Improving the retrieval effectiveness by a similarity thesaurus

Improving the Retrieval Effectiveness by a Similarity Thesaurus

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

Hans-Peter Frei
UBILAB
Union Bank of Switzerland
CH-8021 Zurich, Switzerland

Abstract

A novel information structure and its use for query expansion is presented. The information structure, called a similarity thesaurus, consists of term-term similarities that are based on how the terms of a collection "are indexed" by the documents. In this way, the similarity thesaurus reflects domain knowledge about the collection from which it is constructed. It is used to select and weight additional query terms when expanding an existing query. This is in contrast to conventional query expansion methods as the similarity between candidate terms and the concept of the entire query is taken into account. Experiments on test collections show that the retrieval effectiveness is considerably higher when this method is applied. That this concept-based query expansion model can also be used to produce better results in large-scale operational IR environments is the final aspiration.

Contents

1 Introduction

The growing amount of information and the need for quick access to it, slowly changes the significance of information retrieval (IR) systems. In order to turn IR systems into more useful tools for both the professional and the general user, one usually tends to enrich them with "more intelligence". Often, such "intelligence" aims at improving the retrieval effectiveness by integrating information structures, such as conventional thesauri providing terms for the query formulation (Qiu, 1992). Since it is difficult to build thesauri in a systematic way, they are hardly ever consistent (Schäuble, 1989). In addition, such classical thesauri are rather expensive because they have to be constructed manually.

Therefore, many researchers attempted to construct thesauri automatically. This, however, would have needed well-defined algorithms that are lacking as mentioned above. For this reason, research concentrated on the automatic construction of alternative information structures:

- Term classifications (Lesk, 1969; Sparck-Jones, 1971) and algebras used in the generalized vector space model (Wong *et al.*, 1987) are based on statistical data.
- Linguistic knowledge is used to identify relations between terms (Grefenstette, 1992; Ruge, 1992).
- User relevance information is used to build up pseudo-thesauri (Salton, 1980) and maximum spanning trees (Smeaton and van Rijsbergen, 1983).

There are several ways to employ these kinds of information structures for improving the retrieval effectiveness. The easiest is automatic query expansion (or modification) that has been explored for nearly three decades. The idea was to obtain additional relevant documents through queries that were expanded. The similarities between terms are first calculated based on the association hypothesis and then used to classify terms by setting a similarity threshold value (Lesk, 1969; Sparck-Jones, 1971; Minker *et al.*, 1972). 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. However, this kind of automatic query expansion has not been very successful. 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 (Minker *et al.*, 1972; Peat and Willett, 1991).

Because of the debatable success of such automatically built information structures, some researchers have attempted to generate term relations on the basis of linguistic knowledge and co-occurrence statistics (Grefenstette, 1992; Ruge, 1992). A grammar and a dictionary are used to extract for each term t a list of terms consisting 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 the most similar terms to the ones of the query. This method produces only slightly better results than using the original queries (Grefenstette, 1992).

When relevance information is available, it can be used to construct a global information structure, such as a pseudo-thesaurus (Salton, 1980) or a maximum spanning tree (Smeaton and van Rijsbergen, 1983). A query is expanded by means of such a global information structure which – of course – depends heavily on the user relevance information. The experiments in Smeaton and van Rijsbergen (1983) did not yield a consistent performance improvement. On the other hand, the direct use of relevance information by simply extracting terms from relevant documents without constructing an information structure, is proved to be effective in interactive information retrieval (Salton and Buckley, 1990). However, this approach cannot be used when no relevance information is available or when no relevant documents are found in previous queries.

Research of the last years have shown that systems employing automatically constructed information structures did not live up to expectations: The retrieval effectiveness of expanded queries was not greater than – often even less than – the effectiveness of the original queries.

The information structure that we call a similarity thesaurus (Qiu and Frei, 1993) is a term-term similarity matrix whose entries are arrived at through the "indexing" of the terms of a collection by the documents of the collection. Therefore, relationships between the terms are based on the probabilities of the documents representing the meanings of the terms. In this way, the similarity thesaurus reflects the domain knowledge of the particular collection from which it is constructed. It does not attempt to reflect a general domain of discourse. When we employ the similarity thesaurus for information retrieval, we expand a query by adding those terms that are most similar to the *concept* of the query, rather than selecting terms that are only similar to individual query terms.

In this paper, we first present a method that allows construction of a similarity thesaurus from a given document collection. We believe that these methods can also be used for large commercial databases containing millions of documents and terms. In section 2, the construction algorithm for a similarity thesaurus is presented. Subsequently, the update process is described in section 3. Section 4 is devoted to two concept-based query expansion methods that employ a similarity thesaurus for identifying query concepts. After describing our test setting, some results of experiments carried out with two standard test collections are presented in section 5. Finally, we conclude with the main findings and point out further research and possible applications of the methods presented.

2 Constructing a Similarity Thesaurus

2.1 Similarity Thesaurus

Let us start out with a conventional term-term similarity matrix C. Given is a document-term matrix A:

$$A = \begin{array}{c} \\ d_1 \\ d_2 \\ \vdots \\ d_m \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & \cdots & t_n \end{array} \\ \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{pmatrix} \end{array} \tag{1}$$

where the ${d_{ik}}'$s indicate the weights of terms t_k in documents d_i. Then a conventional term-term similarity matrix C, which is often the base of term classification, can be computed as

$$C = A^T A \tag{2}$$

(Salton and McGill, 1983, p. 79). That is, a conventional term-term matrix is built on the basis of term co-occurrence as well as the weights of the terms representing the documents.

An alternative idea is to base the relationship between terms on the probability of the document representing the term. It is to be noted that this probability is not identical to the probability of a term representing the concept of a document.

Like the matrix C, we can obtain a matrix that consists of term-term similarities. However, it is based on how the terms of the collection "are indexed" by the documents. We call it a similarity thesaurus and show that it 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.

B denotes a term-document matrix, but is clearly not a transpose of the matrix A. That is, $B \neq A^T$:

$$B = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_n \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & \cdots & d_m \end{array} \\ \begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ t_{21} & t_{22} & \cdots & t_{2m} \\ \vdots & \vdots & & \vdots \\ t_{n1} & t_{n2} & \cdots & t_{nm} \end{pmatrix} \end{array} \tag{3}$$

The t_{ik}'s signify feature weights of the indexing features (documents) d_k with respect to the items (terms) t_i. The following factors are taken into account when we estimate these weights:

- 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.

- The larger the number of occurrences of an indexing feature (document) in an item (term), the higher the probability of the document representing the meaning of the term.

- If a term is described by many documents, it may either express a broad concept or have many different meanings. In the latter case, the proportion of the meanings of the term described by a specific document is more likely to be small. More precisely, term weights should be normalized by the length of the term vector.

Therefore, we define the feature weights t_{ik} by the feature frequency (ff), the inverse item frequency (iif), and the maximum feature frequency $(maxff)$ as follows.

$$t_{ik} \quad := \quad \frac{(0.5 + 0.5 * \frac{ff(d_k,t_i)}{maxff(t_i)}) * iif(d_k)}{\sqrt{\sum_{d_j \in t_i} ((0.5 + 0.5 * \frac{ff(d_j,t_i)}{maxff(t_i)}) * iif(d_j))^2}} \tag{4}$$

where,

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

$iif(d_k) = \log(\frac{n}{|d_k|})$ is the inverse item frequency of feature d_k; n 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 different terms appearing in the document d_k. We call it the length of the document.

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

With these definitions, we define the similarity between terms represented by vectors $\vec{t_i} = (t_{i1}, t_{i2}, \cdots, t_{im})^T$ by using a similarity measure such as the simple scalar vector product:

$$SIM(t_i, t_j) \quad := \quad \vec{t_i} \cdot \vec{t_j} = \sum_{d_k \in t_i \cap t_j} t_{ik} * t_{jk} \tag{5}$$

It is to be noted that this similarity is identical to the cosine measure because the term vectors are normalized.

The similarity thesaurus is constructed by determining the similarities of all the term pairs. The result is a symmetric matrix S:

$$S \quad = \quad BB^T \tag{6}$$

Obviously, S is derived from the document collection itself, rather than from the subjective realm of human experience. It reveals the various facets of meaning that the terms have in the particular document collection. Therefore, only the terms that may be useful in finding relevant documents are included.

2.2 Reducing the Number of Terms

Commercial databases usually contain a large number of documents and terms. The constructing, storing, and accessing of a similarity thesaurus containing all the terms would therefore be very expensive. This is why we look for ways to omit the less significant terms. However, the question is what features determine whether a term is useful for retrieval purposes and should therefore be included in a similarity thesaurus?

There are several statistical approaches for choosing useful terms from a document collection. Salton, Yang, and Yu (Salton *et al.*, 1975) introduced the notion of discrimination value. The discrimination value of an indexing term is determined by a change in the inter-document similarity caused by removing the indexing term under discussion from the document collection. The discrimination value model assumes that increasing the average similarity between documents will lead to better retrieval effectiveness when the term is not removed but used. Therefore, a term with a high discrimination value is – according to this model – a desirable indexing term. We call it, consequently, a good term.

The Poisson distribution is a discrete random distribution. The Poisson distribution criterion for selecting indexing terms is based on work in (Bookstein and Swanson, 1974; Srinivasan, 1990; Margulis, 1992). It works under the assumption that the distribution of a good term in the collection is different from the distribution of a poor term. According to this criterion, good terms are those that do not behave according to M Poisson distributions.

Terms that happen to be distributed according to a Poisson distribution are not informative about the documents in which they occur, and hence should be omitted from a similarity thesaurus.

However, it is computationally expensive to compute both the discrimination value and the Poisson distributions of a term. Fortunately, there is a strong correlation between Poisson distributions, document frequencies, and discrimination values (Srinivasan, 1990). In particular, empirical studies showed that the discrimination value of a term is strongly correlated with its document frequency (Salton *et al.*, 1975). Srinivasan also pointed out that Two-Poisson distributions work along the same lines as document frequency properties. For this reason, we chose the document frequency as the criterion for selecting terms to be included in our similarity thesauri.

Since frequently occurring terms tend to discriminate poorly between relevant and non-relevant documents, such terms may be omitted from a similarity thesaurus. Margulis (1992) also found that over 70% of the frequently occurring terms behave according to the M Poisson distributions. In addition, there are many terms statistically associated with frequently occurring terms. By omitting these terms, the number of non-zero entries in the similarity thesaurus can be reduced by 10 to 20% depending on the collection (see experiments in Section 5).

Furthermore, infrequently occurring terms should also be omitted from a similarity thesaurus in order to avoid random associations and reduce the matrix dimension. First of all, we do not have enough knowledge about infrequently occurring terms. With the term-term similarity measure we use, it is easy for infrequently occurring terms to have a high similarity when they occur in the same documents. As a result, there will be a great deal of correlation of infrequently occurring terms, purely because of random associations. Such random associations may seriously decrease the retrieval effectiveness. Secondly, there are normally many infrequently occurring terms in a data collection. For example, the commercial INSPEC database contains roughly 4.5 million documents and 1.5 million terms. Of those terms, roughly half occur in one document only.

In summary, document frequency properties are used to determine whether a term is good or bad. As a rule, very high and very low frequency terms are regarded as bad terms.

2.3 Algorithm for Constructing a Similarity Thesaurus

It is clear that the straightforward algorithm (see also formula (5)) is of no practical use when there are millions of documents and terms involved. For disk access alone, this algorithm has a complexity of $O(n^2)$, where n is the number of terms in the collection. A more efficient algorithm is therefore needed.

Croft pointed out that the vast majority of the similarity values are zero and suggested an algorithm which avoids calculation of the similarity values which will be zero (Croft, 1977). For each document, his algorithm reads descriptions of terms occuring in this document and calculates the similarity between those terms. There is to be less disk access than that required by the straightforward approach. However, his algorithm calculates the similarity every time the terms co-occur in a document, hence, if the terms co-occur frequently the same similarity will be calculated many times (Harding and Willett, 1980). Willett (1981) carried Croft's work further and suggested an algorithm that eliminates all redundant calculations of similarities. However, only a few similarity measures can be applied and complex weighting is quite difficult to be implemented using Willett's algorithm.

Similar to Croft's algorithm, we do not calculate the similarity values which will be zero, and redundant calculations of the similarities are also eliminated. When we use the weighting scheme described in formula (4), we need to know the vector length of the term, i.e., the normalization coefficient of the term. However, this normalization coefficient is known only after processing all the documents of the collection. Hence, we separate the calculation of the normalization coefficients from that of the similarities. That is, we use the weighting scheme described in formula (4), but without cosine normalization, to estimate the unnormalized weight of document d_k in term t_i. It is denoted as t'_{ik}:

$$t'_{ik} \quad := \quad (0.5 + 0.5 * \frac{ff(d_k, t_i)}{maxff(t_i)}) * iif(d_k) \tag{7}$$

For each document, we compute the within-document similarity of each pair $(t_i,\ t_j)$ of terms that occur in the document: $t'_{ik} * t'_{jk}$, and at the same time, the within-document normalization coefficient of terms t_i in the document: ${t'_{ik}}^2$. The global, unnormalized similarity $sim(t_i, t_j)$ of a term pair $(t_i,\ t_j)$ is the sum of all the within-document similarities of that term pair. The global normalization coefficient $c(t_i)$ of a term t_i is the sum of all the within-document normalization coefficients of the term. With these arrangements, the similarity between terms to be included in a similarity thesaurus can be computed as

$$SIM(t_i, t_j) \quad := \quad \frac{sim(t_i, t_j)}{\sqrt{c(t_i) * c(t_j)}} \tag{8}$$

Since $sim(t_i, t_j) = sim(t_j, t_i)\ \forall i, j$, we only need to calculate the upper (or lower) triangular part of the matrix sim. An algorithm for constructing the similarity thesaurus SIM of a document collection is shown in Figure 1. Note that the dimension of the resulting matrix SIM is much smaller than the dimension of the matrix sim because the similarity thesaurus contains only good terms.

The time complexity of the algorithm is $O(m)$ for disk access and $O(m * |d|^2 + n^2)$ for computation, where m is the number of documents, n is the number of terms, and $|d|$ is the average number of different terms in the documents of the collection, i.e., the average "document length" as defined above. It can be easily seen that this algorithm is much more efficient than the straightforward approach. For example, it only takes a few minutes to construct a similarity thesaurus for the CACM test collection using the algorithm presented in Figure 1, whereas it needs several hours using the straightforward approach. Our algorithm is also more efficient than Croft's algorithm (Croft, 1977) because there are no redundant calculations of similarities between terms and less disk access is needed.

In a dynamic document collection, terms that were not chosen as good terms may become good terms after adding new documents. Likewise, the contrary could be the case, namely, good terms could become bad ones. If we do not calculate the similarities between all the terms of the document collection, the update of the similarity thesaurus can not be performed without rescaling some old documents in the collection. In order to solve this problem, we calculate the unnormalized similarities ($sim(t_i, t_j)$ including normalization coefficients $c(t_i)$) between all the terms of the collection and store them in a help file, and only good term-term

```
initialization;

(* calculation of within-document term-term similarities and normalization coefficients *)
for every document $d_k$ do
    read description of the document $d_k$ from disk;
    for every term $t_i$ contained in the document $d_k$ do
        $c(t_i) := c(t_i) + {t'_{ik}}^2$;
        for every term $t_j$ contained in the document $d_k$ and $i \leq j$ do
            $sim(t_i, t_j) := sim(t_i, t_j) + t'_{ik} * t'_{jk}$
        end
    end
end;

(* normalization of term-term similarities *)
for every good term $t_i$ of the collection do
    for every good term $t_j$ of the collection and $i \leq j$ and $sim(t_i, t_j) > 0$ do
        $SIM(t_i, t_j) := SIM(t_j, t_i) := \frac{sim(t_i,t_j)}{\sqrt{c(t_i)*c(t_j)}}$
    end
end.
```

Figure 1: Algorithm for constructing a similarity thesaurus.

pairs are kept in the similarity thesaurus. The advantages of this approach are: the access to the resulting similarity thesaurus is fast because it is small and the update can be done without rescaling the entire collection. The disadvantage is that we have to keep the usually large help file.

3 Updating a Similarity Thesaurus

In a dynamic document collection, relationships between terms may change after adding documents to, or removing documents from the collection. In this case, the similarity thesaurus of the collection may need to be updated. In this section, we present an algorithm for updating a similarity thesaurus and criteria for determining when such an update becomes necessary.

3.1 Algorithm for Updating a Similarity Thesaurus

In the literature there are several approaches proposed for evaluating the term-term similarities based on statistical data (Croft, 1977; Willett, 1981). However, there are very few maintenance algorithms. It is not yet clear how term associations can be updated efficiently.

As mentioned in Section 2.3, $sim(t_i, t_j)$ and $c(t_i)$ for all the terms of the collection are kept in a help file. When documents are added to, or removed from the collection, we can recalculate $sim(t_i, t_j)$ and $c(t_i)$ for the newly arrived or the removed documents by simply taking into account only the terms occurring in these added or removed documents:

$$
\begin{aligned}
sim(t_i,t_j) &:= sim(t_i,t_j) + \sum_{added\ documents\ d_k} t'_{ik} * t'_{jk} - \sum_{removed\ documents\ d_k} t'_{ik} * t'_{jk} \quad (9) \\
c(t_i) &:= c(t_i) + \sum_{added\ documents\ d_k} {t'_{ik}}^2 - \sum_{removed\ documents\ d_k} {t'_{ik}}^2 \quad (10)
\end{aligned}
$$

The similarity values contained in the similarity thesaurus can now be updated by using formula (8). It is to be noted that not all the entries of the similarity thesaurus need to be modified. It suffices to update the entries corresponding to any term occurring in the added or removed documents. Other entries will remain unchanged, obviously a much more efficient way than reconstructing the entire similarity thesaurus.

However, when documents are added to, or removed from the collection, the previously estimated probabilities of documents representing the meanings of terms, t'_{ik} of formula (7), may need to be modified for the following reasons:

- The inverse item frequency iif of a document changes when the number of terms in the collection is changed, although the number of terms remains nearly constant compared to the number of documents of the collection (Hüther, 1990).
- The maximum feature frequency $maxff$ of a term may change especially when many documents with many indexing terms are added or removed.

If these factors are not taken into account, an updated similarity thesaurus may not be identical to the similarity thesaurus built up from scratch using the entire collection. For this reason, we try to make the weighting consistent. It is to be noted that the inverse item frequency represents the relative importance of a feature (document) with respect to other features in the collection. When the number of items (terms) is changed, the relative importance of the feature can still remain constant. Therefore, we adjust the inverse item frequency to

$$iif(d_k) \quad := \quad \frac{1}{\log(|d_k|+1)} \qquad (11)$$

and adapt the weighting scheme to

$$t'_{ik} \quad := \quad ff(d_k,t_i) * \frac{1}{\log(|d_k|+1)} \ . \qquad (12)$$

Similar to the document weighting scheme described in (4) and (7), we also take into account the length of a document. Furthermore, this new weighting scheme is independent of the number of documents and terms of a collection. Therefore, adding new documents into the document collection would not affect the previous estimation of the document weights.

3.2 Factors Affecting a Similarity Thesaurus

Although the update process can be performed without rescaling the entire document collection, it still takes time and may hinder other retrieval transactions. Adding a few documents to a document collection with millions of documents hardly changes the relationship between terms. However, when a great deal of important documents are added, the similarity thesaurus of the document collection should be updated. Let us examine the factors that influence a similarity thesaurus.

Let $SIM_{old}(t_i, t_j)$ be the old similarities between the terms, $SIM_{new}(t_i, t_j)$ be the new similarities between the terms after adding or removing some documents, and $SIM_{part}(t_i, t_j)$ be the partial similarities between terms based only on the newly arrived or removed documents. If we can find the relationships between $SIM_{old}(t_i, t_j)$, $SIM_{new}(t_i, t_j)$, and $SIM_{part}(t_i, t_j)$, we can then decide under which conditions the similarity thesaurus needs to be updated. For the sake of simplicity, we consider only the case when new documents are added to the collection.

Let

$$\begin{aligned} \alpha &= \frac{\sum_{added\ documents\ d_k} {t'_{ik}}^2}{c(t_i)} && (13) \\ \beta &= \frac{\sum_{added\ documents\ d_k} {t'_{jk}}^2}{c(t_j)} && (14) \\ \gamma &= \frac{\sum_{added\ documents\ d_k} t'_{ik} * t'_{jk}}{sim(t_i, t_j)} && (15) \end{aligned}$$

where, $sim(t_i, t_j)$, $c(t_i)$, and $c(t_j)$ are the previously estimated unnormalized similarity between terms and the normalization coefficients before the collection is modified.

Then, it can easily be shown that

$$\begin{aligned} SIM_{new}(t_i, t_j) &= \frac{1 + \gamma}{\sqrt{(1 + \alpha) * (1 + \beta)}} * SIM_{old}(t_i, t_j) && (16) \\ SIM_{part}(t_i, t_j) &= \frac{\gamma}{\sqrt{\alpha * \beta}} * SIM_{old}(t_i, t_j) && (17) \end{aligned}$$

Now let us look at the equations (16) and (17) more closely, and study some special cases:

1. $\alpha = 0$ and $\beta = 0$. In this case, γ is also equal to 0. Therefore, $SIM_{new}(t_i, t_j) = SIM_{old}(t_i, t_j)$, and $SIM_{part}(t_i, t_j) = 0$.

   The terms t_i and t_j do not occur in the added documents. Thus, the similarities between them remain unchanged.

2. $\alpha = \beta = \gamma > 0$. Then $SIM_{new}(t_i, t_j) = SIM_{old}(t_i, t_j) = SIM_{part}(t_i, t_j)$.

   The increase of occurrence of the terms in the added documents is proportional to the increase of co-occurrence of the terms. Therefore, the similarity values between the terms remain constant. This is often the case when the topic of the added documents is already represented in the collection.

3. $\alpha \ll 1$, $\beta \ll 1$, and $\gamma \ll 1$. Then $SIM_{new}(t_i, t_j) \approx SIM_{old}(t_i, t_j)$.

   Most terms contained in the added documents are frequent and only a few documents are added. In this case, the similarities between those terms remain similar. This is often the case when the ideas contained in the added documents have been widely discussed in the collection, i.e., no new topics are contained in the added documents.

4. $\alpha \approx 0$, $\beta > 0$, and $\gamma \approx 0$, In this case, $SIM_{new}(t_i, t_j) \approx \frac{1}{\sqrt{1+\beta}} * SIM_{old}(t_i, t_j) < SIM_{old}(t_i, t_j)$, and $SIM_{part}(t_i, t_j) \approx 0$. Likewise, if $\beta \approx 0$, $\alpha > 0$, and $\gamma \approx 0$, then $SIM_{new}(t_i, t_j) \approx \frac{1}{\sqrt{1+\alpha}} * SIM_{old}(t_i, t_j) < SIM_{old}(t_i, t_j)$, and $SIM_{part}(t_i, t_j) \approx 0$.

   Some terms become more frequent, while others do not. That is, some topics gain more significance than others. The similarities between these two kinds of terms should be reduced.

5. $\gamma > \alpha$ and $\gamma > \beta$. Then $SIM_{new}(t_i, t_j) > SIM_{old}(t_i, t_j)$.

   Terms co-occur more frequently than before. The similarity values between these terms become greater. This happens when a new interdisciplinary topic is introduced into the collection.

6. $\alpha > \gamma$ and $\beta > \gamma$. Then $SIM_{new}(t_i, t_j) < SIM_{old}(t_i, t_j)$.

   Terms occur more frequently than before. However, they seldom co-occur in the added documents. The similarity values between such terms become smaller. This happens, for example, when one topic breaks up into two sub-topics.

In summary, adding new documents may change the relationships between terms. However, the kind of topics that are added decide whether the change in the relationships between terms is significant. Typically, when a completely new topic is introduced, the similarity thesaurus should be updated. When, on the other hand, a relatively small number of documents is added without introducing new topics, the old similarity thesaurus is still useful. The values α, β, and γ can be computed on the ground of the added documents and the previously estimated $sim(t_i, t_j)$, $c(t_i)$, and $c(t_j)$. They serve to formulate criteria as to whether the similarity thesaurus no longer reflects the extended collection and must be updated.

4 Concept-Based Query Expansion

4.1 A Concept-Based Query Expansion Model

One method to improve retrieval effectiveness of an IR system, is to take into account the domain knowledge in order to determine an appropriate interpretation of a user's query. As pointed out earlier, a similarity thesaurus reflects the domain knowledge of a document collection. There are two different ways to exploit a similarity thesaurus: either the user can browse through the similarity thesaurus to find search terms, or the similarity thesaurus can be used for automatic query expansion. Since the first approach depends heavily on the user, we focus on the second one.

As already mentioned, most attempts at automatically expanding queries failed to improve 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 (Peat and Willett, 1991). 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 1 that with most methods, terms are selected that are strongly related to *one* of the query terms. The known 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 a text from one natural language 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 to a *single term* of the query. In what follows we explain how 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.

A query q is represented by a vector $\vec{q} = (q_1, q_2, ..., q_n)^T$ 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.

Since the similarity thesaurus expresses the similarity between the terms of the collection in the document vector space (DVS) (defined by the documents of the collection), we map the vector $\vec{q}$ from space TVS into a vector in space DVS. In 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 2.1. In other words, the concept expressed by the term t_i in the query has an importance of $q_i * \vec{t_i}$ 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:

$$\vec{q_c} \quad := \quad \sum_{t_i \in q} q_i * \vec{t_i} \tag{18}$$

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

$$simqt(q,t) \quad := \quad \vec{q_c} \cdot \vec{t} = (\sum_{t_i \in q} q_i * \vec{t_i}) \cdot \vec{t} = \sum_{t_i \in q} q_i * \vec{t_i} \cdot \vec{t}$$

Where $\vec{t_i} \cdot \vec{t}$ is the similarity between two terms defined in formula (5):

$$simqt(q,t) \quad := \quad \sum_{t_i \in q} q_i * SIM(t_i, t) \tag{19}$$

It is to be noted that the values of $SIM(t_i, t)$ are the entries of our similarity thesaurus and therefore are pre-computed. All the terms in the collection can now be ranked according to their *simqt* value with respect to the query q. The terms t with high $simqt(q,t)$ 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 $simqt(q,t)$:

$$weight_a(q,t) \quad := \quad \frac{simqt(q,t)}{\sum_{t_i \in q} q_i} \tag{20}$$

where, $0 \leq weight_a(q,t) \leq 1$.

As mentioned, we choose only those terms that are ranked in the top positions by the *simqt* function to expand the query. The reason for only choosing the top ranked terms as opposed to setting a weight threshold is for the sake of efficiency. 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

$$\vec{q}_e := (q_{e1}, \ldots, q_{en})^T \tag{21}$$

where,

$$q_{ei} \quad := \quad \begin{cases} weight_a(q,t_i) & t_i \text{ belongs to the top } r \text{ ranked terms} \\ 0 & \text{otherwise} \end{cases}$$

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

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

$$\vec{q}_{expanded} = \vec{q} + \vec{q}_e \tag{22}$$

After this expansion process, new terms may have been added to the original query and/or the weight of an original query term may have been modified had the term belonged to the top ranked terms. The expanded query $q_{expanded}$ is then used to retrieve documents.

The important point of this method is that additional search terms are selected dynamically when a query is submitted. 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.

4.2 Similarity Thesaurus and the Generalized Vector Space Model

In the generalized vector space model (GVSM) (Wong *et al.*, 1987), an atomic expression, or a minterm, in n literals (terms), $t_1, t_2, \cdots, t_n$, is a conjunction in which each literal t_i appears exactly once, either complemented or uncomplemented. Hence, there are 2^n possible minterms in n literals. In a document collection indexed by these n terms, only some minterms are active. According to the definition of the minterm, one can easily derive that each active minterm corresponds to a document class in which all the documents are indexed by the same terms that are uncomplemented.

Hence, terms are in fact represented by document classes in the GVSM. A term-class (term-minterm) matrix is denoted by M:

$$M = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_n \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & \cdots & c_l \end{array} \\ \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1l} \\ c_{21} & c_{22} & \cdots & c_{2l} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nl} \end{pmatrix} \end{array} \qquad (23)$$

where, c_{ij} indicates the weight of term t_i in document class c_j. c_{ij} is the sum of the weights of the term t_i in documents that belong to the document class c_j. When each document constitutes a document class, the matrix M is a transpose of the document-term matrix A defined in Section 2.1: $M = A^T$

In the GVSM, documents and queries are then expressed as the vector sum of the associated term vectors in the vector space of document classes.

$$\vec{d'} = (\vec{d}^T M)^T = M^T \vec{d} \qquad (24)$$

$$\vec{q'} = (\vec{q}^T M)^T = M^T \vec{q} \qquad (25)$$

When the scalar vector product is used to evaluate the similarity between a document and a query, the similarity between the document d and the query q is:

$$RSV(q,d) = \vec{q'} \cdot \vec{d'} = \vec{q}^T M M^T \vec{d} = \vec{q}^T X \vec{d} \qquad (26)$$

where, $X = MM^T$ is a term-term association matrix. From this equation, we can see that the GVSM is identical to the VSM when term dependence is taken into account. However, there is a difference between the GVSM and the VSM, when the cosine measure is used. The similarity between d and q is normalized by the vector length of vectors $\vec{d'}$ and $\vec{q'}$ in the GVSM, whereas by the vector length of the original vectors $\vec{d}$ and $\vec{q}$ in the VSM.

Now let us explain why the GVSM is also a query expansion model. We set r, the number of terms to be added or modified in weight, to n, the total number of terms. In this case, the query q is expanded by q_e containing all the n terms. Furthermore, let us consider an arbitrary document $\vec{d} = (d_1, d_2, \cdots, d_n)^T$ in space TVS where the d_i's signify term weights for this particular document. Then, the similarity between q_e and d is:

$$\begin{aligned} \vec{q_e} \cdot \vec{d} &= \sum_{t_j \in d} q_{ej} * d_j \\ &= \frac{1}{\sum_{t_i \in q} q_i} \sum_{t_j \in d} \sum_{t_i \in q} q_i * SIM(t_i, t_j) * d_j \\ &= a \; \vec{q}^{\,T} S \vec{d} \qquad (27) \end{aligned}$$

$$\textit{where} \qquad a = \frac{1}{\sum_{t_i \in q} q_i}$$

S is the similarity thesaurus defined in (6).

Since the constant a 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 (27) is analogous to the similarity indicated in formula (26) for the GVSM. This means that both the method proposed in this paper and the GVSM go 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 different ways, although both methods use co-occurrence data. We construct a similarity thesaurus as described in Section 2.1. In the GVSM, term association is based on how terms express document classes in which all the documents are indexed by the same terms. However, in some document collections such as MED, each document class contains only one document. In this case, the term-term association matrix X used in the GVSM is identical to the conventional term-term similarity matrix C described in Section 2.1. In the CACM collection, documents contained in the same document classes are in fact identical but numbered differently. Hence, it is questionable whether the document relationship is taken into account to evaluate the similarity between terms in the GVSM, when automatic indexing is applied. Secondly, the GVSM includes all the terms in the expansion and "uses" q_e for ranking documents as shown in formula (27). Yet, in our approach, we expand the query only by a few carefully chosen terms and use $q_{expanded}$. As we have shown in (Qiu and Frei, 1993), expanding a query by most similar terms performs much better than expanding by all the similar terms.

There are also some practical issues about these two methods. In the GVSM, the documents are represented by document classes. This means, document classification needs to be performed before we can obtain the required kind of description for the documents. Furthermore, the description file of the documents is normally quite large. It is larger than a similarity thesaurus. For example, the description file in the CACM test collection is 78 MByte long, whereas the similarity thesaurus containing all the terms of the collection is only 27 MByte. During the query process, a query must be represented by document classes through accessing the term-class matrix M. Hence, the term-class matrix also needs to be stored. As a result, the efficiency of a retrieval system using the GVSM would be lower than that based on our concept-based query expansion model.

4.3 An Extended Concept-Based Query Expansion Model

In the concept-based query expansion model presented above, the concept of a query is represented by the centroid of the query which is calculated from all the search terms. As there may be noise in the query, the centroid does not necessarily represent the query concept correctly. In other words, some of the search terms may not be "to the point" and should not be used when the query concept is constructed. Now the problem is, how to determine whether a search term is "to the point". It could be done by means of relevance information which, however, is normally hard to obtain. For this reason, we concentrate on procedures which do not need information from the user. Our reasoning is based on the following common assumptions in IR:

- Top ranked documents are more likely to be relevant to the user query.
- The distribution of search terms in relevant documents is different from the distribution in non-relevant documents.

- Terms that are similar to a term in a relevant document are more likely to occur in relevant documents, and terms that are similar to a term in a non-relevant document are more likely to occur in non-relevant documents.

Based on these assumptions, we stipulate that query terms which do not occur as indexing terms in the top ranked documents are more likely to be non-relevant for the particular query. We call them bad query terms. Terms that are similar to the non-relevant search terms are more likely to be non-relevant and therefore should not be used to expand the query. Hence, we propose to extend the previous query expansion model. The extension includes a query concept that is represented only by the relevant search terms appearing in the top ranked documents. In analogy to our earlier definition, we call them *good* search terms. Here is the method in more detail:

1. Rank documents in decreasing order of retrieval status value using the original query q. Original search terms that occur in the top ranked documents are considered good search terms and others are considered bad search terms. Let G_q and B_q denote the set of good search terms and the set of bad search terms for the query q, so that $G_q \cup B_q = q$ and $G_q \cap B_q = \phi$.

2. Use the similarity thesaurus of the collection to evaluate the similarity $simqt(q,t)$ between a candidate term t and the query q.

$$\begin{aligned} simqt(q,t) &:= \sum_{t_i \in q} q_i * SIM(t_i,t) - \sum_{t_i \in B_q} q_i * SIM(t_i,t) \\ &= \sum_{t_i \in G_q} q_i * SIM(t_i,t) \end{aligned} \tag{28}$$

3. Rank the candidate terms according to their $simqt(q,t)$ values and choose the top ranked terms as additional search terms for the query q.

4. Estimate the weights $weight_a(q,t)$ of the additional search terms t.

$$weight_a(q,t) := \frac{simqt(q,t)}{\sum_{t_i \in G_q} q_i} \tag{29}$$

5. The original and the additional search terms together form the expanded query that is, consequently, used to retrieve documents. Also, they might form the basis for another expansion step.

5 Evaluation of a Similarity Thesaurus

In order to justify an automatically generated information structure, its usefulness has to be proved. In addition, we would like to answer the question: how is a possible increase in the retrieval effectiveness related to the effort for constructing the similarity thesaurus? To answer these questions, we performed many experiments on the standard test collections CACM and MED. The characteristics of these two test collections are described in (Qiu and Frei, 1993). The aim was to compare the retrieval effectiveness of our methods based on similarity thesauri with the effectiveness of the standard retrieval method which uses original queries and the retrieval method based on the generalized vector space model.

In order to evaluate similarities between queries and documents, term weights in both the documents and the queries were determined according to the "term frequency – inverse document frequency" tfc weighting scheme (Salton and Buckley, 1988), see also formula (4). Then, the two concept-based query expansion methods described in Section 4 were used to expand or modify queries. 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.

5.1 Constructing the Similarity Thesaurus

The construction method described in Section 2 was used to build the similarity thesauri because the test collections we used are static. As mentioned before, the size of a similarity thesaurus is reduced by omitting high and low frequency terms. This can be verified from the figures shown in Table 1. Here the size of a similarity thesaurus is measured by the number of term–term pairs with a non-zero similarity. Table 1 shows the size of the whole similarity thesauri containing all the terms, the reduced similarity thesauri containing terms that have a document frequency greater than or equal to 2 ($df(t) \geq 2$), the reduced ones containing terms that have a document frequency less than or equal to 10% of the number of documents ($df(t) \leq n/10$), and the reduced ones omitting infrequently and frequently occurring terms ($2 \leq df(t) \leq n/10$). The size of the reduced similarity thesaurus omitting infrequently and frequently occurring terms is 76% of that of the entire similarity thesaurus for the CACM test collection, and 63% for MED. Table 1 also shows the ratio of the number of non-zero entries in a similarity thesaurus to the number of all the possible entries. One can see that over 97% of term-term pairs in a document collection have a similarity of 0.

Criterion	Test Collection			
	CACM		MED	
all terms	1,520,686	3.0%	2,347,460	3.1%
$df(t) \geq 2$	1,257,537	2.5%	1,777,632	2.4%
$df(t) \leq n/10$	1,383,915	2.7%	1,966,048	2.6%
$2 \leq df(t) \leq n/10$	1,153,749	2.3%	1,484,742	2.0%

Table 1: The size of the similarity thesauri using different term selection criteria.

In addition, the construction of the reduced similarity thesauri also becomes efficient after the term selection. Table 2 shows the CPU time on a SPARC workstation for constructing the whole similarity thesauri containing all the terms as opposed to the reduced similarity thesauri omitting frequently and infrequently occurring terms.

Criterion	Test Collection	
	CACM	MED
all terms	9.9	14.6
$2 \leq df(t) \leq n/10$	7.9	9.4

Table 2: CPU time in minutes for constructing similarity thesauri.

5.2 Concept-based Query Expansion Model vs. the VSM and the GVSM

After having constructed similarity thesauri for the two document collections, we applied our concept-based query expansion method described in Section 4.1. Terms that are most similar to the concepts of queries are weighted and added to the original queries according to formulae (19), (20), (21), and (22). Figure 2 shows the improvement in the retrieval quality of the expanded queries over the original queries. The results indicate that our automatic query expansion model yields a considerable improvement in the retrieval effectiveness over the standard VSM. In Figure 2, we also show how the number of additional search terms affects the retrieval effectiveness. It can easily be seen that the improvement by the expanded queries increases when the number of the additional search terms increases. When the number of additional search terms is between 100 and 200, the improvement remains nearly constant. We do not show the retrieval performance of the expanded queries when the number of additional search terms is larger than 200. However, we found that the improvement decreases, but the expanded queries still perform better than the original queries as shown in (Qiu and Frei, 1993). The results shown in Figure 2 also indicate that expanding a query by roughly 100 top ranked terms seems to be the safe way to go. The improvement is 23% in CACM and 18% in MED, in this case.

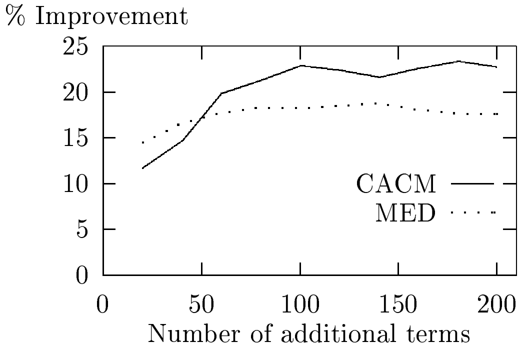


Figure 2: Retrieval improvement of the concept-based query expansion over the VSM.

The GVSM can produce significant improvement in the retrieval effectiveness over the standard VSM as reported by Wong *et al.* (1987). Here, we compare our concept-based query expansion model with the GVSM. The cosine measure is used to evaluate the similarity between documents and queries when the GVSM is applied. The results are shown in Figure 3. They indicate that our model performs better than the GVSM. It goes without saying that the improvement is smaller than the one shown in Figure 2, because the GVSM gives already better results than the VSM.

5.3 Similarity Thesaurus vs. Conventional Term-Term Similarity Matrix

As mentioned in Section 2.1, our similarity thesaurus is based on the probability of the documents representing the meanings of the terms, whereas a conventional term-term similarity matrix is based on the probability of the terms representing the documents. They both reflect

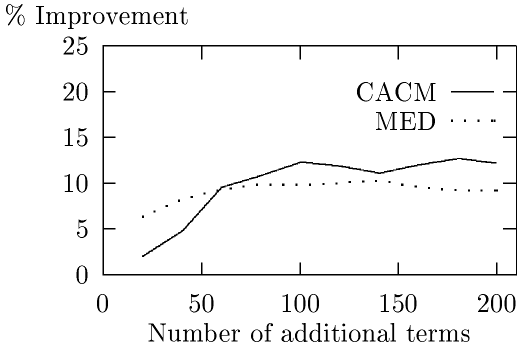


Figure 3: Retrieval improvement of the concept-based query expansion over the GVSM.

domain knowledge and can be used to expand the original queries by means of the concept-based query expansion model. When the conventional term-term matrix is used, the values of $SIM(t_i, t)$ in formula (19) are replaced by the entries of the conventional matrix. Figure 4 shows the improvement of our expansion model based on similarity thesauri or conventional term-term similarity matrices over the VSM. In the CACM collection about computer science, the concept-based query expansion based on the similarity thesaurus gives much better results than the one based on the conventional matrix. In the MED collection about medicine, they perform equally effectively. When the number of additional terms gets to be larger than 60, the method based on the similarity thesaurus produces slightly better results than the other one. This can be explained by the fact that the terms used in medicine are normally well defined which is not always the case in computer science.

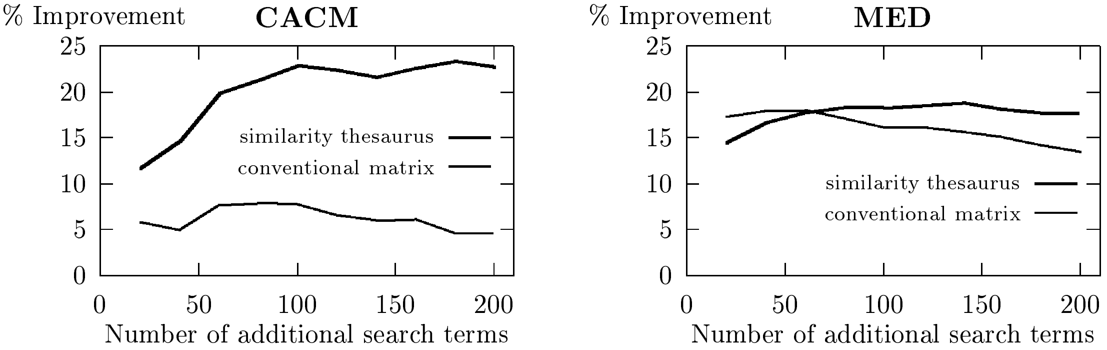
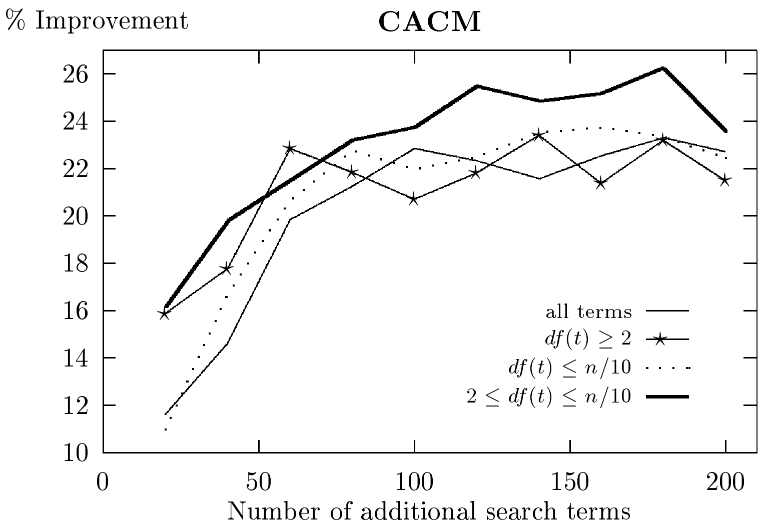


Figure 4: Similarity thesaurus vs. conventional term-term similarity matrix.

5.4 Term Selection

In this section we compare the retrieval performance when similarity thesauri using different term selection criteria are applied. Figure 5 shows the difference in the retrieval qualities



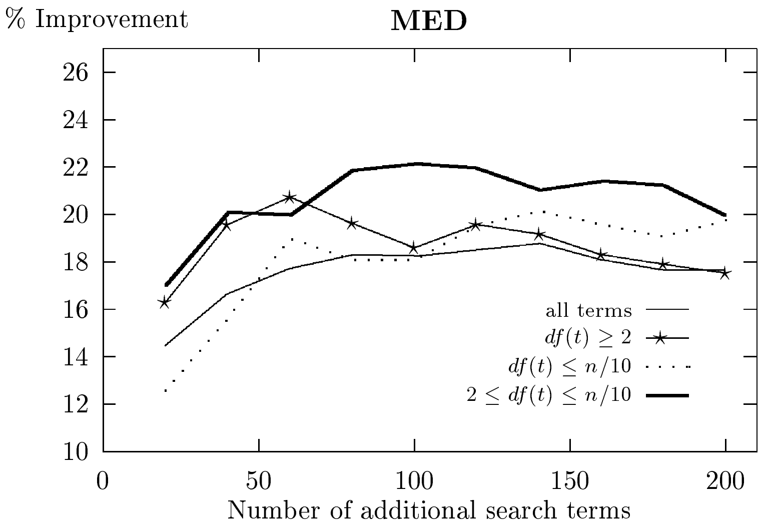


Figure 5: Retrieval improvement using various similarity thesauri.

between the original queries and the expanded queries using the entire similarity thesauri (all the terms) and the following reduced similarity thesauri: $df(t) \geq 2$, $df(t) \leq n/10$, and $2 \leq df(t) \leq n/10$. These results indicate that the similarity thesaurus $2 \leq df(t) \leq n/10$ performs best. In addition, it seems to be crucial to omit infrequently occurring terms when we expand queries with a small number of additional terms because of random associations; whereas, if the number of additional terms is large, it seems to be more important to omit frequently occurring terms.

5.5 Comparison of two Concept-based Query Expansion Methods

Finally, we describe experiments that compare the retrieval effectiveness of the original concept-based query expansion described in Section 4.1 and its extended version described in Section 4.3. Using the extended model, we considered the original search terms appearing in the top 10 ranked documents as good search terms. Then formulae (28) and (29) were used to select additional search terms. The original queries were then expanded according to formulae (21) and (22). The entire similarity thesauri of CACM and MED were used. Figure 6 shows the improvement in the retrieval effectiveness of the original concept-based query expansion and the extended version over the VSM. The results indicate that the extended version produces consistently better retrieval than the original query expansion.

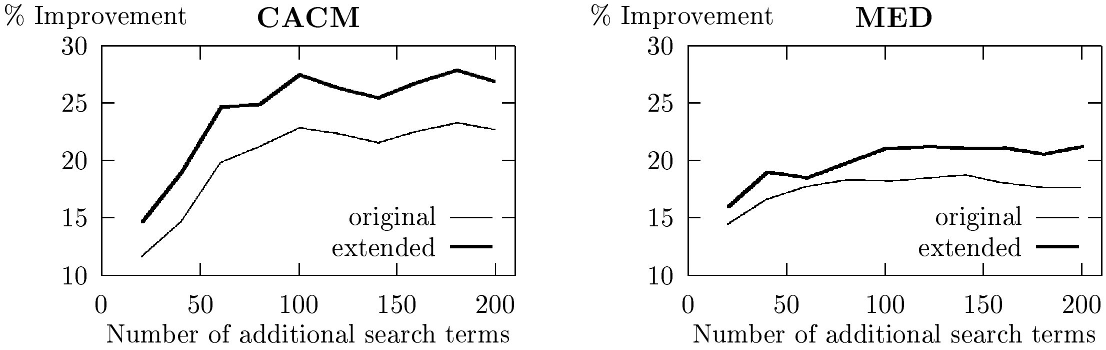


Figure 6: The extended vs. the original concept-based query expansion model, using entire similarity thesauri

Figure 7 shows the difference in the retrieval effectiveness between the original concept-based query expansion and the extended version when the reduced similarity thesauri ($2 \leq df(t) \leq n/10$) were used. Same as above, the results indicate that the extended model is consistently better than the original model. As we have already shown in Figure 5, retrieval when using the reduced similarity thesauri is better than when using the entire similarity thesauri. Therefore, the extended concept-based query expansion model and the reduced similarity thesauri seem to be the best combination.

The main results of this portion of the study are the following:

- The concept-based query expansion model based on a similarity thesaurus can result in a

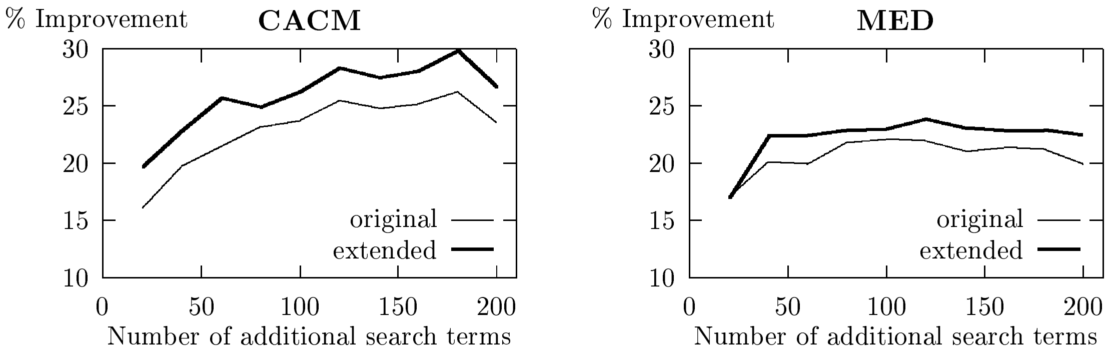


Figure 7: The extended vs. the original concept-based query expansion model, using reduced similarity thesauri.

significant improvement in the retrieval effectiveness when compared with the standard vector space model (VSM) and the generalized vector space model (GVSM).

- A similarity thesaurus serves the concept-based query expansion better than a conventional term-term similarity matrix.
- The retrieval effectiveness using the reduced similarity thesaurus, from which both infrequently and frequently occurring terms are omitted, is better than using the entire similarity thesaurus containing all the terms of the collection.
- The extended concept-based query expansion model produces better retrieval results than the original model, independent of the similarity thesaurus used.

6 Conclusions

In this paper we present two algorithms for constructing and updating a similarity thesaurus for a large document collection. Using our algorithms, a similarity thesaurus can be updated without rescaling the entire document collection. In addition, we also discuss the factors determining when a similarity thesaurus needs to be updated.

Many terms of a document collection can be omitted from its similarity thesaurus. We use the properties of document frequency to decide whether terms are included in the similarity thesaurus or not. The reduced similarity thesaurus excludes infrequently and frequently occurring terms. After the selection of terms, the number of terms and the size of the similarity thesaurus become quite small. As a result, the construction, maintenance, and accessing of the similarity thesaurus become easy and fast. It may come as a surprise, but it turns out that the retrieval effectiveness using the reduced similarity thesaurus is even better than the effectiveness using the entire similarity thesaurus containing all the terms of the document collection.

We also experimented with other criteria for term selection, such as the discrimination value and a combination of discrimination value and document frequency. However, the results obtained using these criteria were inferior to those obtained using the simple document frequency criterion.

In order to use a similarity thesaurus for improving the retrieval effectiveness, we present a concept-based query expansion model. This model is primarily concerned with the two most important issues of query expansion, namely, the selection and the weighting of additional search terms. The term selection and weighting rely on the overall similarity between the query concept and the terms of the collection, rather than on the similarity between an individual query term and the terms of the collection. The experiments carried out on test collections show that consistent improvement in retrieval effectiveness can be expected.

As there may be noise in a query of the user, we carried the work further and extended the query expansion model. First, we determine which search terms of a query are good search terms. The concept of the query is only determined by using these good terms. Second, we choose the terms that are most similar to the query concept as additional search terms. Third, this expanded query is run against the document collection. The results of some experiments on test collections show that this model can produce even better retrieval performance than the original model.

It is to be noted that the extended concept-based query expansion model is entirely based on statistical data. If relevance information is available, it could be that the set of good search terms is a set of terms that occur in the retrieved *relevant* documents. This may be a sensible way of integrating our approach and relevance feedback mechanisms.

A weighted retrieval algorithm has been built directly into the commercial database service Data-Star of RadioSuisse (Frei and Qiu, 1993). A similarity thesaurus for the commercial INSPEC database of around 4.5 million documents and 1.3 million terms has also been constructed (Keller, 1994). The implementation of a retrieval system using the *"real-life"* similarity thesaurus is under way. As soon as this system is available, we are going to carry out some experiments on the INSPEC collection. We hope to be able to show that the concept-based query expansion model can be used to produce better results in an operational IR environment.

References

Bookstein, A. and Swanson, D. (1974). Probabilistic models for automatic indexing. *Journal of the ASIS*, 25(5), 312–318.

Croft, W. B. (1977). Clustering large files of documents using the single-link method. *Journal of the ASIS*, 28(6), 341–344.

Frei, H.-P. and Qiu, Y. (1993). Effectiveness of weighted searching in an operational IR environment. In *Information Retrieval '93: von der Modellierung zur Anwendung*, pages 41–54. Universitätsverlag Konstanz.

Grefenstette, G. (1992). Use of syntactic context to produce term association lists for retrieval. In *Proc. of the 15th International ACM SIGIR Conference on R & D in Information Retrieval*, pages 89–97. ACM Press, New York.

Harding, A. F. and Willett, P. (1980). Indexing exhaustivity and the computation of similarity matrices. *Journal of the ASIS*, 31(4), 298–300.

Hüther, H. (1990). On the interrelationship of dictionary size and completeness. In *Proc. of the 13th International ACM SIGIR Conference on R & D in Information Retrieval*, pages 313–325. ACM Press, New York.

Keller, A. (1994). *Similarity Thesaurus: Data-Star Implementations-Bericht.* Technical report, RadioSuisse, Bern, Switzerland.

Lesk, M. E. (1969). Word-word association in document retrieval systems. *American Documentation*, 20(1), 27–38.

Margulis, E. L. (1992). N-Poisson document modelling. In *Proc. of the 15th International ACM SIGIR Conference on R & D in Information Retrieval*, pages 177–189. ACM Press, New York.

Minker, J., Wilson, G. A., and Zimmerman, B. H. (1972). An evaluation of query expansion by the addition of clustered terms for a document retrieval system. *Information Storage & Retrieval*, 8(6), 329–348.

Peat, H. J. and Willett, P. (1991). The limitations of term co-occurrence data for query expansion in document retrieval systems. *Journal of the ASIS*, 42(5), 378–383.

Qiu, Y. (1992). ISIR: An integrated system for information retrieval. In *Proc. of the 14th Information Retrieval Colloquium*, pages 55–66. Springer-Verlag, London.

Qiu, Y. and Frei, H.-P. (1993). Concept based query expansion. In *Proc. of the 16th International ACM SIGIR Conference on R & D in Information Retrieval*, pages 160–169. ACM Press, New York.

Ruge, G. (1992). Experiments on linguistically-based term associations. *Information Processing & Management*, 28(3), 317–332.

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

Salton, G. and Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), 513–523.

Salton, G. and Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the ASIS*, 41(4), 288–297.

Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval.* McGraw-Hill, New York.

Salton, G., Yang, C. S., and Yu, C. T. (1975). A theory of term importance in automatic text analysis. *Journal of the ASIS*, 26(1), 33–44.

Schäuble, P. (1989). *Information Retrieval Based on Information Structures.* Ph.D. thesis, Swiss Federal Institute of Technology (ETH) Zurich. Informatik-Dissertationen, Nr. 15, VdF-Verlag, Zürich.

Smeaton, A. F. and van Rijsbergen, C. J. (1983). The retrieval effects of query expansion on a feedback document retrieval system. *The Computer Journal*, 26(3), 239–246.

Sparck-Jones, K. (1971). *Automatic Keyword Classification for Information Retrieval.* Butterworths, London.

Srinivasan, P. (1990). A comparison of two-poisson, inverse document frequency and discrimination value models of document representation. *Information Processing & Management*, 26(2), 269–278.

Willett, P. (1981). A fast procedure for the calculation of similarity coefficients in automatic classification. *Information Processing & Management*, 17(2), 53–60.

Wong, S. K. M., Ziarko, W., Raghavan, V. V., and Wong, P. C. N. (1987). On modeling of information retrieval concepts in vector spaces. *ACM Transactions on Database Systems*, 12(2), 299–321.