Experiments with Automatic Query Formulation in the Extended Boolean Model*

Lucie Skorkovská and Pavel Ircing

University of West Bohemia, Faculty of Applied Sciences, Dept. of Cybernetics
Univerzitní 8, 306 14 Plzeň, Czech Republic
{lskorkov,ircing}@kky.zcu.cz

Abstract. This paper concentrates on experiments with automatic creation of queries from natural language topics, suitable for use in the Extended Boolean information retrieval system. Because of the lack and/or inadequacy of the available methods, we propose a new method, based on pairing terms into a binary tree structure. The results of this method are compared with the results achieved by our implementation of the known method proposed by Salton and also with the results obtained with manually created queries. All experiments were performed on the same collection that was used in the CLEF 2007 campaign.

1 Introduction

The field of information retrieval (IR) has received a significant surge of attention in the recent two decades, mainly because of the development of World Wide Web and the rapidly increasing number of documents available in electronic form. As a result, new search paradigms that offer alternatives to (still widely-used) "classic" Boolean and vector space models has been introduced, such as the method employing language models [1] or the approach based on the concept of inference networks [2] that even allows the combination of multiple different IR approaches within a single framework, and many others.

Some of those paradigms are able to accommodate structured queries — for example the inference network framework mentioned above or the extended Boolean model, on which our paper focuses. However, the possibility of structured queries is rarely being exploited and the bag-of-word approach is used instead. This statement is certainly true for the experiments reported in the Czech task of the Cross-Language Speech Retrieval track that was organized within the CLEF 2007 evaluation campaign [3], in spite of the fact that the test collection used in this task contains example information requests in the format of rather richly structured TREC-like topics.

In our paper, we will concentrate on the possible ways of automatic creation of the structured queries from the topics formulated in natural language. All the presented methods of query formulation were developed to be used within the framework of the Extended Boolean model and were tested using the same collection that was used in the

* This work was supported by the Grant Agency of Academy of Sciences of the Czech Republic, project No. 1QS101470516, and by the Ministry of Education of the Czech Republic, project No. MŠMT LC536.

V. Matoušek and P. Mautner (Eds.): TSD 2009, LNAI 5729, pp. 371–378, 2009.

aforementioned CLEF 2007 campaign. This collection consists of automatically transcribed spontaneous interviews, segmented into "documents" [1] and, as was mentioned above, the set of TREC-like topics. More information about the collection can be found in [4].

2 Extended Boolean Model

Extended Boolean model P-Norm was proposed by Salton, Fox and Wu [5] in order to take advantages of vector space retrieval model in Boolean informational retrieval. This model preserves the structure included in Boolean queries, but uses document and query term weights to compute query-document similarity, so the results can be ranked according to decreasing similarity.

There are also many other models referred to as extended Boolean models such as fuzzy set model, Waller-Kraft, Paice and Infinite-One model. All of them were designed to improve the conventional Boolean model, but Lee [6] analyzed the aspects of these models and found out that the P-Norm model is the most suitable for achieving high retrieval effectiveness.

2.1 P-Norm Model

When we think of a simple query containing only two terms, connected with Boolean operator (AND, OR), we can represent term assignment in two dimensional space, where each term is assigned different axis. Then we can say, that for the AND query the point (1,1) (the case that both terms are present) is the most desirable location. On the other hand, for the OR query is the point (0,0) the least desirable location (both terms are absent). The similarity of a document and a query is then computed as a distance between these points and the document, according to the type of the query. For AND query, the query-document similarity is represented by the complement of the distance between the document and the (1,1) point. For OR query the similarity measure is the distance between the document and the point (0,0).

The P-Norm model uses the L_p vector norm for measuring this distances. Considering a set of terms t_1, t_2, t_3,..., t_n, we can denote w_{t_i,d_j} the weight of the term t_i in the document d_j and $w_{t_i,q}$ the weight of the term t_i in a given query q. The query-document similarity measure is then defined as

$$sim\left(q_{t_1 OR t_2}, d_j\right) = \left(\frac{w_{t_1,q}^p w_{t_1,d_j}^p + w_{t_2,q}^p w_{t_2,d_j}^p}{w_{t_1,q}^p + w_{t_2,q}^p}\right)^{1/p} \tag{1}$$

$$sim\left(q_{t_1 AND t_2}, d_j\right) = 1 - \left(\frac{w_{t_1,q}^p \left(1 - w_{t_1,d_j}\right)^p + w_{t_2,q}^p \left(1 - w_{t_2,d_j}\right)^p}{w_{t_1,q}^p + w_{t_2,q}^p}\right)^{1/p} \tag{2}$$

Changing the value of the parameter p from 1 to ∞ will modify the behavior of the model from pure vector space model ($p = 1$) to conventional Boolean model ($p = \infty$,

[1] Those documents are created by sliding a fixed-size window across over a stream of text. Thus they are not in any way topically coherent and that's why we put the term in quotation marks.

term weights only binary). For the values between 1 and ∞ we obtain an intermediate system between the vector space and Boolean model.

2.2 Term Weights

Since we have no special information about document terms importance, we use the combined $tf \cdot idf$ weight as w_{t_i,d_j}. The idf – inverse document frequency part of this weight represents the importance of a term in the whole collection of documents, the tf – term frequency part represents the occurrence of a term in a concrete document.

Suppose we have N documents in the collection, n_i of them contains the term t_i, $tf_{i,j}$ represents the number of occurrences of the term t_i in the document d_j. Then the weight w_{t_i,d_j} can be defined as

$$w_{t_i,d_j} = tf_{i,j} \cdot log\frac{N}{n_i} \tag{3}$$

To ensure, that $0 \leq w_{t_i,d_j} \leq 1$, the expression (3) is used in a normalized form

$$w_{t_i,d_j} = \frac{tf_{i,j}}{tf_{max_{term\ k\ in\ d_j}}} \cdot \frac{idf_i}{idf_{max_{term\ k}}} \tag{4}$$

For the $w_{t_i,q}$ weight we use simple idf_i weight.

3 Automatic Query Formulation

As mentioned before, the topics in our collection were in natural language (see Fig. 1 for example), so we needed to transform them into Boolean queries. It is not possible to always do that manually, because of the large number of topics and terms in them, so we need to create the queries automatically. We implemented Salton's original method for automatic creation of Boolean queries described in [7] and compared it with a new method we proposed, based on pairing the terms into a binary tree. For comparison, we tried simply ORing all terms and, on the other hand, manually creating some queries.

```
<top>
<num>1166
<title>Chasidismus
<desc>Chasidové a jejich nezlomná víra
<narr>Relevantní materiál by měl vypovídat o Chasidismu
v období před holokaustem, v průběhu holokaustu a po
něm. Informace o chasidských dynastiích a založených a
zničených geografických lokalitách.
</top>
```

Fig. 1. Example of a topic from collection

3.1 Simple Method

The simplest method for the automatic creation of a Boolean query is to take all words as they are in a natural language query and connect them with OR operators (we tried also only AND). Boolean query composed in that way takes no advantage of which Boolean model can give us.

3.2 Original Salton's Method

In this section the method for automatic creation of Boolean queries proposed by Salton is described in short. As an input, we need to have the natural language query and the desired number of retrieved documents wanted by the user (m). The algorithm of creating a Boolean query can be described as:

1. Take single terms t_i from the query that do not appear on stop words list. Set n_i as the number of items indexed by term t_i.
2. Generate term pairs with AND operator ($t_i\ AND\ t_j$) and estimate the number of items retrieved as $n_i \cdot n_j/N$. N is the total number of items.
3. Generate term triples ($t_i\ AND\ t_j\ AND\ t_k$) and compute $n_i \cdot n_j \cdot n_k/N^2$. When needed generate also quadruples and estimate $n_i \cdot n_j \cdot n_k \cdot n_l/N^3$.
4. Compute the weight for single terms, term pairs, triples, quadruples as an inverse term occurrence frequency (like idf value),computed as $1 - \frac{n_i}{N}$, $1 - \frac{n_{ij}}{N}$, $1 - \frac{n_{ijk}}{N}$ or $1 - \frac{n_{ijkl}}{N}$.
5. Formulate a broad OR query from single terms with the highest weight. Estimate the number of retrieved items by that query as $\sum n_i$ over all included terms.
6. If $\sum n_i\ > m$ modify the query by iteratively removing single terms of the lowest weight and adding term pairs (combinations of removed terms) . When needed start removing term pairs and adding term triples in the same way, possibly remove triples adding quadruples.
7. Stop the process when $\sum n_i + \sum n_{ij} + \sum n_{ijk} + \sum n_{ijkl}$ is approximately equal to the number of desired items m.

3.3 New "Tree-Growing" Method

The method described above is quite complicated and computationally intensive, so we tried to find out some new simpler method, which however should achieve the same or better results. The resulting query also should have some structure to take advantage of the Boolean query processing. The idea of this method is as follows:

1. Take all terms in a query (possibly remove that terms which appear on the stop words list). We tried some different ways of choosing only some terms from the topic, the comparison of these approaches is shown in Sect. 4.
2. Sort the terms by idf value (computed as described in Sect. 2.2) in decreasing order.
3. Make pairs from the terms by connecting two adjacent terms in the list with AND operator, starting from the terms with highest weights. The result of this step is a list of pairs of terms($t_i\ AND\ t_j$). Compute the idf value of these pairs as an average of the idf weights of contained terms, i.e. $(t_i + t_j)/2$.

4. Sort the list of pairs of terms and connect these pairs further, but this time with the OR operator. Compute the idf weight of this pairs the same way as in step 3.
5. Continue this process (step 4) iteratively until you have the resulting pair (of pairs of pairs ....).

The process described above creates the binary tree structure of a query (see Fig. 2), where as leaves we have terms and as a root we have the final query.

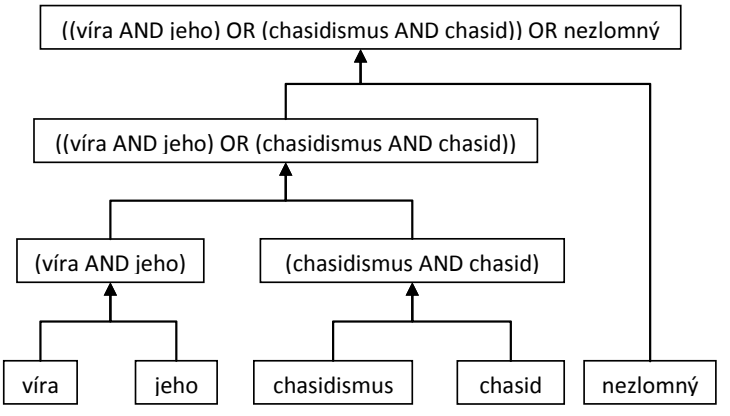

Fig. 2. Creation of the query from terms

3.4 Manual Construction

Queries were manually constructed from topics with respect to the structure of sentences and their meaning. For example, synonyms were connected with OR operator, term phrases with AND operator, then they were parenthesized according to the structure of the sentence. This way we can fully exploit the advantages of Boolean structure in the query.

4 Experiments

The segmented collection we used for experiments (as mentioned above) has 22581 "documents", methods were tested on 29 CLEF 2007 training topics. As an evaluation measure we used mean Generalized Average Precision (mGAP) as used in CLEF 2006 and 2007 Czech task[8] [3]. Both documents and topics were lemmatized in order to achieve better retrieval performance (as showed in [9]) and stop words were omitted. As terms for creating queries were used words from `<title>` (T) and `<desc>` (D) fields of the topics. The `<narr>` (N) field could not be used because then we would have too many terms for manually creating the query and even for the testing of Salton's method.

Salton's method looked promising at the first sight, but when we experimented with it we discovered many problems. This method was designed for queries with small

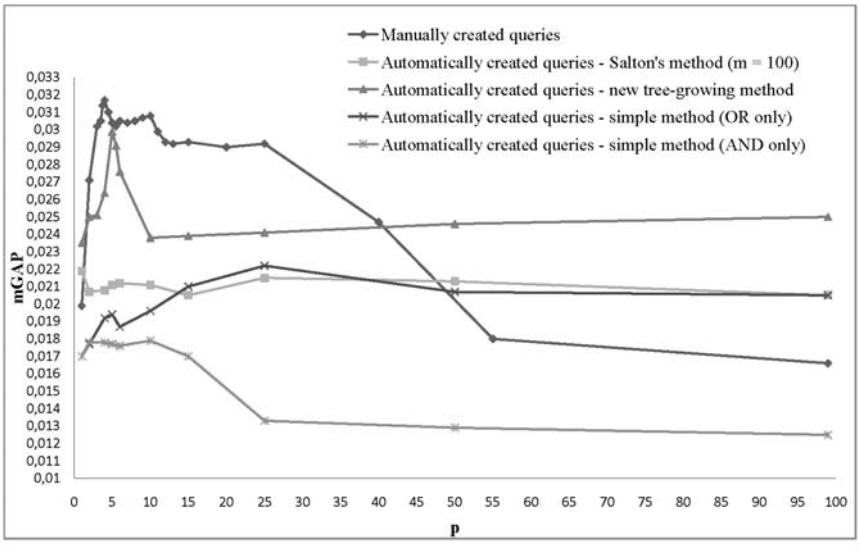

Fig. 3. Comparison of tested methods (p on the x-axis refers to the parameter of the P-Norm model as defined in Sect. 2.1)

number of terms and tested on a very small collection (1033 documents), but we have very long queries (sometimes 30 terms and even more with all TDN fields of the topic) and much bigger collection. For example the query used in original paper as example has 4 terms, which means that they can be combined into 6 pairs or 4 triples. That can be easily created and retrieved. When we take query with 10 terms, we have 45 pairs or 120 triples or 210 quadruples. With 30 terms in original query, we get 435 pairs, 4060 triples or 27405 quadruples. Creating and retrieving such a query is very time and memory consuming, so we had to use only terms with really high idf weight ($idf > 0.9$) in queries that had more than 10 terms in order to get any results at all.

The performance comparison of all tested methods is shown on Fig. 3. As expected, the simple method (both operator variants) had the worst results. This is not surprising, as it did not use any organization of the terms. All other methods had better results, although Salton's method only slightly. Salton's method was tested with different values for parameter m (30 - 1000), the value used in comparison ($m = 100$) had the best results. As can be seen, the method we proposed had better results, almost reaching the performance achieved for the manually created queries for $p = 5$ (which we suggest is the best setting – more experiments bellow). Manually created queries naturally yielded the best results as they were created with regard to the semantic properties of the terms (such as synonymy for example).

We have also done experiments with different ways of creating the query with the proposed method, such as using all terms or only terms with the highest weights, varying the number of levels with AND operator, etc. The results are shown in Fig. 4. As can be seen from the comparison, the best results are obtained when adding all terms

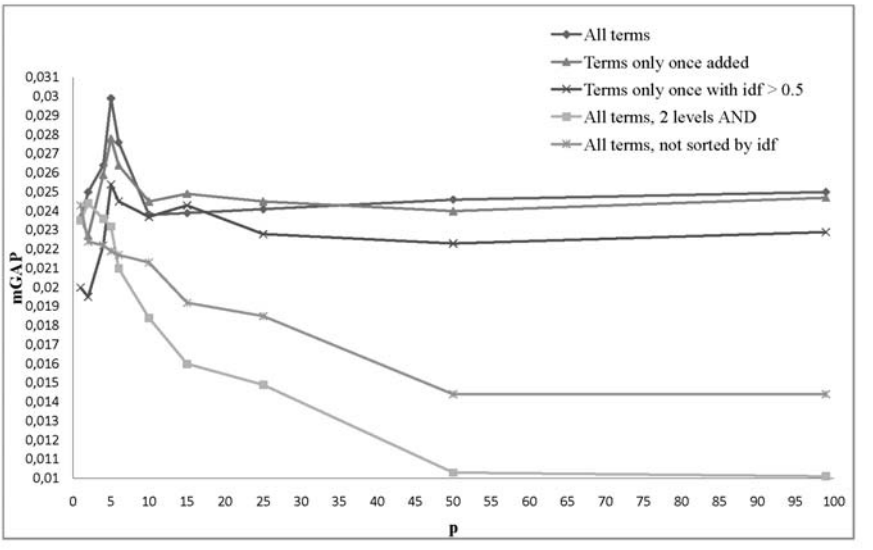

Fig. 4. Different ways of creating the query with the "tree-growing" method (p on the x-axis refers to the parameter of the P-Norm model as defined in Sect. 2.1)

from the topic as many times as they occur in the natural language query. It is also clear that sorting terms by decreasing idf is important.

5 Conclusions and Future Work

As our experiments have shown, our "tree-growing" method almost reaches the performance achieved when using manually created Boolean queries. This method is also much easier to implement and much less computationally intensive than the tested Salton's method. For the verification of our results, we have tested aforementioned methods on 42 CLEF 2007 evaluation topics. The results obtained have shown pretty similar course. The "tree-growing" method achieved even better results than manually created queries, on the other hand, Salton's method achieved slightly worse results than the simple method.

We suppose that using some semantic aspects of the terms in the "tree-growing" method would yield even better results (for example adding synonyms from some vocabulary or using word classes). This would, on the other hand, make the method far more complicated and computationally intensive.

References

1. Croft, W.B., Lafferty, J.: Language Modeling for Information Retrieval. Kluwer Academic Publishers, Norwell (2003)
2. Callan, J.P., Croft, W.B., Harding, S.M.: The INQUERY Retrieval System. In: Proceedings of the Third International Conference on Database and Expert Systems Applications, pp. 78–83 (1992)

3. Pecina, P., Hoffmannová, P., Jones, G.J.F., Zhang, Y., Oard, D.W.: Overview of the CLEF-2007 cross-language speech retrieval track. In: Peters, C., Jijkoun, V., Mandl, T., Müller, H., Oard, D.W., Peñas, A., Petras, V., Santos, D. (eds.) CLEF 2007. LNCS, vol. 5152, pp. 674–686. Springer, Heidelberg (2008)
4. Ircing, P., Pecina, P., Oard, D.W., Wang, J., White, R.W., Hoidekr, J.: Information Retrieval Test Collection for Searching Spontaneous Czech Speech. In: Proceedings of TSD, Plzeň, Czech Republic, pp. 439–446 (2007)
5. Salton, G., Fox, E.A., Wu, H.: Extended Boolean information retrieval. Commun. ACM 26(11), 1022–1036 (1983)
6. Lee, J.H.: Properties of extended boolean models in information retrieval. In: SIGIR 1994: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 182–190. Springer, New York (1994)
7. Salton, G.: A blueprint for automatic Boolean query processing. SIGIR Forum 17(2), 6–24 (1982)
8. Oard, D.W., Wang, J., Jones, G.J.F., White, R.W., Pecina, P., Soergel, D., Huang, X., Shafran, I.: Overview of the CLEF-2006 cross-language speech retrieval track. In: Peters, C., Clough, P., Gey, F.C., Karlgren, J., Magnini, B., Oard, D.W., de Rijke, M., Stempfhuber, M. (eds.) CLEF 2006. LNCS, vol. 4730, pp. 744–758. Springer, Heidelberg (2007)
9. Ircing, P., Oard, D., Hoidekr, J.: First Experiments Searching Spontaneous Czech Speech. In: Proceedings of SIGIR 2007, Amsterdam, The Netherlands (2007)