Abstract. Information retrieval focuses on finding those documents whose content matches a user’s query from a large collection of documents. Formulate queries which are well designed for retrieval purposes is difficult for most users, suggesting use query expansion to retrieve relevant information. Query expansion techniques are widely used in biomedical literature for improving the efficiency of textual information retrieval systems. This techniques help to overcome vocabulary mismatch issues, such as synonymy between the original query terms and a relevant document, by expanding the original query with additional relevant terms and reweighting the terms in the expanded query. In this paper, a variety query expansion approaches are presented for improving the documents initially retrieved by a query. A corpus belonging to MEDLINE, called Cystic Fibrosis, is used as a knowledge source. The results obtained are similar to the results obtained by other authors.

Keywords: Query expansion, Biomedical information retrieval, Lemur, MEDLINE

1 Introduction

Biomedical knowledge is growing at a high pace and large collections of publications offer an excellent opportunity for discovering hidden biomedical knowledge by applying information retrieval (IR) and related technologies. Information retrieval (IR) is related with representation, storage, organization and access to information terms. Terms must be represented and organized in order to allow that user has an easy access to the information of interest (it is not a simple problem to recognize it). The user’s requireriments must be presented in a good format so to be translated these information into a query which can be processed by the search engine (or IR systems). This translation is presented like a set of keywords (or index terms) which summarizes the information in which the user is interested [1].

The main goal of a IR system is to retrieve information useful or relevant to the user on a subject. Information Retrieval is not equal to data retrieval that satisfy a query which consists mainly in determining what documents in the collection contain the keywords entered in the query, not a subject. Information retrieval using only keywords (as in the case of retrieve data) is not usually very efficient. In general, information about a particular issue can be represented with different keywords, which could not coincide exactly with the terms entered in the query by the user. The user’s query can include keywords that are not present in documents, but documents could be relevant because they have another words with the same meaning. Using query expansion (QE) a query is reformulated to improve retrieval performance obtaining additional relevant documents by expanding the original query with additional relevant terms and reweighting the terms in the expanded query. For this reason, query expansion techniques are widely used in biomedical literature for improving the efficiency of textual information retrieval systems, helping to overcome vocabulary mismatch issues including words in queries with same/related meaning.

In this paper, different techniques of QE were analized in order to know whether techniques tested could provide a better or equal retrieval performance than the results shown by other authors.

This research aims to improve the searches based on the classic Boolean model (where documents are retrieved if to a small part of them is related with queries, as a word) performs Pubmed (a free database of references and abstracts on life sciences and biomedical topics) using expansion techniques. PubMed allow accessing primarily the MEDLINE database which contains documents with different fields such as MeSH field, where we can find MeSH Headings indexing documents, concepts that describe the main content of documents of great importance in research.

The rest of the article is organized as follows: section 2 presents an overview of the query expansion, section 3 describes methods of QE employed in this research for retrieving relevant documents and results obtained comparing them with results obtained by other authors. The article conclude with the conclusions and future work in section 4.

2 Query expansion

Information retrieval searching is composed of two main processes, indexing and matching (see Fig. 1). The first step, in order to retrieve documents, is index documents and queries, belonging to a corpus composed of three main elements in information retrieval: documents, queries and relevance judgements given by the experts, for obtaining keywords to be used in the process (these keywords represent relevant words in documents and queries). At this point, is vital to analyze the use of stemming and stopwords lists (for obtaining the keywords that represent documents and queries) in order to reduce related words to their stem, base or root form. This is achieved launching affix removal for adapting different derivational or inflectional variants of the same word to a single indexing form and remove words without significant information to the document. Further, queries can be enhanced by expansion techniques modifying the words they contains such as the use of other keywords that represent them in a manner more consistent with the content of documents (for example MeSH Headings by means of MeSH Browser, a tool that can be selected from the sidebar menu in PubMed and provides users the option to search MeSH for matching concepts to the entered terms or phrases being used in the expansion process to locate MeSH Headings).

Matching is the process of computing a measure of similarity between documents and queries by weighting algorithms of terms, being TF-IDF and BM25 the most important. Most retrieval systems return a ranked list of documents in response to a query which are ordered such that the documents the system believes more similar to the query are first on the list. Once obtained the first documents ranked for a query, can be applied expansion techniques. The analysis of previous documents retrieved is through taking into account new keywords representing them to add to initial query in order to reranking documents with the benefits of these new keywords. This process is known as feedback. For the process indexing, mapping or feedback, Lemur Language Modeling Toolkit \((a\ software\ tool\ designed\ to\ facilitate\ research\ in\ language\ modeling\ and\ information\ retrieval(IR)\ providing\ methods\ for\ parsing\ query,\ indexing\ documents\ and\ retrieve\ documents\ related\ to\ queries\ using\ several\ weighting\ algorithms)\ has\ been\ used\ [2,3].

In order to evaluate results obtained (a set of documents for each query) in the retrieval process is used a program inside the TREC conference trec_eval\(^3\). This program allows to get several measures like Total number of documents over all queries (Retrieved, Relevant and Rel_ret(relevant and retrieved)) or MAP, R-prec and, Interpolated Recall-Precision Averages. These measures are used to compare our results with those of other authors. Details of the comments above have been presented in next sections.

2.1 Terms Definitions

When working with a information retrieval system certain key issues in QE can improve the results. Understanding the process of query expansion some concepts related with Corpus of documents, Stemming, Stopwords, Use of Acronyms, Okapi BM25 weighting algorithm, TF-IDF weighting algorithm, Blind relevance feedback and Measures are introduced in the next subsections.

Corpus of documents

An information retrieval test corpus used to analyze the effectiveness of query expansion is composed by three main elements: the set of documents, the textual descriptions of the users queries called topics, and the right answers called relevance judgements given by the experts \([4]\). The documents represent a sampling of articles published, the format of the data uses a labeled bracketing, the topics are a description in natural language of the information that the user needs, typically one sentence, and finally,

\(^2\) Lemur Project http://www.lemurproject.org/
\(^3\) trec_eval http://trec.nist.gov/trec_eval/
the relevance judgements done by potential users (called experts or judges) in a carefully controlled experiment, allow to calculate the effectiveness of IR systems.

Most of the corpus used in biomedical information retrieval belong to MEDLINE (like those used in the conference TREC, Cystic fibrosis and OHSUMED⁴), a very large database maintained by the National Library of Medicine (NLM) with about 18 million of abstracts of research papers in medical domain. Each record is structured according to a specific set of fields⁵. From an IR perspective, the most important fields are the article Title(TI), the Abstract(AB) and the set of manually assigned MeSH Headings(MH) extracted from the MeSH⁶ Thesaurus.

MeSH (Medical Subject Headings) is a extensive list of the controlled vocabulary thesaurus used to indexing journal articles for subject analysis of biomedical literature NLM. It imposes uniformity and consistency to the indexing of biomedical literature. MeSH has a hierarchical structure with sets of terms naming descriptors that permits searching at various levels of specificity. Expert annotators, based on indexed content of documents, assign MeSH Headings terms to the documents in order to allow to the user retrieve the information that explains the same concept with different terminology. On average, 5 to 15 subject headings are assigned by document, 3 to 4 of them being Major Headings and the others being Minor Headings. Major MeSH terms describing the primary/main topics of the document and Minor MeSH terms giving more details/secondary content about it [5–9]. Other databases have not MH field with these keywords. MeSH contains approximately 26 thousand terms and is updated annually to reflect changes in medicine and medical terminology. MeSH Headings often appear together with MeSH Subheadings that are used to help describe more completely a particular aspect of a subject. For example, the drug therapy of asthma is displayed as asthma/drug therapy in MeSH Terms[MH]. Furthermore, each MeSH Heading is related with several Entry terms. Entry terms, are synonyms, alternate forms, and other closely related terms in a given MeSH record that are generally used interchangeably with the MeSH Heading for the purposes of indexing and retrieval, thus increasing the access points to MeSH-indexed data.

⁴ OHSUMED http://ir.ohsu.edu/ohsumed/ohsumed.html
⁵ MEDLINE Fields http://www.ncbi.nlm.nih.gov/books/NBK3827/?rendertype=table&id=pubmedhelp.T44
Stemming

The aim of stemming (process for reducing related words to their stem, base or root form through affix removal) is to adapt different derivational or inflectional variants of the same word to a single indexing form. Affix removal is language-dependent and creating a new stemmer involves analyzing text resources to produce rules [10,11]. Stemmers have to be adaptable for new languages, incorporating the language-specific morphological rules in order to form words which can be an expensive and time-consuming task. The most stemmers are rule-based and are widely available only for English and other west European languages. There are two major stemmers in use for English IR: the Porter stemmer and the Krovetz stemmer.

Porter Stemmer was developed by Martin Porter, at the University of Cambridge in 1980 and was first published in Porter, M.F. [12]. As described above, “The Porter stemming algorithm (or Porter stemmer)” is a process for removing the common morphological and inflexional endings from words in English. It has been very widely used and coded in various programming languages. It is based mainly on stemming operations that remove suffixes from words, such as gerunds, plurals, and replacing words ending. It is composed by rules where each of these deals with a specific suffix and having certain conditions to satisfy. The suffixes of words are checked against each rule in a sequential manner until it matches one, the conditions in the rule are tested and it may result in a suffix removal or modification [13,14].

Krovetz Stemmer was developed by Bob Krovetz, at the University of Massachusetts, in 1993. The Krovetz Stemmer removes inflectional suffixes in three steps, the conversion of a plural to its single form, the conversion of past to present tense, and the removal of -ing. The process firstly removes the suffix, and then through a process of checking in a dictionary returns the stem to a word [10].

Stopwords

In information retrieval, a document is indexed by frequency of words in it. Statistical analysis of that process showed that some words have quite low frequency, while others have high frequency [15]. For example, and, of, and the appear frequently in the documents without significant information to the document. This set of words is referred as stop words. Elimination of stop words could significantly reduce the size of the indexing structure, can speed up the calculation and increase the accuracy. Up to now, a lot of stop word lists have been developed for English language for example The U. S. National Library of Medicines official list of stopwords7 and The stopword list built by Gerard Salton and Chris Buckley for the experimental SMART information retrieval system at Cornell University8.

Acronyms

Acronyms are widely used in biomedical literature. The names of many clinical diseases and procedures, and of common entities such as genes or proteins, have widely used acronyms so it is vital to recognize them in information retrieval. There are a lot of research related with acronyms, long forms and their relations [16, 17]. Identify acronyms and long forms is vital in query expansion because work with both forms in queries help to retrieve documents where appear one long form, one acronym or both [18–20].

Okapi BM25 weighting algorithm

Okapi BM25 or BM25, is a ranking function used to rank documents according to their relevance to a given search query9. It is based on the probabilistic retrieval. Okapi BM25 function is used by many researchers and different corpus to retrieve relevant documents, such as TREC [21,22] or OSHUMED [23] where:

\[
\text{score}(d,q) = \sum_{T \in Q} \text{idf}(q_i) \cdot \frac{(k_1 + 1) \cdot df_T}{K + df_T} \cdot \frac{(k_3 + 1) \cdot qf}{k_3 + qf}
\]

(1)

\[
\text{idf}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}
\]

(2)

– \(N\) is the total number of documents in the collection.

8 SMART stop-word list http://www.lextek.com/manuals/onix/stopwords2.html
− \( n(q_i) \) is the number of documents containing \( q_i \),
− \( Q \) is a query, containing terms \( T \)
− \( K \) is \( k_1 \cdot ((1 - b) + b \cdot \frac{dl}{avdl}) \)
− \( k_1, b, \) and \( k_3 \) are parameters which depend on the on the nature of the queries and possibly on the database; \( k_1 \) and \( b \) default to 1.2 and 0.75 respectively, but smaller values of \( b \) are sometimes advantageous; in long queries \( k_3 \) is often set to 7 or 1000 (effectively infinite)
− \( dtf \) is the frequency of occurrence of the term within a specific document
− \( qtf \) is the frequency of the term within the topic from which \( Q \) was derived
− \( dl \) and \( avdl \) are respectively the document length and average document length measured in some suitable unit.

When using feedback the Okapi BM25 formula was replaced by the following relevance weight:

\[
\text{score}(d, q) = \sum_{T \in Q} w^{(1)} \cdot \frac{(k_1 + 1) \cdot dtf}{K + dtf} \cdot \frac{(k_3 + 1) \cdot qtf}{k_3 + qtf}
\]  

− \( w^{(1)} \) is the Robertson Sparck Jones weight \([24, 25]\) of \( T \) in \( Q \)

\[
\log \frac{(R + 0.5)/(R - r + 0.5)}{(n - r + 0.5)/(N - n - R + r + 0.5)}
\]

− \( R \) is the number of documents known to be relevant to a specific topic
− \( r \) is the number of relevant documents containing the term

The values of the parameters \( k_1, k_3, \) and \( b \) \([26–28]\) should be adjusted based on the collection and type of queries used, although the commonly default values: \( k_1 \) and \( k_3 \) between 1.2 and 2, usually 1.2 but \( k_3 \) may increase to between 7 and 1000 in the case of long queries, and \( b = 0.75 \) although small values can sometimes report improvements.

**TF-IDF weighting algorithm**

The TF–IDF weighting algorithm (term frequency–inverse document frequency) is often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word to a document is in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the tf–ids weighting scheme are often used by search engines as a central tool in scoring and ranking a document’s relevance given a user query \([29, 30]\).

\[
(tf-idf)_{i,j} = tf_{i,j} \cdot idf_i
\]

\[
\log \frac{|D|}{|\{d : t_i \in d\}|}
\]

− \(|D|\): cardinality of \( D \), or the total number of documents in the corpus.
− \(|\{d : t_i \in d\}|\): number of documents where the term \( t_i \) appears (that is \( n_{i,j} \neq 0 \)). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to use \( 1 + |\{d : t_i \in d\}| \)

A high weight in \( tf-idf \) is obtained with an high term frequency in the document and a low document frequency in the whole collection. The weights hence tend to filter out common terms, if a term appears in all documents, his idf will be zero.

TF-IDF algorithm as mentioned above have variations used in information retrieval to weight the documents and queries \([30–32]\). Each document and each query are represented by a term frequency vector \( \vec{d} = (x_1, x_2, \ldots, x_n) \) and \( \vec{q} = (y_1, y_2, \ldots, y_n) \) respectively, where \( n \) is the total number of terms, or the size of the vocabulary and \( x_i, y_i \) are the frequency of term \( t_i \) in \( d \) and \( q \) respectively. Given a
Table 1. tf formulas for TF-IDF weighting algorithm.

<table>
<thead>
<tr>
<th>Formulas</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tf_d(x_i) = \frac{k_1 x_i}{x_i + k_1 \cdot \frac{1}{</td>
<td>C</td>
</tr>
<tr>
<td>$tf_q(y_i) = \frac{k_1 y_i}{y_i + k_1 \cdot \frac{1}{</td>
<td>C</td>
</tr>
<tr>
<td>$tf_d(x_i) = x_i$</td>
<td>RawTF formulas</td>
</tr>
<tr>
<td>$tf_q(y_i) = y_i$</td>
<td></td>
</tr>
<tr>
<td>$tf_d(x_i) = \log(rawTF + 1)$</td>
<td>LogTF formulas</td>
</tr>
<tr>
<td>$tf_q(y_i) = \log(rawTF + 1)$</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Correspondence between parameters of algorithms.

<table>
<thead>
<tr>
<th>BM25</th>
<th>TF-IDF</th>
<th>BM25 TF</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tf_d$</td>
<td>$k_1$</td>
<td>$k_1$</td>
</tr>
<tr>
<td>$tf_d$ and $tf_q$</td>
<td>$b$</td>
<td>$b$</td>
</tr>
<tr>
<td>$tf_q$</td>
<td>$k_3$</td>
<td>$k_1$</td>
</tr>
</tbody>
</table>

collection $C$, the inverse-document-frequency $idf = (|C|/n_t)$ where $n_t$ is the number of documents with $t$, the weighted vectors for $\vec{d}$ and $\vec{q}$ are:

$$\vec{d} = (tf_d(x_1) \cdot idf(t_1), tf_d(x_2) \cdot idf(t_2), \ldots, tf_d(x_n) \cdot idf(t_n))$$ (8)

$$\vec{q} = (tf_q(y_1) \cdot idf(t_1), tf_q(y_2) \cdot idf(t_2), \ldots, tf_q(y_n) \cdot idf(t_n))$$ (9)

In Table 1 appear tf formulas for TF-IDF weighting algorithm. The score of document $\vec{d}$ against query $\vec{q}$ is

$$s(\vec{d}, \vec{q}) = \sum_{i=1}^{n} tf_d(x_i)tf_q(y_i)idf(t_i)^2$$ (10)

Table 2 contains the correspondence between the parameters of BM25 weighting algorithm and TF-IDF weighting algorithm with BM25 TF formulas.

User Relevance Feedback

Relevance feedback is the most popular query reformulation strategy. Firstly the user selects the most relevant documents in the answer set obtained with the initial query - Only top 10 (or 20) ranked documents need to be examined. The representative terms (or expressions) of the selected documents are included in a new query formulation. The new query will be reformulated towards relevant documents and away from the non-relevant ones. Considers that the term-weight vectors of the documents identified as relevant (to a given query) have similarities among themselves (relevant documents seem each other and non-relevant documents have term-weight vectors which are different from the ones). The idea is reformulate the query such that it gets closer to relevant documents [10,33,34].

Advantages of the relevance feedback:
— It shields the user from the details of the query reformulation process because the user only provides a relevance judgement on documents.
— It breaks down the whole searching task into a sequence of small steps which are easier to grasp.
— It provides a controlled process designed to emphasize some terms (relevant ones) and minimize others (non-relevant ones).

The Rocchio feedback approach (see Eq. (11)) is based on the assumption that most users have a general conception of which documents should be denoted as relevant or non-relevant. The motivation is that in practice the original query $q$ may contain important information. Usually, the information contained in the relevant documents is more important than the information provided by non-relevant documents. This suggests making the constant $\gamma$ which represents weight attached to the set of known non-relevant documents smaller than the constant $\beta$ which represents weight attached to the set of known relevant documents. An alternative approach is to set $\gamma$ to 0 which yields a positive feedback strategy. Weights are increased or decreased for a particular category of documents, the coordinates for the modified vector begin to move either closer, or farther away, from the centroid of the document collection. Thus if the weight is increased for relevant documents, then the modified vectors coordinate’s will reflect being closer to the centroid of relevant documents.

$$q_m = \alpha q + \frac{\beta}{|D_r|} \sum \forall d_j \in D_r d_j - \frac{\gamma}{|D_n|} \sum \forall d_j \in D_n d_j$$

(11)

where:
- $D_r$: set of relevant documents identified by the user among the retrieved documents
- $D_n$: set of non-relevant documents among the retrieved documents
- $C_r$: set of relevant documents among all documents in the collection
- $R_d$: set of retrieved documents by the user
- $|D_r|, |D_n|, |C_r|$: number of documents in the sets $D_r, D_n$, and $C_r$
- $\alpha, \beta, \gamma$: tuning constants (The number of relevant and non-relevant documents allowed to enter a query)
- $\alpha$: Original Query Weight
- $\beta$: Relevant Documents Weight
- $\gamma$: Non-Relevant Documents Weight

The main advantages of the above relevance feedback techniques are simplicity (modified term weights are computed directly from the set of retrieved documents) and good results (are observed experimentally, the modified query vector does reflect a portion of the intended query semantics). The main disadvantage is that no optimality criterion is adopted.

Measures

In order to evaluate results exist one program which is called trec_eval\(^{10}\) which allow to get several measures related with information retrieval [35]. The most commonly used are the following:

**Precision at 11 standard recall levels**

In IR systems the precision averages at 11 standard recall levels are used to compare the performance of different systems in a recall-precision graph. Each recall-precision average is computed by summing the interpolated precisions at the specified recall cutoff value (denoted by $P \lambda$ where $P \lambda$ is the interpolated precision at recall level $\lambda$) and then dividing by the number of topics.

$$\sum_{i=1}^{|Q|} P \lambda \lambda = \{0.0, 0.1, 0.2, 0.3, \ldots, 1.0\}$$

**Average Precision**

For systems that return a ranked sequence of documents, it is desirable to also consider the order in which the returned documents are presented. Average precision emphasizes ranking relevant documents higher. It is the average of precisions computed at the point of each of the relevant documents in the ranked sequence:

\[
AveP = \frac{\sum_{r=1}^{N} (P(r) \times rel(r))}{C_r}
\]

\(r\) is the rank
\(N\) the number retrieved
\(rel(r)\) a binary function on the relevance of a given rank
\(P(r)\) precision at a given cut-off rank

\[
P(r) = \frac{|\{D_r|_{<r}\}|}{r}
\]

**Mean Average precision**

Is the mean of the average precision scores for each query.

\[
MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{|Q|}
\]

**R-precision**

R-Precision is the precision after \(R\) documents have been retrieved, where \(R\) is the number of relevant documents for the topic.

### 3 Methodology and Results

In this section an outline is presented in accordance with the methodology of the process of expansion of this research and the results are shown (see Fig. 4). The corpus used for testing was *Cystic Fibrosis Corpus*\(^{11}\) (CF) in our study. It consists of 1239 documents published from 1974 to 1979 discussing Cystic Fibrosis Aspects in the National Library of Medicine’s Medline composed by Abstract, Title and MeSH fields between others (see Fig. 2), and a set of 100 queries with the respective relevant documents as answers belong to experts \([1]\)(see Fig. 3).

![Fig. 2. A sample of MEDLINE data](http://grupoweb.upf.es/WRG/mir2ed/ref.php)

Information Retrieval process has been divided in two: the expansion in Lemur and the expansion outside Lemur. Once arrived at a particular point of evidence, and given that Lemur is a closed tool, we decided, to perform weighting tasks with MeSH terms in documents giving more weight to the Major MeSH Headings that the Minor, to work outside of it. The tests performed within Lemur are based on an analysis of the benefits produced by the stemming and stopwords in the index of documents and queries and the benefits of using acronyms in the collection. To recognize acronyms is vital in order to

not be affected by stemming and stopwords processes and thus make a direct mapping between them in documents and queries. Expanding acronyms in queries could be obtained advantages to locate them in documents, in addition to own acronym, the long forms of the same. Parameterize weighting algorithms is an important point to get results commensurate with the collection used, for this, this was the next test in our process. Following previous studies by other authors, has been also tested the benefit of working with all fields of the documents compared to only work with one of them (so far our tests were focused on the Abstract). To improve the results of the above processes, we proceed to make relevance feedback. MeSH field in the documents is considered one of the most important fields in retrieving documents that are appropriate to the queries, so firstly, we have made tests against the MeSH field to see what results were obtained (with the queries and the expanded queries leaving only the MeSH Headings). We have explored also the possibility of feedback the Abstract results with the list of retrieved documents by searching in the MeSH field. To improve the results of the previous step, as mentioned above, process has been taken out of Lemur to analyze the possibility of giving more weight to the Major MeSH Headings that the Minor, within documents. The results have been combined with those obtained previously by the Abstract field to see how to improve them. Finally it has also been combined with the results obtained by the Title field to achieve the final result.

Fig. 4. Steps into Methodology.
Query expansion in Lemur

The expansion process is explained below performed within Lemur and then continue with subsequent steps out of the tool.

Stemming and Stopwords

Based on query expansion terms definitions discussed in section 2, we have analyzed the impact of stemming algorithms (Porter and Krovetz) and lists of stopwords (NLM and SMART) in the retrieving of documents for queries of corpus Cystic fibrosis. This test is made in order to obtain the improvements using stemming and stopwords lists. Okapi BM25 weighting algorithm was used with basic parameters \((k_1=1.2, b=0.75, k_3=7)\) using Abstract field in order to see improvement [21,22,36].

Table 3. Improvements using different techniques combined of stemming and stopwords.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>MAP</th>
<th>R-prec</th>
<th>Dr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.1545</td>
<td>0.2098</td>
<td>683</td>
</tr>
<tr>
<td>Porter</td>
<td>0.1663</td>
<td>0.2154</td>
<td>747</td>
</tr>
<tr>
<td>Krovetz</td>
<td>0.1663</td>
<td>0.2231</td>
<td>740</td>
</tr>
<tr>
<td>Stopwords NLM</td>
<td>0.1681</td>
<td>0.2242</td>
<td>723</td>
</tr>
<tr>
<td>Stopwords SMART</td>
<td>0.1695</td>
<td>0.2243</td>
<td>728</td>
</tr>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.1790</td>
<td>0.2332</td>
<td>786</td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.1808</td>
<td>0.2342</td>
<td>790</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.1799</td>
<td>0.2333</td>
<td>782</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.1808</td>
<td>0.2324</td>
<td>786</td>
</tr>
</tbody>
</table>

Table 3 shows a comparison of stemming functions, firstly. We can see that stemming is an effective technique to improve MAP. Normally between weak (Krovetz) and strong (Porter) stemming methods, the performances are comparable although in our case by itself does not provide information because results are the same. In terms of MAP, strong stemming is a bit better. In terms of R-prec, weak stemming is a bit better, as observed for R-prec in Porter and Krovetz in the table. If compared different stop word removal methods it shows that removing stop words improves the performance. From our experiments, using the stopword list more large (SMART) results are better than using list with less stopwords. We can see how the Porter stemmer with stopword list of SMART gives us the best results. In last four combinations (highlight with bold style), we don’t see significative differences so we had conduct tests with these four combinations. Of this test can be deduced why most research groups use Porter stemmer combined with stopwords to index. We also note that the best result is to use a long list of stopwords (SMART).

York University at TREC 2006 [36] analyzed how the use of stemming and stop word removal can improve the performance, but the degree of improvement depends on the stemming method and the stop word list used. According to our results obtained but using a different corpus, they concluded that in terms of MAP, strong stemming is a bit better (Porter in our case) but in terms of R-prec, weak stemming (Krovetz) is a bit better, with results in a range of 3% of improvement in MAP. Results obtained with our test had a range of 2% of improvement in our collection. They concluded that the degree of improvement depends on the stop-word list used achieving an improvement in a range of 1% in MAP. From our experiments, using the stop word list that contains few stop terms (NLM in our case) is better than using the larger list that contains a lot of stop words (SMART) achieved a range between 1-2% of improvement in our collection. Combining both options (stemming and stopwords) got a range of 3% of improvement.

Acronyms

Identify acronyms and long forms is vital in query expansion because work with both forms in queries helps to retrieve documents where appear one long form, one acronym or both [18–20]. Okapi BM25
weighting algorithm was used with basic parameters ($k_1=1.2$, $b=0.75$, $k_3=7$) using Abstract field in order to see improvement [19,21,22].

**Table 4. Okapi BM25 with Acronym List.**

<table>
<thead>
<tr>
<th>Combinations</th>
<th>Without acronyms</th>
<th>With acronyms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.1790</td>
<td>0.1790</td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.1808</td>
<td>0.1808</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.1799</td>
<td>0.1799</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.1808</td>
<td>0.1808</td>
</tr>
</tbody>
</table>

In Table 4 we have worked with the recognition of acronyms in queries and documents so that neither the stemming nor stopwords lists act on them and keeping them intact in documents and queries for exact matching, through a recognition module of acronyms that Lemur\textsuperscript{12} offers. The table shows that acronyms in this case aren’t beneficial. We can deduce that expanding the query using acronyms and their long forms will not be beneficial for this collection. After a study of the collection, we realized that the queries only have an acronym CF (apart of some isolated case) linked to long form cystic fibrosis, and in almost all documents, appears the same acronym. So, being the acronym CF in almost all queries and in almost all documents and being cystic fibrosis the central theme, we will not get benefit since it related a lot of documents to any question. Results are shown below related to this issue in the process out of Lemur.

**Parameterization of weighting algorithms**

In general Okapi BM25 with default parameters have been used by researchers for the study of the methods. One of the most important steps, when working with a information retrieval systems, is to choose the appropriate weighting algorithms for ranking documents. Once an investigation and a description of weighting algorithms have been done in section 2 of the two most important, Okapi BM25 and TF-IDF, the parameters for Cystic fibrosis collection have been adjusted, since parameters can vary from collections to other. In Okapi BM25 the values of the parameters $k_1$, $k_3$, and $b$ [26–28] should be adjusted based on the collection and type of queries used, although the commonly default values: $k_1$ and $k_3$ between 1.2 and 2, usually 1.2 but $k_3$ may increase to between 7 and 1000 in the case of long queries, and $b=0.75$ although small values can sometimes report improvements. MAP has been used as a measure in order to base our calculations.

**Table 5. Finding the best $k_1$ parameter for BM25.**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>1.5</th>
<th>1.3</th>
<th>1.2</th>
<th>1.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b$</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>$k_3$</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Combinations</th>
<th>$k_1=1.3$, $b=0.6$, and $k_3=1.2$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.1521 0.1803 0.1792 0.1792 0.1807 0.1798 0.1792</td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.1531 0.1820 0.1783 0.1788 0.1813 0.1815 0.1807</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.1521 0.1797 0.1778 0.1802 0.1804 0.1806 0.1799</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.1557 0.1828 0.1793 0.1801 0.1824 0.1815 0.1819</td>
</tr>
</tbody>
</table>

The best values that we obtained (see Tables 5, 6, 7) were: $k_1=1.3$, $b=0.6$, and $k_3=1.2$.

For TF-IDF weighting algorithm with $tf$ formula given by Okapi we found that the parameters obtained above were indeed the best approach because it uses the BM25 approximation and its parameters.

Moreover, the values obtained with the other two formulas (LogTF and RawTF) for \( tf \) without parameters were studied verifying that they are worse than the approximation BM25 (see Table 8: last two columns).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Permutations</th>
<th>Combinations</th>
<th>Porter-Stopwords NLM</th>
<th>Porter-Stopwords SMART</th>
<th>Krovetz-Stopwords NLM</th>
<th>Krovetz-Stopwords SMART</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k_1 )</td>
<td>0.6</td>
<td>1.2</td>
<td>0.6</td>
<td>1.2</td>
<td>0.6</td>
<td>1.2</td>
</tr>
<tr>
<td>( b )</td>
<td>0</td>
<td>1</td>
<td>0.65</td>
<td>0.6</td>
<td>0.55</td>
<td>0.7</td>
</tr>
<tr>
<td>( k_3 )</td>
<td>0.75</td>
<td>0.6</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
</tr>
</tbody>
</table>

Table 8. TF-IDF BM25 best parameters and results for RawTF and LogTF formulas.

In Tables 7, 8 (highlight columns) we can see that TF-IDF Okapi \( tf \) formula was better than others, and so we continued our study with this approximation. Many researchers use the BM25 algorithm in articles conducting their studies to retrieve information in the several fields of the same not only in the Abstract field. By this assumption, we tested how the MAP increases if we look for documents related to the queries in the Abstract, Title and Mesh fields using the BM25, TF-IDF BM25, TF-IDF LogTF and TF-IDF RawTF formulas, and also checked how the values obtained were related to values obtained for Cystic Fibrosis Collection by other authors [36–38] (see Table 9).

The results obtained with Okapi BM25 are consistent with those presented by Andrew Tortman in [37, 38] where shows a value of 0.2728 in MAP obtained with BM25 algorithm with Cystic Fibrosis Collection.

**Relevance Feedback**

The Lemur toolkit implements a simplified Rocchio feedback algorithm. Let \( R \) and \( \bar{R} \) be the relevant document set and non-relevant document set respectively, to be used for feedback. The rocchio algorithm
Table 9. Abstract, Title and MeSH fields.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>Algorithms</th>
<th>BM25</th>
<th>TF-IDF</th>
<th>BM25</th>
<th>LogTF</th>
<th>RawTF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.2717</td>
<td>0.2953</td>
<td>0.2683</td>
<td>0.2209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.2733</td>
<td>0.2930</td>
<td>0.2655</td>
<td>0.2221</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.2719</td>
<td>0.2929</td>
<td>0.2684</td>
<td>0.2208</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.2737</td>
<td>0.2904</td>
<td>0.2654</td>
<td>0.2228</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

simply “moves” the query vector closer to the centroid vector of \( R \) and away from the centroid vector of \( \bar{R} \):

\[
\vec{q}_{new} = \vec{q}_{old} + \alpha \vec{d}_R - \beta \vec{d}_{\bar{R}} \tag{16}
\]

\( \vec{d}_R \) is the centroid vector of all weighted document vectors in \( R \)
\( \vec{d}_{\bar{R}} \) is the centroid vector of all weighted document vectors in \( \bar{R} \)

In Lemur toolkit do not exist \( \bar{R} \) because it only use the centroid of relevant documents (\( \beta = 0 \)). It uses a positive feedback algorithm.

Parameters for Rocchio:

\( M \) number of documents in feedback [10-100]
\( K \) number of terms selected in feedback [10-100]
\( \alpha \) coefficient adjustment (0,4]

Table 10. Finding the best \( M \) parameter for TF-IDF BM25 feedback.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>Parameters</th>
<th>( M )</th>
<th>10</th>
<th>5</th>
<th>15</th>
<th>30</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Baseline</td>
<td>( K )</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>( \alpha )</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.1868</td>
<td><strong>0.2079</strong></td>
<td>0.1991</td>
<td>0.2070</td>
<td>0.2053</td>
<td>0.2035</td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.1898</td>
<td><strong>0.2075</strong></td>
<td>0.2026</td>
<td>0.2053</td>
<td>0.2052</td>
<td>0.2021</td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.1843</td>
<td><strong>0.2094</strong></td>
<td>0.1974</td>
<td>0.2007</td>
<td>0.2025</td>
<td>0.2007</td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.1866</td>
<td><strong>0.2074</strong></td>
<td>0.1998</td>
<td>0.2022</td>
<td>0.2056</td>
<td>0.2029</td>
<td></td>
</tr>
</tbody>
</table>

Table 11. Finding the best \( K \) parameter for TF-IDF BM25 feedback.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>Parameters</th>
<th>( M )</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( K )</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>25</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( \alpha )</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.2079</td>
<td>0.2103</td>
<td>0.2117</td>
<td>0.2102</td>
<td>0.2113</td>
<td><strong>0.2116</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.2075</td>
<td>0.2083</td>
<td>0.2105</td>
<td>0.2087</td>
<td>0.2055</td>
<td><strong>0.2100</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.2094</td>
<td>0.2068</td>
<td>0.2075</td>
<td>0.2068</td>
<td>0.2077</td>
<td><strong>0.2093</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.2074</td>
<td>0.2061</td>
<td>0.2015</td>
<td>0.2032</td>
<td>0.2050</td>
<td><strong>0.2033</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The best values obtained for Abstract Relevance feedback (see Tables 10, 11, 12) were: \( M = 10, \ K = 28, \) and \( \alpha = 0.5 \) obtaining a MAP value between 0.20 and 0.21. We tested how the MAP increases if we look for documents related to the queries in the Abstract, Title and Mesh fields using the TF-IDF BM25...
Table 12. Finding the best $\alpha$ parameter for TF-IDF BM25 feedback.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>$M$</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>10</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.5</td>
<td>0.1</td>
<td>1</td>
<td>0.9</td>
<td>0.4</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.2116</td>
<td>0.1957</td>
<td>0.2103</td>
<td>0.2099</td>
<td>0.2105</td>
<td>0.1972</td>
<td></td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.2100</td>
<td>0.1956</td>
<td>0.2035</td>
<td>0.2091</td>
<td>0.2093</td>
<td>0.1956</td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.2093</td>
<td>0.1904</td>
<td>0.2072</td>
<td>0.2097</td>
<td>0.2089</td>
<td>0.1907</td>
<td></td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.2033</td>
<td>0.1932</td>
<td>0.2024</td>
<td>0.2031</td>
<td>0.2061</td>
<td>0.1918</td>
<td></td>
</tr>
</tbody>
</table>

relevance feedback algorithm (see Table 13) obtaining a MAP value between 0.33 and 0.35. These results can be compared with those obtained by K. Shin and S.Y. Han in their expansion system presented in [39] where achieved a maximum value of 0.35 for R-prec, being our R-Prec of 0.37 or 0.38 in some case.

Table 13. TF-IDF BM25 relevance feedback best parameters for Abstract, Title and MeSH fields.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>MAP</th>
<th>R-Prec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.3468</td>
<td>0.3780</td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.3391</td>
<td>0.3731</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.3475</td>
<td>0.3834</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.3435</td>
<td>0.3790</td>
</tr>
</tbody>
</table>

MeSH Terms in Information Retrieval combined with Abstract

Many authors have worked with MeSH and abstract fields to retrieve information, so we focused this part of our research to improve the expansion obtained by the abstract with the MeSH field [39]. As mentioned before, MeSH Terms index journal articles for subject analysis of biomedical literature NLM. It imposes uniformity and consistency to the indexing of biomedical literature. MeSH has a hierarchical structure with sets of terms naming descriptors that allow searching at various levels of specificity. Expert annotators, based on indexed content of documents, assign MeSH Headings terms to the documents in order to allow to the user retrieve the information that explains the same concept with different terminology. On average, 5 to 15 subject headings are assigned by document, 3 to 4 of them being Major Headings and the others being Minor Headings. We studied the advantage of using the MeSH field respect to the Abstract field (see Table 14: 2 column). Mesh search field did not benefit our retrieval, and the feedback obtained is worse than the original result (see Table 14: 3 column). Add MeSH terms to the query with feedback was not advantageous in this case.

Table 14. TF-IDF BM25 different fields MAP

<table>
<thead>
<tr>
<th>Combinations</th>
<th>AB</th>
<th>MH</th>
<th>MH Feedback</th>
<th>MH QH</th>
<th>MH QH Feedback</th>
<th>AB FB with MH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM</td>
<td>0.1868</td>
<td>0.1841</td>
<td>0.1820</td>
<td>0.1948</td>
<td>0.1918</td>
<td>0.2061</td>
</tr>
<tr>
<td>Porter-Stopwords SMART</td>
<td>0.1898</td>
<td>0.1813</td>
<td>0.1773</td>
<td>0.1889</td>
<td>0.1823</td>
<td>0.2039</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM</td>
<td>0.1843</td>
<td>0.1853</td>
<td>0.1837</td>
<td>0.1926</td>
<td>0.1924</td>
<td>0.2006</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART</td>
<td>0.1866</td>
<td>0.1826</td>
<td>0.1782</td>
<td>0.1863</td>
<td>0.1819</td>
<td>0.2005</td>
</tr>
</tbody>
</table>
Then a new form of expansion was tested consisting in search the MeSH Headings of the Entry Terms in queries and launching queries against the field MeSH of documents. It is associated with the Entry Terms that are in the query and are replaced by the Preferred Terms would be the Descriptors (we used PubMed in order to obtain descriptors as commented above) contained in the MeSH field of documents, either Major or Minor. Our expansion method was based on work of Kwangcheol Shin and Sang-Young Han in their expansion system presented in [39] proving the advantage of using MeSH Headings (based on a previous study of MeSH and its structure) to expand the query instead of working with terms that are not related to the MeSH field on documents (see Table 14: 4 column Queries Headings). Feedback is non-relevant again (see Table 14: 5 column Queries Headings Feedback).

Documents retrieved with the MeSH process were selected (see Table 14: 5 column Queries Headings) to make feedback and see if better results were obtained, and found that there is not much variation between the Abstract feedback with documents previously retrieved in the same field or feedback with the documents retrieved from the MeSH process (see Table 14: 6 column - Abstract Feedback with MeSH). We proceed to isolate the process of Lemur to see if we improve this list of documents retrieved in MeSH.

**Query expansion outside Lemur**

As Lemur is closed tool and we performed weighting operations on the terms of the documents, follow the rest of our tests out of it. To perform our tests out of Lemur, we first adapted the queries and MeSH field documents to conduct searches of MeSH Headings in documents. In the first run we worked only with frequencies of occurrence of the query term within documents without distinguishing whether they are part of the Major or Minor terms in them. The total weight of a document for a particular query is the sum of the frequencies of the different terms in query inside the document. As we commented before, proceeded to access at online tool PubMed for locate possible sentences of MeSH Heading terms within the queries, which matching with Entry Terms, and add the Descriptors (MeSH Headings) matched to the new queries. We realized that two of the queries have not related MeSH terms and we only retrieved results for 93 queries. MAP in this moment was of 0.1570.

At this point, we performed a test to try to expand with the long form the acronym CF “Cystic fibrosis” queries, previously discussed in expansion into Lemur in paragraph related to Acronyms. We inserted it in all of queries where appears, previously we had not included it in order to reduce noise. By this way, we wanted to check whether the MAP increases with MeSH terms for all queries. We obtained a MAP of 0.1272, then we could see that MAP decreases and this was not beneficial.

As previously discussed, Major MeSH terms describing the primary/main topics of the document and Minor MeSH terms giving more details/secondary content about it. For this reason in the next test, we gave values to the MeSH terms depending on whether they were on Major or Minor fields into documents. If the term was a Major term in the document, we gave it a weight of 5 and if the term was a Minor term of 1. The other proceedings did not vary in relation to previous test. We obtained a MAP of 0.1641 and processed 93 out of 100 queries - rest of them had not results.

From the above results, we concluded that many MeSH terms may be contained in substrings within the Major and Minor Terms of documents, so we proceeded to find substrings instead of an identical mapping between them. We achieved a MAP of 0.1739. In this case, we retrieved results for 98 queries.

To improve the system, we proceeded to do the mixing of the results obtained with the procedure in MeSH fields calculated with the results obtained in previous section for the Abstract field. We prioritized those documents on both lists and then did re-ranking of documents in order to place them in the resulting list according to their total weight (see Table 15). At this point we have improved our technique with the values obtained in Table 14, last column.

We tried to give more weight to the list of MeSH terms to test what would be a new way of giving priority to expanding the list but results are worst, probably due to the fact that MeSH do not retrieve results for all queries (see Table 16).

Finally, we added Title field in order to see how improved results (see Table 17) These results can be compared with those obtained by K. Shin and S.Y. Han in their expansion system presented in [39] where achieved a maximum value of 0.35 for R-prec, being our R-Prec of 0.31 or 0.32 in some case. Our results are located between the two highest values.

Finally, a Recall-Precision Graph (see Fig. 5) have been included showing the improvement obtained with the last expansion method using the MeSH, Abstract and Title fields (see Table 16) with respect to the retrieve of documents using only the Abstract field with TF-IDF BM25 (see Table 8). Show results
Table 15. MeSH and Abstract fields.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>MAP</th>
<th>R-Prec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM Abstract</td>
<td>0.2344</td>
<td>0.2737</td>
</tr>
<tr>
<td>Porter-Stopwords SMART Abstract</td>
<td>0.2341</td>
<td>0.2778</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM Abstract</td>
<td>0.2344</td>
<td>0.2747</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART Abstract</td>
<td>0.2352</td>
<td>0.2723</td>
</tr>
</tbody>
</table>

Table 16. MeSH and Abstract fields plus weighting MeSH list.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>MAP</th>
<th>R-Prec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM Abstract</td>
<td>0.2300</td>
<td>0.2679</td>
</tr>
<tr>
<td>Porter-Stopwords SMART Abstract</td>
<td>0.2296</td>
<td>0.2708</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM Abstract</td>
<td>0.2314</td>
<td>0.2698</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART Abstract</td>
<td>0.2303</td>
<td>0.2707</td>
</tr>
</tbody>
</table>

for the combination Krovetz-SMART where curves closest to the upper right-hand corner of the graph (where recall and precision are maximized) indicate the best performance.

4 Conclusions and Future work

We have developed and evaluated query expansion techniques for retrieving documents in several fields of biomedical articles belonging to the corpus Cystic Fibrosis, a corpus of MEDLINE documents. We have tested the benefit of using stemming and stopwords in the query expansion following the investigations of other authors. Besides were analyzed the benefits that could be obtained from the use of acronyms, not satisfactory in our case by the characteristics of the corpus.

We compared the weighting algorithms Okapi BM25 and TF-IDF availables in the Lemur tool, concluding that TF-IDF with $tf$ formula given by BM25 approximation is superior in its results.

Document retrieval based on Abstract, MeSH and Title fields seems more effective than looking at each of these fields individually. Also, the use of feedback in retrieving of documents, a technique widely used by researchers in this field, results a great improvement in retrieving the relevant documents, based on this case in Rocchio, allowed us to obtain good results improving MAP and other measures.

We performed a study to improve searching in the Abstract field with the information available in the MeSH field on documents. For this, we have enhanced queries locating Entry terms in them and obtaining MeSH Headings in PubMed in order to map with the documents, giving greater importance to those belonging to Major Terms in documents. The results have been good at making use of the Title and Abstract fields to improve the list of documents retrieved in the MeSH field compared to other research. Queries without Entry Terms or with few related documents produced a negative impact on our retrieval process based on MeSH.

Table 17. MeSH, Title and Abstract fields.

<table>
<thead>
<tr>
<th>Combinations</th>
<th>MAP</th>
<th>R-Prec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Porter-Stopwords NLM Abstract</td>
<td>0.2815</td>
<td>0.3234</td>
</tr>
<tr>
<td>Porter-Stopwords SMART Abstract</td>
<td>0.2780</td>
<td>0.3182</td>
</tr>
<tr>
<td>Krovetz-Stopwords NLM Abstract</td>
<td>0.2822</td>
<td>0.3208</td>
</tr>
<tr>
<td>Krovetz-Stopwords SMART Abstract</td>
<td>0.2798</td>
<td>0.3191</td>
</tr>
</tbody>
</table>
As future work, stemming could be used when working with expansion technique based on Entry Terms following the research of K. Shin and S.Y. Han [39]. Working with MeSH Headings and MeSH fields in documents, the mapping between terms should be sufficient. Also it would be interesting test whether reduce the root of MeSH Headings in queries and documents could improve the results.

Another method that we believe interesting would be using clustering techniques in order to expand queries [1]. It is based on the idea of co-occurrence of stems (or terms) inside documents (stems which co-occur frequently inside documents, those which distance between them is small inside documents or those with similar neighborhoods have some synonymity relationship). The technique of cluster is a feedback technique in which, once retrieved relevant documents to a query, terms are localized inside the documents with higher frequency, or with a fixed distance or those that appear very often together, are used for improving initial query.

On the other hand, we intend to study the possibility of using other tools for indexing and retrieval of information that may offer advantages over Lemur. Some have already been discussed as Zettair, Smart or Terrier but now we want to study the possibility of creating our own tool for this purpose.

To continue our studies related to MeSH terms, we want to make QE using Entry terms. To do this, we must work on documents and queries by modifying the preprocessing and indexing of Lemur. In actual tests, we lose concepts composed of several words or symbols that are of vital importance when working with biomedical issues.

Finally, we want to implement query expansion techniques applying different scientific dictionaries used in many investigations as: EntrezGene, HUGO, Eugenes and ARGH for gene/protein names; GO for molecular functions, biological processes and cellular components; UMLSKS for related names and symbols or WordNet for general words/phrases.
All tests so far and future will be focused on TREC\textsuperscript{13}. TREC 2005 is an ad hoc retrieval task collection where we have queries derived from a generic topic template which includes a certain number of semantic types. TREC has different corpus depend on year. Queries in the 2005 ad hoc retrieval task were collected from real biologists. It has 50 queries derived from 5 generic topic templates (GTT), each of which has 10 instances. TREC 2005 has 10-year Medline subset, which includes 4,591,008 documents from year 1994 to 2003. The corpus has the same components that Cystic Fibrosis collection. TREC is a large corpus that will allow generalize results of different expansion techniques.

References


\textsuperscript{13} TREC \url{http://trec.nist.gov/}
5 Resumen

El conocimiento biomédico crece a pasos agigantados alimentando grandes colecciones de documentos científicos. Estas colecciones de documentos ofrecen una excelente oportunidad para extraer conocimiento biomédico aplicando técnicas de recuperación de información, que debe estar representada de un modo que vaya a permitir un buen acceso a ella por parte de los usuarios.

Un sistema de recuperación de información textual debe ser capaz de procesar la consulta o petición de información de un usuario, de manera que extraiga las palabras clave necesarias para recuperar los documentos relevantes a esta consulta, es decir, aquellos que están relacionados con lo solicitado por el usuario. Para validar un sistema de recuperación de información utilizamos corpus formados por documentos, consultas y un fichero con los documentos que los expertos han considerado relevantes a cada consulta. De este modo verificamos los resultados obtenidos de las búsquedas sobre el corpus.

Muchas veces los conceptos introducidos por los usuarios en las consultas no van a coincidir con los conceptos que aparecen en los documentos con los que están relacionados, por eso, las técnicas de expansión de consultas son de vital importancia ya que permiten obtener los términos clave que se necesitan para recuperar los documentos de una manera más precisa. A lo largo de nuestro artículo hemos analizado varias técnicas de expansión de consultas para evaluar su eficiencia.

La recuperación de la información está compuesta por dos procesos principales: el indexado y el mapeo.

1. El proceso de indexado o parseado de consultas trata de reducir los términos de una consulta a las palabras clave relevantes para asociar documentos con consultas. En este proceso, para la expansión de consultas son vitales las técnicas de stemming y stopwords para reducir las variaciones léxicas de las palabras a su raíz y eliminar aquellas que no van a ser relevantes, de manera que todas las palabras clave que partan de una misma raíz se unifiquen. Antes de realizar el parseado de las consultas, se realizan métodos de expansión, como el añadir a las consultas sinónimos o palabras relacionadas, para representar los mismos conceptos de diferentes modos ya que de antemano no se sabe con qué palabra exacta se va a representar un concepto en un documento determinado.

2. El proceso de mapeo consiste en utilizar algoritmos de pesado como pueden ser el BM25 o TF-IDF que nos van a pesar los documentos en función de los términos que contienen, relacionados con una consulta determinada, pesando más aquellos documentos que tienen más términos relacionados con la consulta y menos aquellos que sólo van a tener un emparejamiento parcial.

Una vez que hemos obtenido los documentos relevantes a las consultas, podemos realizar otro proceso de expansión que va a consistir en añadir a la consulta inicial aquellos términos de más peso dentro de los documentos, de esta manera enriquecemos la consulta para que se vaya a acercar más a conceptos que tienen valor dentro de los documentos relevantes a la misma para así poder recuperar documentos más cercanos al tema concreto a tratar por la consulta.

Existen diferentes herramientas para la recuperación de información. En nuestro artículo nos centrarnos en Lemur. Por otro lado, el corpus que centra nuestra investigación es el Cystic Fibrosis, formado por 1239 documentos y 100 consultas, además del fichero de relevancia dado por los expertos. Cada uno de los documentos va a tener varios campos, destacando el Abstract, Título y términos MeSH. Los MeSH Headings son términos (asignados por expertos) que representan el contenido de un documento pudiendo distinguir entre Major terms y Minor terms. Los major terms son 4 o 5 MeSH Headings que van a representar de manera principal el contenido del artículo, mientras que los Minor terms son conceptos que van a servir de base para completar la información acerca del contenido del mismo. Los Major terms definirán la temática concreta mientras los Minor nos van a reflejar temas secundarios tratados en ellos.

Además, a la hora de tratar una consulta o documento, va a ser de vital importancia el reconocimiento de acrónimos, ya que tenemos que tener en cuenta un posible mapeo entre estos y sus formas largas, por lo que muchas veces son de vital importancia en expansión ya que no sabemos de qué forma pueden estar representados en documentos y consultas, y por ello se ha de trabajar buscando ambas formas. En nuestro artículo mostramos que trabajar con reconocimiento de acrónimos no es beneficioso, ya que el corpus no posee un gran número de ellos, por lo que el reconocimiento o la expansión ensucia los resultados.
Nuestro proceso de recuperación de información utilizando expansión de consultas está dividido en dos partes: una de ellas realizada en Lemur y otra realizada fuera de Lemur. Se ha decidido realizarlo así debido a que una parte de la expansión de consultas está centrada en dar diferentes pesos a términos MeSH de las consultas presentes dentro de los documentos en función de si son Major o Minor terms y, dado que Lemur es una herramienta cerrada que no nos va a permitir hacer este tipo de proceso, por lo que llegados a este punto de expansión procedimos a sacar este proceso de la herramienta.

La expansión dentro de Lemur comienza analizando las ventajas de utilizar stemming y stopwords así como del uso de los acrónimos en documentos y en consultas. A continuación se analiza la parametrización adecuada de los algoritmos de pesado utilizados que nos va a permitir obtener los mejores resultados para nuestra colección. Una vez que tenemos la parametrización adecuada (para lo que hemos usado el campo Abstract), obtenemos resultados para el campo MeSH con los mismo algoritmos, que nos van a servir de base para analizar los resultados de retroalimentar el Abstract con los términos de los documentos recuperados previamente para este campo o realimentarlo con los resultados obtenidos con el uso del campo MeSH. Fuera de Lemur vamos a analizar la posibilidad, anteriormente citada, de dar distintos pesos a los términos en función del campo en el que aparecen, para obtener el ranking de documentos. Esta lista se va a mezclar con la lista obtenida previamente en Lemur de documentos obtenidos buscando en el campo Abstract, priorizando aquellos que aparecen en ambas listas. Para mejorar el resultado final se amplía este proceso combinando lo anterior también con el campo Title, demostrando los resultados que la mejor combinación es usar el campo MeSH con mayor pesado para Major terms junto con el Abstract y el Title.