Some terms are more interchangeable than others

Jakob Klok, Samuel Driessen, Marvin Brunner
Océ Technologies
P.O. Box 101
NL-5900 MA Venlo
The Netherlands
{klok | sjdr | mbru}@oce.nl

Abstract

This paper describes the work done by the Information Retrieval Research Group at Océ Technologies B.V., for the Cross-Language Evaluation Forum (CLEF) 2001. We have participated in the Dutch Monolingual task and submitted three runs. In this paper we introduce a new method for relevance computation using two new query operators. This method is meant to be applied to Multilingual Retrieval in the future.

1 Introduction

This is the first time the Information Retrieval Research Group at Océ Technologies B.V. participates in the CLEF. We started in March 2001 with the implementation of a new method for relevance computation and were interested to see how it compares to other methods. In this paper we present our query model and accompanying ranking formulas. Only monolingual Dutch runs were done so far, but the model has been created with multi-lingual IR in mind.

2 Heuristics

In our system the likelihood of relevance for a document with respect to a query is computed using term rareness, term frequency, document length, completeness, occurrence in important parts (topic & article), and occurrence of phrases / multiword terms. How these factors influence likelihood of relevance is formulated in six heuristics. A document is more likely to be relevant if:

1. it contains rare terms from the query
2. it contains terms from the query many times (taking into account the length of the document)
3. it contains terms that originate from the topic title
4. it contains terms from the query in an important field (title or lead)
5. it contains phrases (consisting of more than one word) from the query
6. it is complete, in the sense that it contains many (non-interchangeable) terms from the query

While heuristics 1 through 4 are common in Information Retrieval, number 5 and 6 need further explanation, which is given in the next two sections.

2.1 Query terms

Heuristic 5 says that documents containing phrases from the topic are more likely to be relevant. Phrases have a more specific meaning than the individual words that they are composed of. For instance, the words *Ierse*, *Republikeinse*, *Leger* (translation: *Irish*, *Republic* and *Army*) have meaning in themselves, but the combination *Ierse Republikeinse Leger* (translation: *Irish Republic Army*) has a narrower and more precise meaning. This example is obvious, but we also extracted phrase like *Japanse automobielindustrie* (translation: *Japanese automotive industry*), *Koreaanse president Kim Il Sung* (translation: *Korean president Kim Il Sung*), and *officiële devaluatie van de Chinese munt* (translation: *official devaluation of the Chinese currency*). We expect that including phrases like these in the query will improve the ranking.

The examples given above are all noun phrases. We choose to extract only noun phrases because they have the most obvious and specific meaning In the future we might include other types of phrases like verb, preposition, adjective and determiner phrases. A rule-based tagger/parser for Dutch was used to find and extract the noun phrases from the topics. This tagger/parser is called Amazon/Casus, mainly built by the Language and Speech

department of the University of Nijmegen, The Netherlands. It tags and parses sentences syntactically and semantically. (For more information on Amazon/Casus refer to [Ama] and [Cas].) All Dutch determiners and pronomina that occur as the first word of the phrase were removed from the term phrase. We also made a short list of stop-phrases, which were removed as well.

We think that adding more phrases to the query improves the ranking. Therefore we generated more phrases by extracting phrases that fall in between stop-words and added those to the queries as well.

2.2 Completeness and interchangeable terms

Heuristic 6 says that we expect a document to be more likely to be relevant when it contains more terms from the query. For instance if we have a query that contains the very different terms "babyvoeding" and "pesticide" (translation: “baby food” and “pesticide”) then it is reasonable to expect that documents that contain both terms are more likely to be relevant than documents that contain only one of the terms. However, there are usually terms in the query that have similar meaning (especially after expansion of terms), in which case the occurrence of all of these similar terms does not make the document more likely to be relevant than the occurrence of only one. For a query on “pesticide” and “pesticiden” (Dutch plural for pesticide), for instance, it does not really matter if a document contains both terms or only one of them.

3 Queries

3.1 Query Definition

In the previous section we described three different kinds of terms, namely single words, noun-phrases, and phrases between stop-words. The set of the terms is defined as the union of the three:

$$Terms = Words \cup Noun\text{-}phrases \cup Between\text{-}stopword\text{-}phrases$$

We also define two query operators, corresponding to the idea that some terms are interchangeable and others are not (cf. section 2.2). We’ll use the symbol ‘|’ as an operator for interchangeable terms and ‘&’ for non-interchangeable terms. We’ll pronounce these as ‘interchange’ and ‘distinguish’ respectively.

We define a query as follows:

- T is a query, if $T \in$ Terms
- $|(Q_1..Q_n)$ is a query, if $Q_1..Q_n$ are queries
- $\&(Q_1..Q_n)$ is a query, if $Q_1..Q_n$ are queries

Note that with the last two rules new queries can be constructed by combining (sub-) queries using our two operators (| and &). This means that not only terms can be interchanged, but also sub-queries.

Actually we use weighted queries, so for each query there is a function *w*, that assigns a weight to each term or sub-query. Notice that interchangeability does not mean that the terms or queries are weighted equally.

3.2 Query Construction

In this section we discuss the construction of queries, the way we did it for run 2. At the highest level a query consists of two sub-queries. One for the terms from the topic title and one for the terms in the topic narrative. These two sub-queries are joined together with the distinguish operator. The reason for not using the interchange operator is that the topic narratives generally appear to contain terms that are not interchangeable with any of the terms from the topic title. The weights for these queries were set to $^5/_7$ and $^2/_7$ for the title and the narrative respectively. This implements heuristic 3.

Each of these two sub-queries again consists of sub-queries. There is a sub-query for each of the terms that has been extracted from the part of the topic at hand. For joining these queries the distinguish operator is used, since the terms are not known to be interchangeable[1]. Phrases are included in the query, following heuristic 5. Stop-words[2] and stop-phrases are excluded. Noun-phrases and phases between stop-words receive twice as much weight as single words. At this point the rareness of terms is also taken into account, because of heuristic 1. We adjusted the weights for these queries using expression (1), which gives a higher weight to rare terms. In (1) *w* is

[1] With the use of a thesaurus or semantic network this might be detected though.

[2] The stop-word list that was used was a standard list, with the only difference that *weer* was left out of it, since it also means *weather*, in which case it is not a stop-word.

the weight for the sub-query, *T* is the term that this query is about, *df(T)* is the document frequency for *T*, and *N* is the number of documents in the collection.

$$w := w \times \frac{\log \frac{N}{df(T)}}{\log N} \qquad (1)$$

Again these queries consist of sub-queries, for the various stemming variants of the terms. We applied stemming expansion only to single words, not to phrases. Open Muscat's implementation of Porter's stemming algorithm [Porter] for Dutch was used for this purpose. Queries for the terms that literally appeared in the topics were given a twice-higher weight than the stemming variants. The interchange operator is used to join the queries, since the variants are interchangeable.

Queries for words with accented characters were expanded further into one sub-query for the accented variant of the word and one for the unaccented version, where the first received twice the weight of the latter.

At the deepest level there are sub-queries for each part of the article parts, i.e. title, lead, and body-text; this implements heuristic 4. These sub-queries are joined together with the interchange-operator. The weights for these sub-queries are $^2/_7$ for those in the title and lead and $^3/_7$ for the one in the body-text.

A graphical view of a query with its four levels of sub-queries is depicted in figure 1 below. The original query constructed by our software from topic 47 is pruned for display purposes, yet it shows the aspects of weights, phrases, stemming variants and the use of the two operators.

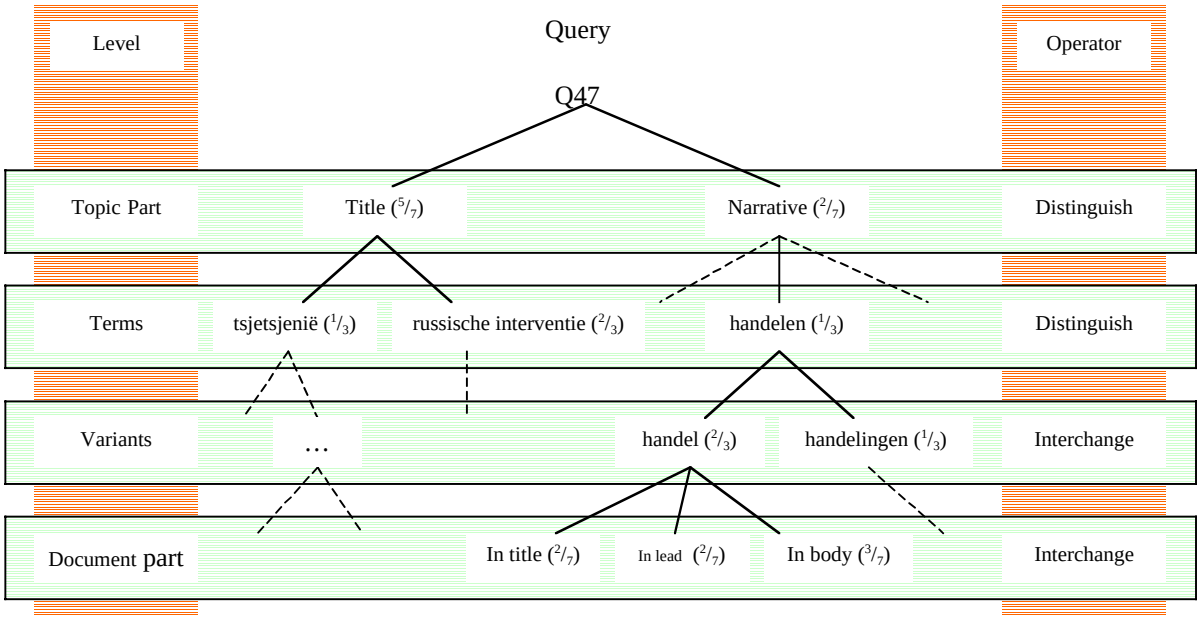

Figure 1: A graphical view of a query with sub-queries

4 Relevance Computation

Now that queries can be constructed, we discuss the estimation of likelihood of relevance. Since we work with a recursive definition of queries, it is logical to estimate likelihood of relevance for each of the sub-queries and combine those. The estimates for the sub-queries do need to be comparable in the sense that they are on the same scale. We decided that relevance estimates should fall between 0 and 1, where 0 indicates a low estimated likelihood of relevance and 1 high, in comparison to the other documents (a).

$$0 \le REL(Q, D) \le 1 \qquad (a)$$

Remembering our definition of a query we see that some queries consists of nothing more than a term. We'll use heuristic 2 for estimating likelihood of relevance for this type of query. First we define term frequency *tf* (b). In this formula *rawtf* is the number of occurrences of T in D and the length of a document is the total number of words in a document. We divide by the logarithm of the length**,** because our intuition was that dividing by the length itself would be too much.

$$tf(T,D) = \frac{rawtf(T,D)}{\ln(length(D))} \quad \text{(b)}$$

Following heuristic 2 and satisfying (a), we formulate (c) for the estimation of likelihood of relevance for a query consisting of just one term. The relevancy is based on median value of frequencies for the term in other documents and used in run 2. In run 3 we used (c'), which uses the maximum term frequency in the collection for the term. In the future a formula will be used that is based on both the median and the maximum.

$$REL(T,D) = \begin{cases} \dfrac{tf(T,D)}{2 \times \underset{d \in DOCS}{median}(T,d)} & if \ \dfrac{tf(T,D)}{2 \times \underset{d \in DOCS}{median}(T,d)} \leq 1 \\ 1 & if \ \dfrac{tf(T,D)}{2 \times \underset{d \in DOCS}{median}(T,d)} \geq 1 \end{cases} \quad \text{(c)}$$

$$REL(T,D) = \frac{tf(T,D)}{\max\limits_{d \in DOCS}(T,d)} \quad \text{(c')}$$

For calculation of the relevance of combined queries, two different formulas for the two operators (cf. Section 3.1) are defined, following heuristic 6. For a combination of non-interchangeable sub-queries we define (d). This formula gives a higher score for "complete" documents than for non-complete.

$$REL(\&(Q_1..Q_n), D) = \sum_{q \in Q_1..Q_n} w(q) \cdot REL(q,D) \quad \text{(d)}$$

The formula for queries with interchangeable sub-queries defined by (e) is more complicated. It scores documents regardless of completeness.

$$REL(|(Q_1, Q_2..Q_n), D) = w(Q_1) \cdot REL(Q_1, D) + (1 - REL(Q_1, D)) \cdot REL(|(Q_2..Q_n), D) \quad \text{(e)}$$

In order to satisfy constraint (a) for both (d) and (e) we normalise the weights such that the sum of the weights for a query is exactly one (f).

$$\sum_{q \in Q_1..Q_n} w(q) = 1, \text{ for queries } \begin{cases} |(Q_1..Q_n) \\ \&(Q_1..Q_n) \end{cases} \quad \text{(f)}$$

5 Results

We'll concentrate our discussion on run 2, which was the best run. Considering that we had not started on CLEF until March and spent most of our time on implementation, rather than experimentation, we had not expected very good results. Indeed figure 2 shows that for a number of queries the results are well below the median. Nevertheless, the results are well above the median for a number of queries as well.

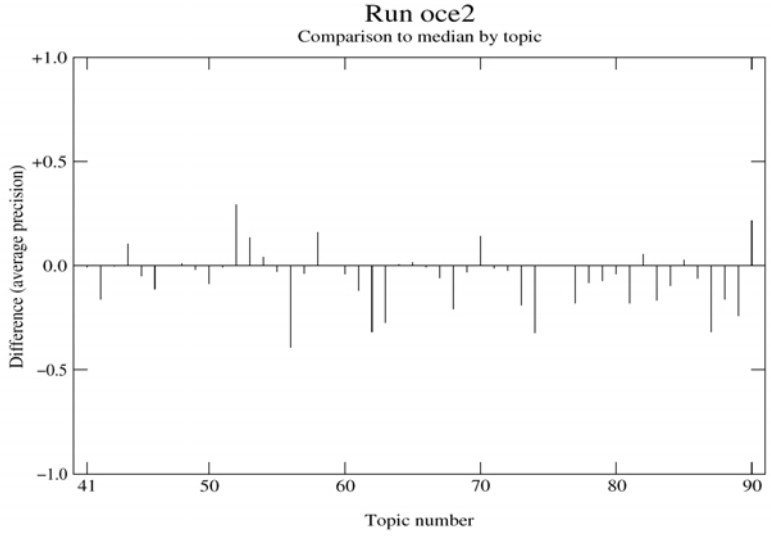

Figure 2: Comparison to median by topic for run2

One obvious opportunity for improvement is the tuning of the parameters. The parameters used so far were based on intuition only. Better values for the parameters could probably be found by running experiments with different values and comparing the results.

A second cause for poor results for some topics can be the fact that we did not try to interpret the narrative parts of the topics. We only used the narratives to generate more terms for our queries. In some cases the narrative explicitly disqualifies a portion of documents. For instance topic 074 about the channel tunnel requires that names of prominent persons that were present at the opening were mentioned in the article. Our program did not check for this in any way. Topic 56 puts a similar restriction on the set of relevant documents.

Unfortunately a third cause for poor results for at least one topic appears to have been an error in the code. The query that was generated for topic 087 about the Brazilian elections did not contain the terms from the topic narrative. If it had, it would have contained among other terms “Plan Real”, “Braziliaanse regering” (English: Brazilian government), and “Braziliaanse verkiezingen” (English: Brazilian election). Undoubtedly this would have made a considerable difference.

6 Conclusions and future work

In the future we plan to run a number of experiments to investigate various aspects of our retrieval model. We want to systematically compare the influence of each of the six heuristics, formulated in section 2, on the retrieval performance of the system.

We shall also investigate other query expansion mechanisms, such as expansion with synonyms and splitting of compound words. Pseudo relevancy feedback can also be built into the model. Finally, we shall attempt multilingual retrieval by expanding terms with translations in other languages.

7 References:

[Ama] http://lands.let.kun.nl/projects/structuralist.en.html
[Cas] http://lands.let.kun.nl/TSpublic/dreumel/amazon_casus.en.html
[Porter] M. F. Porter, .“An algorithm for suffix stripping”, in: *Program*, 14(3): 130—137, 1980.