot \vec{t} = \Big(\sum_{t_i \in q} q_i \cdot \vec{t}_i \Big)^T \cdot \vec{t} \\ &= \sum_{t_i \in q} q_i \cdot (\vec{t}_i^{\,T} \cdot \vec{t} \,) \end{aligned}$$

Where $(\vec{t}_i^{\,T} \cdot \vec{t}\,)$ is the similarity between two terms defined in formula (3):

$$Simqt(q,t) = \sum_{t_i \in q} q_i \cdot SIM(t_i,t) \qquad (7)$$

It is to be noted that the values of $SIM(t_i,t)$ are the entries of our similarity thesaurus and therefore are precomputed. All the terms of T can now be ranked according to their Simqt value with respect to the query q. The terms t with higher Simqt(q,t) are candidates to be considered as additional search terms.

It seems natural to choose the weight $weight_a(q,t)$ of a selected additional search term t as a function of Simqt(q,t):

$$weight_a(q,t) = \frac{Simqt(q,t)}{\sum_{t_i \in q} q_i} \qquad (8)$$

$$\text{where } 0 \leq weight_a(q,t) \leq 1$$

With this choice, additional search terms with similarity 1 to all the terms in the query get a weight of 1, additional search terms with similarity 0 to all the terms in the query get a weight of 0.

After having determined how terms are selected and weighted, we can take into account the domain knowledge contained in the similarity thesaurus to find the most likely intended interpretation for the user's query. When relevance feedback techniques are used, queries are expanded by adding terms from the retrieved relevant documents. The experiments in [Har 92] show that adding as few as 20 top properly ranked terms, rather than all the terms from the retrieved relevant documents, can result in significant performance improvement. This is the reason we also add only those terms that are ranked in the top positions by the Simqt function.

Another reason for only choosing the top ranked terms as opposed to setting a weight threshold is for the efficiency sake. The efficiency (response time) of an IR system depends heavily on the number of terms of the query submitted to the system. With a threshold, this number cannot be predicted.

Therefore, the query q is expanded by adding the following query q_e:

$$\vec{q}_e = (q_{e1},\ q_{e2}, \ldots,\ q_{em})^T \qquad (9)$$

where

$$q_{ej} = \begin{cases} \text{weight}_a(q,t_j) & \text{if } t_j \text{ belongs to the top r ranked terms} \\ 0 & \text{otherwise} \end{cases}$$

r is the number of terms to be added or modified in weight.

The resulting expanded query $q_{expanded}$ is:

$$\vec{q}_{expanded} = \vec{q} + \vec{q}_e \qquad (10)$$

After this expansion process, new terms may have been added to the original query and the weight of an original query term may have been modified had the term belonged to the top ranked terms.

The important point of this method is that additional search terms are selected dynamically when a query is submitted. More precisely, the query and the terms most similar to the query concept are classified in the same class. This is in contrast to earlier studies when term-classification was done statically. We believe an important weakness of the static classification is that it is far too limited to capture both the rich semantics of data collections and the information need of users.

Let us explain our approach by setting r, the number of terms to be added or modified in weight, to m, the total number of terms. In this case, the query q is expanded by q_e containing all the m terms. Furthermore, let us consider an arbitrary document $\vec{d} = (d_1,\ d_2, \ldots,\ d_m)^T$ in the TVS where the d_i's signify term weights for this particular document. Then, the similarity between $\vec{d}$ and $\vec{q}_e$ is:

$$\begin{aligned} \vec{d}^T \cdot \vec{q}_e &= \sum_{t_j \in d} d_j \cdot q_{ej} \\ &= \frac{1}{\sum_{t_i \in q} q_i} \sum_{t_j \in d} d_j \cdot \sum_{t_i \in q} q_i \cdot \text{SIM}(t_i,t_j) \\ &= C_q \sum_{t_j \in d} \sum_{t_i \in q} d_j \cdot q_i \cdot \text{SIM}(t_i,t_j) \qquad (11) \end{aligned}$$

$$\text{where} \quad C_q = \frac{1}{\sum_{t_i \in q} q_i}$$

Since the constant C_q depends only on the query, it does not affect the ranking of the documents with respect to the query. It is to be noted that formula (11) is analogous to the similarity indicated in [Won 87, p. 303] for the Generalized Vector Space Model (GVSM). This means that both the method proposed in this paper and the GVSM are going along the same lines. Therefore, the GVSM can also be interpreted as a kind of query expansion method.

There are, however, two significant differences between the two methods. First, the relationship between terms is computed in a different way, although both methods use co-occurrence data. We construct a similarity thesaurus as described in section 3. In the GVSM, concepts are derived from terms and used as a basis of the vector space in which similarities are computed. Secondly, the GVSM includes all the terms in the expansion and "uses" q_e for ranking documents as shown in formula (11). Yet, in our approach, we expand the query only by a few carefully chosen terms and use $q_{expanded}$.

Similarly, the latent semantic indexing (LSI) approach [Dee 90] tends to find which terms are used to describe a document or a query. In LSI, a set of terms used to index documents is replaced by a relatively small number of orthogonal factors. These factors represent extracted common meaning components of many different terms or documents. However, the choice of the number of factors is critical to LSI. If the number of factors needs to be changed, the latent semantic indexing analysis, a time consuming process, has to be reperformed. Although the choice of r in formula (9) is also critical, it can easily be changed to satisfy the user information need.

5. Experiments and their Results

For our experiments, we used the three standard test collections shown in Table 1. We compared the retrieval effectiveness of our automatic query expansion approach with the standard retrieval method using original queries only. For the collections CACM and MED, after extracting all the words from the collections and removing stop words, we used stemmed terms to index both queries and documents. For NPL, we used the existing indexed form. Table 1 indicates the number of documents, the number of queries with relevance information, the number of terms, the average number of terms per document and query, the number of terms in queries and the average number of relevant documents per query. It can be seen that the MED collection is rather small and the NPL collection is quite sizable as a test collection. The CACM collection is of medium size.

Collection	MED	CACM	NPL
documents	1033	3204	11429
queries	30	52	93
terms	8663	7121	7492
avg. doc length	54.69	24.26	19.96
avg. query length	10.45	11.5	7.15
terms in queries	271	356	337
avg. rele. docs	23.2	15.31	22.41

Table 1: Collections used for experiments

Term weights in both documents and queries are determined according to the normalized tf • idf weighting scheme [Sal 88], see also formula (1). For the similarity calculations, the scalar vector product was used. In addition, the construction method described in section 3 was used to determine the similarities between all the terms in the collections, i.e., to build up the similarity thesauri.

Then, for each query, we rank the terms of the collection in decreasing order according to formula (7). Note that this can be achieved very efficiently as the pre-computed similarities from the similarity thesauri can be used. The top ranked terms are chosen to expand or modify the query according to formulae (8), (9), and (10).

The results were evaluated by applying the average precision of a set of queries at three representative recall points, namely 0.25, 0.50, and 0.75. In addition, the usefulness [Fre 91] was measured to compare the automatic query expansion method with the standard method using original queries.

Table 2 shows the retrieval quality difference between the original queries and the expanded queries. The figures indicate that our automatic query expansion method yields a considerable improvement in the retrieval effectiveness in both automatically indexed document collections, i.e., MED and CACM. In addition, there is also an improvement with the collection indexed by carefully chosen terms, i.e., NPL.

Collection	MED	CACM	NPL
avg. precision of original queries	0.5446	0.2718	0.1818
Number of additional terms	80	100	800
avg. precision of expanded queries	0.6443	0.3339	0.2349
Improvement	+ 18.31 %	+ 22.85 %	+ 29.21 %

Table 2: Improvement using expanded queries

It seems that the improvement increases with the size of the collection. In addition, the improvement increases with the number of additional search terms that expand the original query as long as the collection is large enough. Obviously, the large collection contains more domain knowledge than the small ones. As a consequence, the quality of the similarity thesaurus created from the large collection is better than the quality of the thesauri belonging to the small collections. This seems to be the reason our query expansion method using the similarity thesaurus yields higher performance improvement in the large collection than in the two small ones.

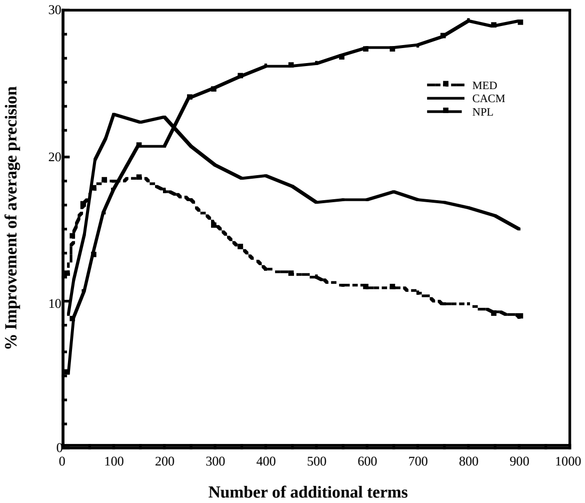

Fig. 2: Improvement using expanded queries with various numbers of additional terms

In Fig. 2, we show how the number of additional terms affects the retrieval effectiveness. It can be seen easily that the improvement by expanded queries increases when the number of additional terms increases. When the number of additional terms is between 100 and 200, the improvement of the retrieval effectiveness remains constant in the small collections MED and CACM. Once the number of additional terms gets to be larger than 200, the improvement decreases in the small collections, but continues to increase in the relatively large collection NPL. This could be explained by the fact that more search terms are needed to distinguish relevant documents from non-relevant documents in large collections.

The results shown in Fig. 2 indicate that expanding a query by roughly 100 top terms seems to be the safe way to go. Yet, when using relevance feedback information, the number of additional terms should be as few as 20 [Har 92]. Why is the number of terms to be added smaller when relevance feedback is used as opposed to our automatic expansion method? With relevance feedback, only the terms contained in the retrieved documents are considered. The number of these terms is much smaller than if one were to consider all the terms of the collection. Furthermore, some of them may be dissimilar to the query concept. In contrast to relevance feedback, there is a much larger number of terms that are considered to be additional search terms.

The same evaluation was done by applying the usefulness measure instead of the adjusted precision. The usefulness measure [Fre 91] is a relative measure which compares a retrieval method A and a retrieval method B. It is based on relative relevance judgments as opposed to absolute relevant and non-relevant assessment. The measure has the added ability of determining an error probability that indicates how stable the result is. With the measure, the usefulness u(A,B) indicates how often, on the average, method B performs better than method A. The adjusted usefulness u*(A,B) indicates how much method B performs better than method A. The error probability Pk expresses the reliability of the usefulness value. The values of u(A,B) and u*(A,B) are between -1.0 and 1.0, Pk is between 0.0 and 1.0. The higher u(A,B) and u*(A,B) are, the more effective method B is compared to method A. The smaller Pk, the more reliable the usefulness value.

In our experiments, the query expansion method is B, the method using the original - not expanded - queries is A. We compared the retrieval effectiveness of the two methods using the top 20 documents ranked by the two methods for each query. In the evaluation, the preferences derived from the top 20 documents ranked by both methods are used.

Collection	MED	CACM	NPL
Number of additional terms	80	100	800
u(A,B)	0.7328	0.5698	0.7478
u*(A,B)	0.0597	0.0438	0.0925
Pk	0.0004	0.0006	0.0000

Table 3: Usefulness of expanded queries with respect to original queries

The usefulness of the automatic query expansion method with respect to the method using original queries is shown in Table 3. The results confirm that the query expansion method performs consistently better than the method using original queries in the three collections. The error probability values are quite small, 0.0 or almost 0.0. This is an indication that the usefulness values here are reliable. Since users of an IR system are normally interested in the top ranked documents, the information needs of the users are much better satisfied by using the expanded queries than the original queries.

In Fig. 3, we study how the number of expanded terms affects the usefulness of the query expansion method with respect to the method using original queries. The results shown in Fig. 3 are consistent to the ones when the recall-precision measure is used. That is, the number of terms to be added should be determined according to the number of documents of the collections in order to produce a high usefulness. The number of expanded terms suggested in Fig. 3 is around 50 for the collections MED and CACM, and around 350 for the collection NPL.

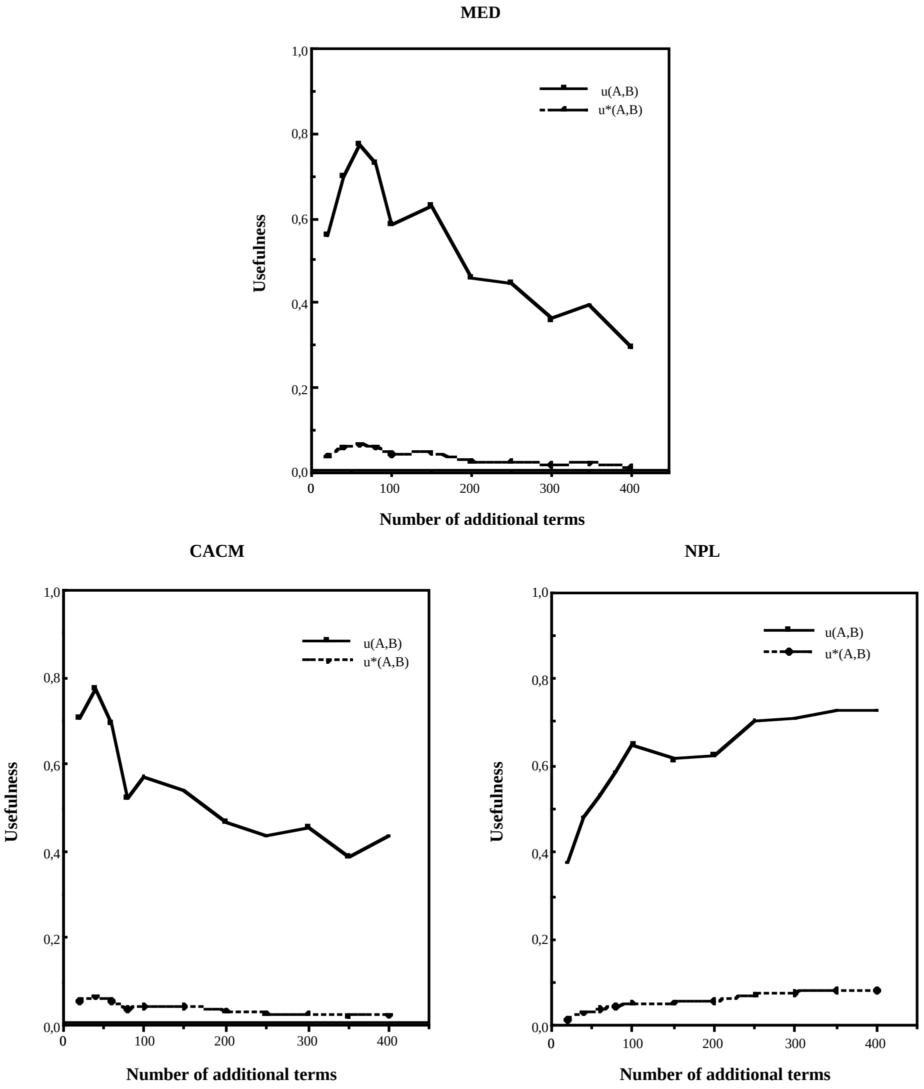

Fig. 3: Usefulness and the number of additional terms

6. Why did many of the early methods fail?

As already mentioned, the usual query expansion methods tend to add a term when it is strongly related to one of the query terms. In other words, during the expansion process those term-term pairs with a similarity value less than a threshold value are not taken into account. As we also mentioned, most of these methods failed to improve the retrieval effectiveness. With the same idea, we carried out some experiments to see how a threshold affects the retrieval effectiveness when using our query expansion model. Only those terms that have a high similarity value to query terms are considered as candidates of additional search terms. They are ranked according to the following adjusted Simqt function:

$$\mathrm{Simqt}(q,t) = \sum_{(t_i \in q) \text{ and } (\mathrm{SIM}(t_i,t) \geq \mathrm{threshold})} q_i \cdot \mathrm{SIM}(t_i,t) \qquad (12)$$

As is done in many traditional query expansion methods, we also added the top ranked terms to the original query according to formulae (8), (9), and (10).

Fig. 4 shows the relationships among the retrieval effectiveness (improvement of average precision of expanded queries over original queries), the threshold (0.0 ~ 1.0) of similarities between terms and the number of additional terms. The results indicate that retrieval effectiveness decreases when the threshold value increases. When the threshold value is greater than 0.6 in MED, 0.8 in CACM and 0.3 in NPL, the expanded queries perform even less effective than the original queries.

However, with the term classification methods mentioned in section 2, high threshold value should be used. Otherwise too many terms are classified in the same class and terms are difficult to distinguish from each other. As we have seen, however, when we choose a high threshold value, the retrieval effectiveness decreases. This is one reason why many of the early automatic query expansion methods failed.

Fig. 5 shows a feature of the similarity thesauri of the three collections. One can see that the distribution of the number of term-term pairs with a similarity value greater than 0 is the λ distribution. Most term-term pairs have a quite small similarity value and few term-term pairs have a high one. When the threshold value of similarities gets larger, the number of those term-term pairs with a similarity greater than the threshold value decreases rapidly. The number of candidates of additional search terms becomes quite small. As a result, the top ranked terms may be dissimilar to query concepts. This is another reason why early automatic query expansion methods failed to improve the retrieval effectiveness.

7. Conclusion

In this paper, we present a query expansion model based on the domain knowledge contained in an automatically constructed similarity thesaurus. This model is primarily concerned with the two important problems of query expansion, namely with the selection and with the weighting of additional search terms. The term selection relies on the overall similarity between the query concept and terms of the collection rather than on the similarity between a query term and the terms of the collection. The experiments carried out on the three test collections show that consistent improvement in the retrieval effectiveness can be expected.

The main results of the study are summarized as fol-

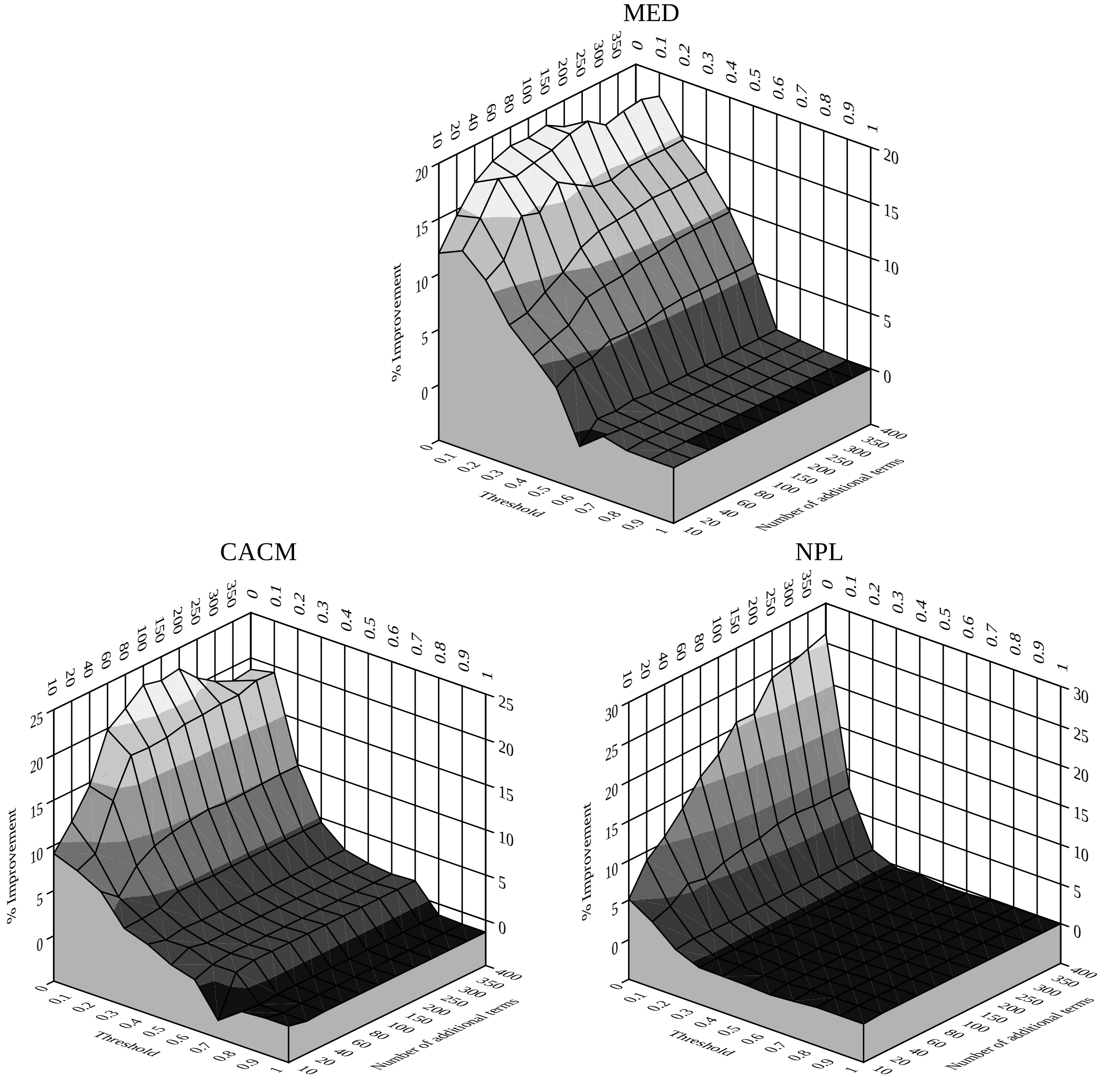

Fig. 4: Effect of the similarity threshold.

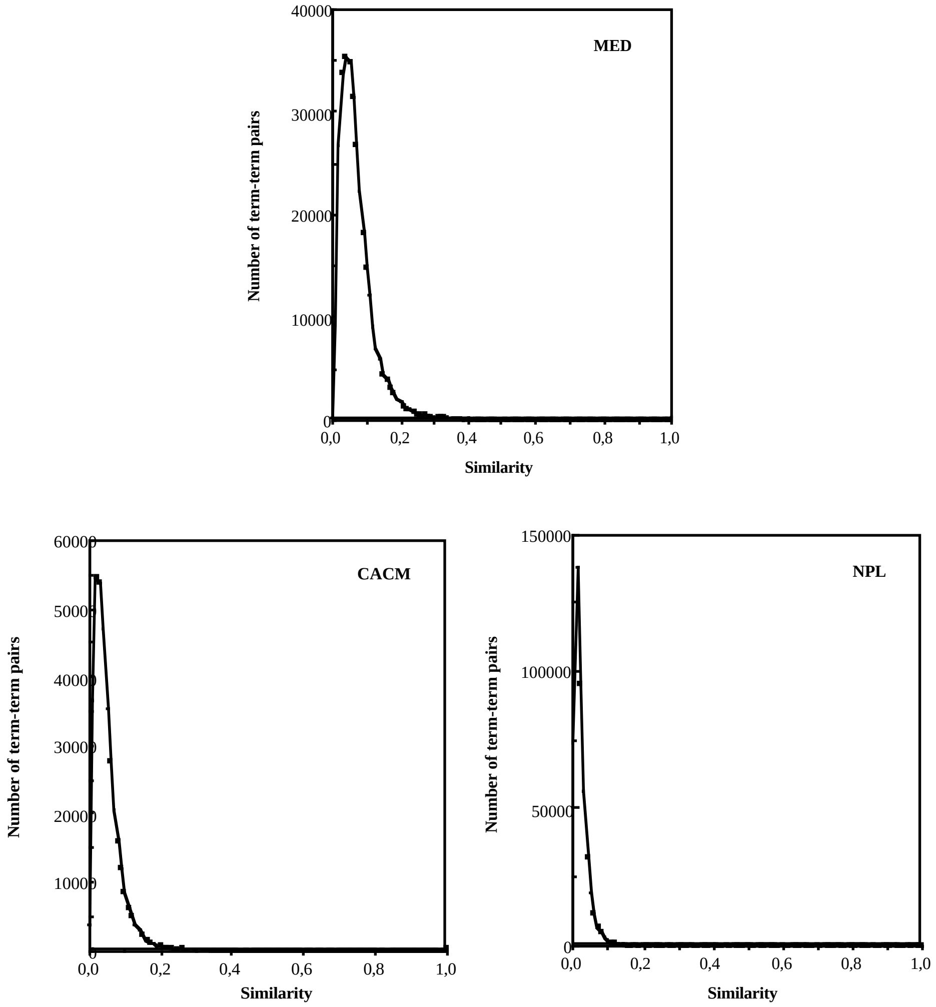

Fig. 5: Feature of similarity thesauri

lows:

1) The automatic query expansion method based on statistical co-occurrence data can result in significant improvement in the retrieval effectiveness when measured using both recall-precision and usefulness. Consistent performance improvement was achieved in both automatically indexed test collections and a test collection indexed by carefully chosen terms.
2) Since the quality of the similarity thesaurus created for a large collection seems to be better than the one for a small collection, the retrieval effectiveness seems to increase with the size of the collection. Likewise, the number of additional terms per query seems to increase with the size of the collection too.
3) The methods that rely on relevance feedback information only select among the terms of a few retrieved documents. In contrast, the method described selects additional search terms out of the entire term set. Therefore, the number of additional search terms is usually larger.
4) We point out two reasons why early attempts in automatic query expansion failed to improve the retrieval effectiveness. The results shown in the paper primarily indicate how dangerous threshold values are.

A commercial database with millions of documents contains a great deal of terms. The construction of a similarity thesaurus could therefore be computationally expensive. Hence, the construction algorithm as well as the storing and the accessing of such a similarity thesaurus has to be studied carefully. With our test collections, all the terms of the collection were taken into account. Since frequent terms tend to discriminate poorly between relevant and non-relevant documents [Pea 91, Sal 75], they could be omitted from the similarity thesaurus of a large collection. However, which terms to ignore has to be studied carefully. The retrieval effectiveness could be even improved by omitting some of the poor discriminators.

The improvement of the retrieval effectiveness by using our approach is around 20-30%. This is less than the one reported when using relevance feedback information.

However, the relevance feedback methods depend heavily on the kind and quality of the user relevance information. In addition, this information is hard to get.

The advantage of our method is that it is fully automatic. Furthermore, our method can be used in the first run in an IR system when no relevance information is yet available. In case relevance information is available, feedback techniques could be introduced to retrieve even more relevant documents.

In our future research we will concentrate on a sensible combination of our novel query expansion method and relevance feedback mechanisms as well as on applying these techniques on commercial - and therefore voluminous - databases.

References

[Cro 90] Crouch, C.J., An approach to the automatic construction of global thesauri, *Information Processing & Management*, 26(5): 629-40, 1990.

[Cro 92] Crouch, C.J., Yong, B., Experiments in automatic statistical thesaurus construction, *SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval*, Copenhagen, Denmark, 77-87, June 1992.

[Dee 90] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman R., Indexing by latent semantic analysis, *J. of the ASIS*, 41(6): 391-407, 1990.

[Ekm 92] Ekmekcioglu, F.C., Robertson, A.M., Willett, P., Effectiveness of query expansion in ranked-output document retrieval systems, *J. of Information Science*, 18(2): 139-47, 1992.

[Fre 91] Frei, H.P., Schäuble, P., Determining the effectiveness of retrieval algorithms, *Information Processing & Management,* 27(2-3): 153-164, 1991.

[Gre 92] Grefenstette, G., Use of syntactic context to produce term association lists for retrieval, *SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval*, Copenhagen, Denmark, 89-97, June 1992.

[Han 92] Hancock-Beaulieu, M., Query expansion: advances in research in on-line catalogues, *J. of Information Science*, 18(2): 99-103, 1992.

[Har 92] Harman, D., Relevance feedback revisited, *SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval*, Copenhagen, Denmark, 1-10, June 1992.

[Les 69] Lesk, M.E., Word-word association in document retrieval systems, *American Documentation*, 20(1): 27-38, 1969.

[Min 72] Minker, J., Wilson, G.A., Zimmerman, B.H., An evaluation of query expansion by the addition of clustered terms for a document retrieval system, *Information Storage and Retrieval*, 8(6): 329-48, 1972.

[Pea 91] Peat, H.J., Willett, P., The limitations of term co-occurrence data for query expansion in document retrieval systems, *J. of the ASIS*, 42(5): 378-83, 1991.

[Qiu 92] Qiu, Y., ISIR: an integrated system for information retrieval, *Proc. 14th IR Colloquium, British Computer Society*, Lancaster, 1992.

[Rug 92] Ruge, G., Experiments on linguistically-based term associations, *Information Processing & Management*, 28(3): 317-32, 1992.

[Sal 71] Salton, G., Experiments in automatic thesaurus construction for information retrieval, *Information Processing 71,* 1: 115-123, 1971.

[Sal 75] Salton, G., Yang, C.S., Yu, C.T., A theory of term importance in automatic text analysis, *J. of the ASIS*, 26(1): 33-44, 1975.

[Sal 80] Salton, G., Automatic term class construction using relevance-a summary of work in automatic pseudoclassification, *Information Processing & Management*, 16(1): 1-15, 1980.

[Sal 88] Salton, G., Buckly, C., Term weighting approaches in automatic text retrieval, *Information Processing & Management,* 24(5): 513-523, 1988.

[Sal 90] Salton, G., Buckley, C.: Improving Retrieval Performance by Relevance Feedback. *J. of the ASIS*, 41(4): 288-297, 1990.

[Sch 92] Schäuble, P., Knaus, D., The various roles of information structures, *16. Jahrestagung der Gesellschaft für Klassifikation*, Dortmund, 1992.

[Sme 83] Smeaton, A.F., van Rijsbergen, C.J., The retrieval effects of query expansion on a feedback document retrieval system, *The Computer Journal*, 26(3): 239-46, 1983.

[Spa 71] Sparck-Jones, K., Barber, E.B., What makes an automatic keyword classification effective? *J. of the ASIS*, 18: 166-175, 1971.

[Spa 91] Sparck-Jones, K., Notes and references on early classification work. *SIGIR Forum,* 25(1): 10-17, 1991.

[Wad 88] Wade, S.J., Willett, P., INSTRUCT: a teaching package for experimental methods in information retrieval. III. Browsing, clustering and query expansion, *Program*, 22(1): 44-61, 1988.

[Won 87] Wong, S.K.M., Ziarko, W., Raghavan, V.V., Wong, P.C.N., On modeling of information retrieval concepts in vector spaces, *ACM TODS*, 12(2): 299-321, 1987.