INTEGRATING LOGICAL OPERATORS IN QUERY EXPANSION IN VECTOR SPACE MODEL

Jian-Yun Nie, Fuman Jin
DIRO, Université de Montréal
C.P. 6128, succursale Centre-ville, Montreal
Quebec, H3C 3J7 Canada

Abstract

Query expansion is an effective way to extend the coverage of retrieval to the related documents. Various approaches have been proposed and many of them are based on the vector space model. The expansion process consists of simply adding expansion terms into the original vector. In this paper we argue that this simple expansion method can bias the focus of the original query, because the expanded terms add additional emphasis to the original term. Instead of adding expansion terms into the vector, we propose to combine them with the original terms by means of the logical OR operator. In this way, the expansion terms are considered as alternatives to the original terms, and the focus of the whole query remain unchanged. Our experiments on a TREC collection show that this expansion approach is more appropriate than the simple addition approach.

Key words

Logical operator, vector space model, query expansion, mutual information, Wordnet

1. INTRODUCTION

One of the problems in IR is that queries are often a very partial specification of the information need. This is especially the case in the Web environment. Query expansion has often been suggested as a solution to this

problem. Query expansion tries to extend the coverage of the retrieval to the related documents that do not necessarily contain the same words as the query. In the previous studies on query expansion, much emphasis has been put on the selection of expansion terms, either through a manually constructed thesaurus [7], a statistical measure based on co-occurrences or a combination of them [3].

The previous studies have shown various results with query expansion. Voorhees showed that using Wordnet [4] for query expansion in this way, the IR effectiveness is rather decreased. In the work of Mandala et al., the relations stored in Wordnet are combined with Mutual Information (MI) and another similarity measure based on syntactic dependency. They show that such a combination can greatly improve IR effectiveness.

However, in all these studies, it has been taken as granted that the expanded terms should be added into the original vector directly. For example, if the term A is a related term to B, than a query containing B can be expanded by adding A into the query vector. No question has been raised concerning the appropriate way to integrate expansion terms into the query vector.

In this study, we propose a different way to integrate expansion terms into the query vector. Our basic idea is that an expansion term represents an alternative expression of the original term rather than an additional expression. Therefore, it should be combined with the original term by the logical relation OR.

In the following section, we will further motivate our approach. In section 3, this approach is compared with the direct addition approach on a TREC test collection. Our experiments show that our expansion approach is more appropriate.

2. EXPANSION TERMS AS ALTERNATIVES

2.1 Principle of query expansion

Let us firs give an example of the direct addition approach to query expansion. Suppose that an original query contains two terms A0 and B0 describing two aspects of the query. Suppose that through a resource (either a manual thesaurus or a statistical calculation), we know that A1, A2, A3 and A4 are strongly related to A0, and no related term is known for B0. If all the related terms are added into the query vector through the direct query expansion process, then the new vector will contain 6 terms or 6 non-empty dimensions: <A0, B0, A1, A2, A3, A4>. 5 of them are related to the aspect

A and one to the aspect B. We have to notice that all the dimensions in this vector are treated in the same way independently. For example, the inner product similarity will simply sum up the similarities of a document to every dimension. We can easily see that a document will have 5 more chances to satisfy the aspect A than the aspect B.

Now let us observe the following two cases.

1. The expansion terms are true synonyms. They have little chance to co-occur with the original query term and with each other.

   In this case, one may consider that no document will simultaneously contain more than one of the terms A0, A1, …, A4. Then in practice, the expanded query vector is equivalent to a set of alternative vectors: (A0, B0), (A1, B0), …, (A4, B0). In this particular case, no particular problem arises from the direct addition approach.

2. The expansion terms are related terms that may co-occur with the original term and with each other.

   This case occurs much more often in practice. In fact, few studies have limited query expansion to the true synonymy relation. This is due to the fact that the true synonymy relation can only expand a query with a very limited number of additional terms, and it is difficult to guarantee that all the expansion terms are true synonyms (unless they have been selected rigorously and manually). Even if we use resources such as Wordnet, we cannot guarantee that no synonym will co-occur with the original term.

   More often, query expansion exploits statistical relationships based on word co-occurrences. The expansion terms are more related ones than true synonyms. They are highly probable to co-occur in the same document as the original query term. If we return to our example, this means that a document will likely contain A0, A1, … and A4. If all the occurrences of these terms are considered and their similarities are summed up into the correspondence of the document to the query, then the A aspect will take much higher importance than the aspect B that has not been expanded. In other words, in the expanded query vector, the aspect that has been more expanded will be implicitly attributed higher importance. This potentially creates a bias to the original information need. For example, suppose that the original query contains two terms with equal importance: A0 = “computer” and B0 = “graphics”; and suppose the expansion terms are A1 = “calculator”, A2 = “data processor”, A3 = “electronic computer” and A4 = “information”, the expanded query will contain the following terms:

```
(computer, graphics, calculator,
 data_processor, electronic_computer,
 information_processing).
```

One can clearly see that much higher emphasis has been put on "computer" than on "graphics".

The above example shows the possible bias that query expansion may create. The bias is due to the unbalanced numbers of expansion terms for each query term. Even if one tries to take the number of expansion terms into account (e.g. by dividing the weight of each expansion term by the number of the expansion terms), this problem cannot be solved appropriately. In fact, such a division can solve the unbalance problem in the second case shown above, but will create additional problem in the first case: if a document only contains one of the many true synonyms, its similarity to this aspect is not fully considered.

The fundamental problem in the above approach is the following inconsistence: On one hand, when query is expanded, all he expansion terms are considered to be dependent to the original query term. On the other hand, when the expanded query is evaluated, all the dimensions are considered independently. This contradiction is the basic problem of the simple addition approach.

A radical solution lies in the creation of a more appropriate relationship between the original query term and the expanded terms in the expanded vector. That is, when the expansion terms are added because they are related to an original term, this relation should also be taken into account during the evaluation process.

Intuitively, when a query term is expanded with another term, the later should be considered as an alternative expression of the former. Therefore, the most natural logical operator to connect them is OR. We therefore suggest a vector representation that integrate the logical OR relation.

For our earlier example, the expanded query would rather be:

```
(computer OR calculator OR data_processor OR
    electronic_computer OR  information_processing,
 graphics).
```

The first composite element represents one single aspect of this vector. It may be expressed in 5 alternative ways. The whole vector contains two aspects, as in the original query. The relative importance between them is unchanged. The only change we bring into the new vector is the way to evaluate the first dimension.

In this expansion method, we still assume independency between the aspects specified in the original query (i.e. "computer" and "graphics"). This is not always correct. Indeed, "computer graphics" is different from "computer" and "graphics". However, the recognition of the relationship between the terms in a query is a difficult problem that has not been solved

correctly. In most of the cases, it is reasonable to assume that the terms in a query represent different aspects. This is particularly true for short queries.

A more general form of expanded query would contain a weight associated to each term, such as follows:

```
(computer^w1 OR calculator^w11 OR data_processor^w12 OR
    electronic_computer^w13 OR information_processng^w14,
 graphics^w2).
```

The weights w1 and w2 are those in the original query that can be determined either by the user or with a tf*idf weighting schema. We will come back later on the problem of determining the weights w11, …, w14 for the expansion terms. In the next section, let us assume that such an expanded query has been created, and examine the problem of its evaluation.

2.2 Evaluation of an expanded query

Two problems have to be dealt with for the evaluation of an expanded query:
1. How to evaluate the OR relation?
2. How to integrate this logical operator in a vector space model?

Because of the uncertainty of query evaluation, the classical evaluation of OR cannot be used here. For the evaluation of OR in our case, we can use the evaluations proposed in the fuzzy set theory. The two following forms of evaluation of OR are among the most often used (where W is a fuzzy function which corresponds in our case to the similarity of a document to a query A OR B or part of it):

$$W(A \text{ OR } B) = \max(W(A), W(B)) \quad (1)$$

$$W(A \text{ OR } B) = W(A) + W(B) - W(A)*W(B) \quad (2)$$

The first formula is often used in studies of fuzzy set theory. However, this formula only considers the dominant element among the two elements. This does not seem reasonable for IR. The second formula takes into account the contributions of both elements in all the cases. This is more reasonable for IR.

The choice of an appropriate formula can be done though experiments. In our preliminary tests described in Section 3, it will turn out that the second formula performs better than the first one.

The next question is how to integrate these evaluations into the calculation of similarity in vector space model. There are to solutions:

- We can transform the expanded query into a logical combination (with OR) of vectors. For example, the expanded query (A OR B, C) is equivalent to (A, C) OR (B, C).
- We can also directly evaluate each dimension of the expanded query. For a dimension corresponding to a logical expression, a singe similarity value will be calculated. The calculation of this value will take into account the correspondence of all the alternative terms with the document and the logical relation between them.

The first implementation is inefficient. In fact, as the term C in the example may repeat in several elements in the disjunction, it has to be evaluated several times. Therefore, we implement the evaluation in the second way.

Our implementation also makes use of inverted file. We choose to use inner product as the formula of similarity. The following algorithm is used to obtain the resulting set of weighted documents for an expanded query Exp_query:

```
eval(Exp_query)
 list = ∅;
 For each dimension_i in Exp_query:
   doc_list_i = eval_d(dimension_i);
        list = list ⊕ doc_list_i;
 return list;

eval_d(dimension)
 list = ∅;
 for each disjuncted term t_j^wj in dimension:
        list_j = list of documents containing t_j,
   the weight of
                each document is multiplied by
        w_j;
   list = list ∇ list_j;
 return list;
```

In this algorithm the operator $\oplus$ denotes the union of two lists of weighted documents by summing up the weights of the common elements in the two lists. This is the operator corresponding to the classical evaluation in vector space model. The operator ∇ is a combination of two lists of weighted documents by combining the weights of common elements according to one of the evaluation formulas for OR (formulas (1) and (2)).

It is interesting to notice that the above algorithm is a generalization of the classical inner product evaluation of vector space model. To reproduce

the classical evaluation, it is sufficient to use the following evaluation for OR:

$$W(A \text{ OR } B) = W(A) + W(B) \qquad (3)$$

2.3 Determining related terms and their weights

Another important question in query expansion is how to determine the related terms to be used and their weights. In several studies (e.g. [7]) Wordnet is used to provide related terms. It turns out that using these terms to expand queries do not help in IR evaluation. Our preliminary tests also confirmed this fact. In [3], Wordnet is combined with statistical measures between words based on their co-occurrences. It is shown that this combination greatly improves IR effectiveness. However, as all the measures have been tested in combination, it is not clear how each of the factors between Wordnet and statistical measures contributed in the evaluations.

In our experiments, we compared the use of Wordnet with that of statistical relations based on Mutual Information (MI). The latter performed much better. Therefore, in this paper, we only report the use of MI. Mutual information is defined as follows:

$$MI(x, y) = P(x, y) \times \log \frac{P(x, y)}{P(X) \times P(y)}$$

where x and y are terms, $P(x)$ is the probability of x determined by

$P(x)$ = #doc. containing x / #doc. of the collection;

$P(x,y)$ is the probability that x and y co-occur in a document:

$P(x,y)$ = #doc. containing x and y / #doc. of the collection.

The values of MI vary greatly from term to term. In order to make MI more comparable, we use the following normalized MI:

$$NMI(x, y) = MI(x, y) / \max_{y_i} MI(x, y_i)$$

where x is an original query term, and y_i any possible expansion term.

As the expansion terms determined by MI are usually only related to some degree to the original information need, it is reasonable to assign a lower weight to them than to the original query terms. Therefore, an additional coefficient C ($0 \le C \le 1$) is added to decrease the importance of expansion terms. The weight of an expansion term y for the original query term x is then determined as follows:

$$W_x(y) = C * NMI(x, y)$$

The value of C is set manually according to experiments. Several values of C will be tested in our experiments described in the next section.

3. EXPERIMENTS

We tested the methods with the AP collection used in TREC 7 CLIR track. This collection contains 242,818 documents in English. 28 queries are provided with standard answers. We use both tites and descriptions in our experiments. A modified version of Smart [1] is used for our experiments.

3.1 Experiments with Wordnet

We first used Wordnet to determine the related terms. We only use the synonymy relation implied in Wordnet synsets. The weight of a related term is assigned a uniform value. Several values have been tested, but the value 0.2 produced the best results. In this case, the direct expansion with formula (3) results in an improvement of 1.2% in average precision over the baseline method without expansion. The expansion with logical OR – formula (2) – results in an improvement of 1.5%. Both improvements are not significant. In addition, these are the best cases. In the other cases, we often obtain degradations. These results are compatible with he tests of Voorhees [7, 8]. In the experiments of [8], it is shown that even with a manually selection of the expansion terms from Wordnet, the retrieval effectiveness cannot be improved. A possible conclusion of this is that Wordnet is not a suitable resource for query expansion in IR.

3.2 Expansion with Mutual Information

In this section, we determine the expansion terms by Mutual Information (MI) calculated from term co-occurrences, instead of Wordnet.

We notice that terms having a marginal MI with original query terms are usually noise that are not related to the query. They happen to co-occur with query terms in some documents. It is better to eliminate them in query expansion. Therefore, we also impose a fixed number N of expansion terms for each query term.

The following table shows the results we have obtained so far with the three expansion methods. The numbers are percentages of improvement with query expansion over the standard vector evaluation without query expansion.

Table 1. Experimental results with query expansion

		Number of expansion terms *N*									
formula	**C**	2	5	10	15	20	25	30	50	60	90
(3)	0.25	1.4	7.2	9.2	**10.1**	2.8	3.3	3.2	3.9	2.4	2.6
(2)	0.05	1.0	4.0	6.2	7.2	7.4	7.8	8.0	9.9	9.9	**10.6**
	0.1	1.6	6.4	9.8	11.3	10.4	11.3	11.3	12.5	12.3	**12.6**
	0.2	1.8	7.8	11.3	**12.8**	11.9	12.0	10.5	9.9	9.7	8.0
	0.25	1.5	7.8	10.7	11.3	**11.4**	10.6	9.3	2.3	1.1	-3.7
	0.3	1.9	7.8	**10.9**	10.5	10.2	3.8	8.0	-2.4	4.2	-7.6
(1)	0.05	0.7	1.3	**1.4**	1.3	1.1	1.3	1.2	1.1	1.0	0.9
	0.1	0.9	**2.7**	2.4	2.6	2.1	2.0	1.7	1.5	1.4	1.3
	0.2	0.2	3.1	**4.0**	3.2	1.8	1.5	1.2	0.5	0.2	-0.2
	0.25	0.1	3.1	**4.3**	3.2	1.3	1.0	0.6	-0.4	-1.1	-1.0
	0.3	-0.4	2.8	**4.2**	3.1	0.6	-0.1	-0.1	-0.9	-1.7	-2.3

As we can see, when a reasonable number of query expansion terms are used, query expansion usually increases IR effectiveness. The best number of expansion terms seems to be between 10 and 20 in most of the cases (this is however dependent on the formula used and *C*).

If we compare the proposed expansion method that integrates logical OR with Formula (2), with simple addition approach (Formula (3), C=0.25), we can see that the former generally performs slightly better.

If we compare the results wit different values of *C*, we can see that when *C* is small, it is better to use more expansion terms. When *C* increases, the number of expansion terms should be less. This seems coherent with our intuition: If we attribute a high importance to the expansion terms, the consideration of more terms may bring noise terms with relatively high weights in the resulting vector, which may bias the original query. Thus the number of expansion terms should be limited. When the value of *C* is small, we can allow more expansion terms because they are weighted lower. In this case, the method is also less sensitive to the number of expansion terms. For example, when *C*=0.1, we obtain god results with a wide range of number of expansion terms (from 15 to 90). Among the cases we tested, the value of 0.1 and .2 for *C* seem to produce the best results.

If we compare the two evolutions of logical OR (formulas (1) and (2)), we can see that formula (2) produces much better results. This confirms that formula (2) is more appropriate for query expansion than Formula (3).

3.3 Global MI v.s. local MI

The values of MI we used previously are calculated on the whole document collection. In several recent studies, it is shown that statistical

relationships calculated on a subset of documents is better than using global relationships.

In [2], it is shown that one can obtain significant improvements by pseudo-relevance feedback: a set of 300 terms are extracted from the top-ranked documents retrieved with the original query; these terms are added into the query vector for query expansion; the expanded query produces much better results than the original query. In a similar way, [10] also proposes a local expansion method that selects a set of expansion terms from the top-ranked documents of the first-round retrieval. However, the MI of these terms is determined on the whole document collection. Nevertheless, the expansion method used in both [2] and [12] is still the direct addition.

In order to compare the direct addition approach and the expansion with logical OR for local expansion, we conducted the following test in a similar way to [12]: from the top 30 documents determined in the first-round retrieval, we select 100 strongest terms as expansion terms. The MI for these terms is determined globally on the whole collection. These terms are used in two ways: added directly or connected by logical OR. However, in this case, it is difficult to determine which expansion term is related to which original term. It is difficult to produce the same expanded vector as before. We will try to develop a method that uses logical operators for local query expansion in the future.

4. RELATED WORK

Our approach to query expansion is different from most previous studies. In this study, we argue that an appropriate combination of the expansion terms with the original terms is an important problem to deal with in query expansion. In the previous studies, it has been taken as granted that expansion terms should be added as additional dimensions in the resulting vector (e.g. in [7] and [3]). Our preliminary results seem to support the claim that considering the expansion terms as logical alternatives is a better solution.

The work is also comparable to the tentative of [9] that tries to create more complex relationships within vectors. Wong et al. observed that the underlying independence assumption in vector representation is not reasonable. They suggest considering dependencies between dimensions in a Generalized Vector Space Model proposed. However, the method of Wong et al. suffers from the complexity problem. In practice, it is difficult to fully implement it. Out method does not suffer from this problem. It is a method that can be efficiently implemented in practice.

In fact, in our approach, we have implicitly assumed independence among original query terms. This means that each word in a query represents a different aspect of the information need. This assumption is reasonable, in particular, when queries are short. In short queries, the same aspect usually does not repeat. However, when queries are expanded, we can no longer assume that the expansion terms are also independent from the original terms and from each other among them. Our present study proposes to expand each dimension of the original query independently, and an expanded dimension is represented as a logical disjunction.

This work is also related to the Extended Boolean Model proposed by Salton et al. [6]. In this study on p_norm model, they show that the relationship between different dimensions in a vector is indeed a neutralized logical relation (i.e. no distinction between AND and OR). In our study, we propose a stronger OR relation to replace this neutralized logical relation to connect expansion terms to the original query terms.

In this study, each dimension is expanded independently. A more reasonable approach would be considering the expansion in dependence with the whole query. That is to consider other query terms as a context to help determine the most appropriate expansion terms. Such an expansion process is in the same line as the work by Qiu and Frei [5]. We will implement such an expansion process.

5. CONCLUSIONS

Query expansion is considered as an effective way to extend the coverage of a query in order to find strongly related documents. In the previous studies using vector space model, expansion terms have been usually directly added into the vector as additional dimensions. As dimensions are assumed to be independent, this simple addition approach can greatly alter the emphasis of the original query.

In this study, we proposed to use logical OR to connect the expansion terms to the original query terms. The idea is to consider expansion terms as alternative expressions of the original terms.

Our experimental tests suggest that this new expansion method is more appropriate than the simple addition approach.

We are currently undertaking additional tests. This expansion method will be compared with more other expansion methods. This will be reported later.

REFERENCES

1. Buckley, C. *Implementation of the SMART information retrieval system*, Technical report, #85-686, Cornell University, 1985.
2. Buckley, C., Salton, G., Allan, J., Singhal, A., utomatic Query Expansion Using SMART: TREC 3, NIST Special Publication 500-226, pp. 69-80, 1995.
3. Mandala, R., Tokunaga, T. Combining multiple evidence from different types of thesaurus for query expansion, in *Proceedings of ACM-SIGIR'99*, 1999, pp. 191-197.
4. Miller, G., Wordnet: an on-line lexical database, in *International Journal of Lexicography*, vol. 3, 1990.
5. Qiu, Y., Frei, H.-P. Concept based query expansion in *Proceedings of ACM-SIGIR'93*, 1993, pp. 160-169.
6. Salton, G., Fox, E. A., and Wu, H. Extended Boolean information retrieval. *Comm. of ACM*, 26(11):1022--1036, Nov. 1983.
7. Voorhees, E. M. Using Wordnet to disambiguate word senses for text retrieval. in *Proceedings of ACM-SIGIR'93*, Pittsburgh, 1993, pp. 171-180.
8. Voorhees, E. M. Query Expansion Using Lexical-Semantic Relations, in *Proceedings of ACM-SIGIR'94*, 1994, pp. 61-69.
9. Wong, S.K.M., Ziarko, W., Wong P.C.N., Generalized vector space model in information retrieval, in *Proceedings of ACM-SIGIR'95*, 1985, pp. 18-25.
10. Xu, J. and Croft, B.W., Improving the effectiveness of information retrieval with local context analysis, ACM Transactions on Information Systems, 18(1): 79-112., 2000.