Probabilistic Query Expansion Method Based on a Query Recommendation Algorithm

Btihal El Ghali[1], Abderrahim El Qadi[2], Omar El Midaoui[1], Mohamed Ouadou[1], Driss Aboutajdine[1]
[1]LRIT Associated Unit to the CNRST - URAC n°29
Faculty of Science
Mohammed V-Agdal University
Rabat, Morocco
[2]TIM, High School of Technology
Moulay Ismaïl University
Meknes, Morocco
{btihal.elghali, omarelmidaoui, ouadou55}@gmail.com, aboutaj@fsr.ac.ma, elqadi_a@yahoo.com

ABSTRACT: *Query Expansion Methods are proposed to solve many problems of information retrieval systems, but most of these methods do not use the information of interactions between the users and the system. In our approach, we applied a Query Recommendation Algorithm on a list of past user queries, to extract the most associated queries to the input query, and used it in a Probabilistic Query Expansion method, that is constructed as a language model to search in set of candidate terms for the most relevant terms for the initial query that we have to expand. The output of this approach is a table of terms that are candidates for the expansion of the user query, and their values of correlation with the whole query. We did our experiments using the database CISI of the standard collection of test SMART. The results show that the best values are reached by adding fifteen terms to the input queries, using the five most relevant documents of the input query and the five most relevant documents for each of the recommended queries used, for short and long queries. Our experiments shows also, that for short queries we need to use just the best recommended query in the process of expansion to have a very high value of the Interpolated Average Precision (IAP), but concerning long queries we need to employ also the second best recommended query to have the highest value of IAP. The final experiment was a comparison between our approach and the Rocchio Algorithm, and it shows that our approach gives best results except in the case of using four Recommended Queries (RQs) for short input queries.*

Keywords: Query Expansion, Query Recommendation, Probabilistic Method, Past User Queries

Received: 19 October 2012, Revised 18 December 2012, Accepted 21 December 2012

1. Introduction

The main problematic in Information Retrieval is the huge amount of data available on the World Wide Web, the enlargement of the accessibility of the internet, and the lack of organization of the information in it. In addition to that, users usually submit queries that contain ambiguous terms, these queries can retrieve documents that do not correspond to the field of research of the user and may lead him to error if he does not have specific information about what he is searching or if he is not expert in the domain of his research.

Another issue is the fact that, from an author to another the vocabulary used to mean the same thing varies considerably, without neglecting that users tend generally to submit queries containing terms that are different from the web documents terms.

The length of the queries also represent a big problematic in information retrieval. It is observed generally that web users submit very short queries and that the average length of web queries is two words or less [1]. Short queries usually do not have enough words to cover useful search terms and thus affect the search performance negatively.

According to Baziz [2], retrieving relevant documents by using the initial user query is an almost impossible task, because of the increasing volume of information bases.

To solve these problems and improve the effectiveness of the information retrieval system, the techniques used are the Query Expansion methods that involve the reformulation of queries by adding new terms to the initial user query, and the Query Recommendation methods that propose the nearest queries to the initial query, taking as an input a set of queries.

Query Expansion traditional approaches, were focused on expanding the query with terms extracted from various sources of information. In the case of Global Expansion techniques, the goal [3] is to complete the user query using the global information extracted from the whole corpus. This information is used in the form of a Thesaurus [4] (ie. A data structure that stores associations between terms) such as WordNet [2]. The second type is the Local Expansion techniques, which extract the expansions terms from the top-ranked documents [5].

However, these methods of expansion were limited in the extraction of expansion terms from a document collection, and have not used information about interactions between the user and the system; this is the case of the method that we will use in this paper, and that use the set of user past queries.

Using the terms of the most recommended queries and there relevant documents, we choose the most associated terms with the query of the user, and that allows the generation of an expanded query that respond to the need of information of the user.

We summarize the query expansion process presented in this paper by the following three steps:

- Extraction of information, by the construction of a bipartite graph called the graph of click, and whose edges connect a past query with their relevant documents. Then, the representation of queries as vectors of documents taking into account only the information of the graph,

- The application of the developed query recommendation algorithm, which uses as an input the vectors generated from the graph of clicks and the vectors of the terms contained in past queries, and which classify the past user queries according to their degree of association with the query to expand.

- The expansion of the initial query by the terms that are judged the most associated with the whole user query. These terms are extracted from the relevant documents of the initial query, its most recommended queries and their relevant documents.

Among the different approaches that exist in Query Expansion, we relied on the technique of queries recommendation proposed in [6], by applying to it some modifications, concerning the weighting function and similarity expressions used, and the Query Expansion method used in [7] [8].

2. Description of the Query Expansion Method

The past user queries has always been stored in the archives of web sites that contain a search engine. These information have been widely studied with various techniques of web mining in recent years to improve the effectiveness of information retrieval systems and the usability of search engines [8] [9].

In what follows, we consider that the top-ranked documents returned by the system for a query, are a relevant documents to this query.

In the graph of figure 1, we assume that the relationship between queries and their relevant documents is a bipartite graph, where: If two queries are linked with the same relevant document, we believe that these queries are associated with each other in some way, and the terms contained in the associated query and it's relevant documents can be used as candidate terms for the expansion of the other query.

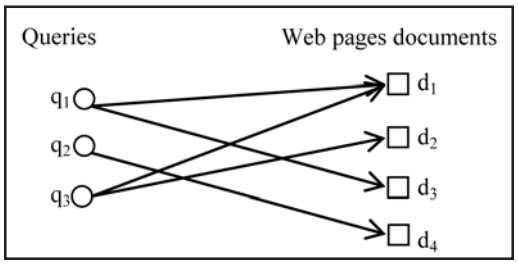

Figure 1. Query – Documents representation as a bipartite graph

We exploit the graph information by representing each query as a document vector $q_i=\{d_1, d_2, \ldots d_N\}$, whose elements describe the presence or absence of a document in the set of relevant documents to this query.
For a query q_i, $d_j = 1$ if the document D_j is relevant to it, otherwise $d_j = 0$.

We also represent each query as a vector of terms $q_k=\{t_1, t_2, \ldots t_M\}$ depending on the presence or the absence of the terms contained in the initial query and the past user queries. Where $t_1 = 1$ if the term T_1 is contained in the query, otherwise $t_1 = 0$.

The queries vectors built in this step, are used as an input for the Query Recommendation Algorithm. In this Algorithm, we used the weighting function LTC and the expression of similarity Cosine, because those functions were judged the best in a previous work that we did [10], and where we compared the best known similarity measures and the most commonly used weighting functions.

2.1 Weighting function: LTC
Most of the approaches proposed in information retrieval consider that the weighting is a fundamental step [11].

In our approach, we calculate the weight of a document or a term for a query, which reflects the degree of relevance of this document for this query, or the degree of presence of this term in the query. While, The weighting functions are usually used to measure the weight of a term in a document.

In this work, we chose the weighting scheme LTC [12] that is constructed by combining two factors:

- A local weighting factor (number of occurrences of the term in the document), which reflects the importance of the term within a document.

- A second factor, of global weighting, which is the value of the importance of the term throughout the whole collection whose formula is: $\log(N/n_i)$ with N the size of the collection and n_i the number of documents where appears the term t_i.

The formula of this weighting function for a document $D_j = [r_{1j}, r_{2j}, \ldots r_{mj}]$ and a term t_i is as follows [12]:

$$r_{ij} = \frac{\log(tf_{ij} + 1) \times idf_{ij}}{\sqrt{\sum_{k=1}^{m} [\log(tf_{kj} + 1) \times idf_{kj}]}} \quad (1)$$

By using the logarithm of the local weighting factor, which represents the frequency of appearance of a term in the document, this expression reduces the effects of large differences in frequencies.

2.2 Similarity measure: The Cosine Similarity

The objective of the measuring of similarities between the queries, represented as a document vectors, is to search for queries that have many common relevant documents, and then make the link between the queries and their relevant documents. In the other hand, the aim of computing similarities between queries, using their term vectors representation, is the search for queries that have an important number of common terms, to consider that these queries are associated depending on the terms that they contain.

In a previous work that we have done [10], the cosine similarity was judged as the best similarity measure. Thus, we chose to use this similarity measure, which is a measure that calculates the similarity between two vectors by determining the angle between them.

The expression of similarity based on cosine is:

$$Simc(\vec{q_i}, \vec{q_j}) = \cos(\vec{q_i}, \vec{q_j}) = \frac{\vec{q_i} \times \vec{q_j}}{|\vec{q_i}| \times |\vec{q_j}|} \quad (2)$$

This expression consider that two objects (queries for example) are similar if their vectors are confounded [13]. Otherwise, the two objects are not similar and their vectors form an angle $(\vec{q_i}, \vec{q_j})$, whose cosine is the value of the similarity.

2.3 Query Recommendation Algorithm

In this paper, we used an algorithm [6] that finds the appropriate group of related queries to the input query and classify them according to their scores that represent their relevance to it. The interest of a related query is measured by its rank (score), which is obtained by combining the following concepts:

- The similarity of the query with the input query, that is measured using the expression of similarity cosine defined above.
- The support of the query, which is a measure of the relevance of the query in the collection.

The similarity and the support of a query can be normalized and then combined linearly to generate the rank of the query.

The Query Recommendation Algorithm steps are as follows:

1. We built two vectors for each query in the space of queries $Q_j = \{D_1^{(j)}, D_2^{(j)}, \ldots, D_n^{(j)}\}$ and $Q_j = \{T_1^{(j)}, T_2^{(j)}, \ldots, T_m^{(j)}\}$, where $D_i^{(j)}$ is the weight of the i^{th} document in the documents vector, and $T_i^{(j)}$ represents the weight of the i^{th} term for the same query in the terms vector. These weights are defined by classic weighting measure LTC:

$$D_i^{(j)} = \frac{\log(tf_i^{(j)} + 1) \times idf_i^{(j)}}{\sqrt{\sum_{k=1}^{m} [\log(tf_k^{(j)} + 1) \times idf_k^{(j)}]}} \quad (3)$$

with: $idf_i^{(j)} = \log(N/n_i)$

With $tf_i^{(j)}$ is the frequency of the i^{th} document in the documents vector Q_j, N is the total number of queries in the collection and n_i is the number of queries for which the i^{th} document is relevant.

We calculate each $T_i^{(j)}$ with the same expression.

2. We measure the similarities between our initial query and all the past user queries by using the expression of similarity Cosine defined in (2.2).

3. We compute the support of each query q_j, which we measured as the ratio of the number of relevant documents to the query q_j and the total number of relevant documents for all the queries of the collection, by the expression:

$$sup(q_j) = \frac{N_{RDs}^{(q_j)}}{\sum_{k=1}^{N_q} N_{RDs}^{(q_k)}} \quad (4)$$

With N_q is the number of queries and $N_{RDs}^{(q_j)}$ is the number of relevant documents to the query q_j.

Then we calculate the support for these queries in the case of the terms vectors representation.

(4) We classify the queries based on their ranking score that we compute by using the two representations of each query. The ranking score based on documents vectors is as follow:

$$Rank_D(q_j) = [\alpha \times Sim_D(q_j, q_i) + (1 - \alpha) \times sup_D(q_j)] \quad (5)$$

Our contribution in this algorithm takes into account also the use of the queries terms, and the ranking score based on terms vectors is measured as follow:

$$Rank_T(q_j) = [\beta \times Sim_T(q_j, q_i) + (1-\beta) \times sup_T(q_j)] \quad (6)$$

In our Query Recommendation Algorithm, we will use the combination of these two ranking scores. Thus, the recommendation score of the query q_j for the initial query q_i is measured by:

$$Rank_T(q_j) = [\alpha \times Sim_D(q_j, q_i) + (1-\alpha) \times sup_D(q_j)] + [\beta \times Sim_T(q_j, q_i) + (1-\beta) \times sup_T(q_j)] \quad (7)$$

Where α and $\beta \in [0, 1]$, and they are used for normalization.

2.4 Language Models

The concept of language model [14] [15] [16] [17] [18] is exploited in the field of information retrieval, in order to model the relation of relevance between a document and a query. Thus, we bind the relevance of a document for a query to the probability that the query can be generated by the language model of the document.

Considering this definition, a document *D* is considered as a sublanguage for which we build a language model M_D.

The relevance score of a document *D* for a query *Q* is estimated by:

$$Score(Q, D) = P(Q \mid M_D) \quad (8)$$

Knowing that a query is a sequence of words t_i, $Q = t_1, t_2 \ldots t_n$. This score can be written by:

$$Score(Q, D) = P(t_1, t_2 \ldots t_n \mid M_D) \quad (8)$$

This formulation of the problem of information retrieval makes it similar to the statistical modeling of the language. Thus, the techniques developed for the language can be implemented in the information retrieval.

The main problem that occurs for the language models is due to the fact that the size of the training corpus, can't reach the size of a language. This causes an "*under-representation of data*", because the absent words in the training corpus are estimated by a null probability. Therefore, a null probability is assigned to any sequence of words containing that word.

To solve the problem of "*under-representation of data*", the researchers have focused in "*Smoothing*".

The Smoothing aims to assign a not null probability to the absent words from the training corpus, by redistributing the probability mass observed. Several smoothing methods have been developed [16], the best known among them are:

- The Laplace smoothing;
- The Good-Turing (GT) smoothing;
- The Backoff smoothing ;
- The smoothing by interpolation (known as "*Jelinek and Mercer*" *JM*);
- Dirichlet smoothing;
- Absolute discount.

According to [17], the choice of the appropriate smoothing technique depends on the environment of experimentation.

The probabilistic method that we will present in (2.5) is a language model that we used to model the relation of relevance between the terms of the documents and the initial query of the user.

2.5 Query Expansion Method

The expansion method that we have chosen to use in combination with the query recommendation algorithm is the method proposed in [7], and in our work it takes as inputs the terms of the user query (initial query), and the candidate terms for the expansion of this query: the terms extracted from the relevant documents (RDs) of the initial query, the best recommended queries and their relevant documents (RDs). These terms are classified according to the whole query, using our language model that includes three steps:

1. We compute the weight of the terms of the documents $w_i^{(d)}$ using the same weighting function that we used in the Query Recommendation Algorithm, LTC, and it's formula is:

$$w_i^{(d)} = \frac{\log(tf_i^{(d)} + 1) \times idf_i^{(d)}}{\sqrt{\sum_{\forall D \in S} [\log(tf_i^{(d)} + 1) \times idf_i^{(d)}]}} \quad (10)$$

With S is the set of documents used for expansion (recommended queries are also considered as documents), $tf_i^{(d)}$ the frequency of the i^{th} term in the document D, and $idf_i^{(d)} = \log(N / n_i)$ where N represents the number of documents in the set S and n_i the number of document containing the term t_i.

We compute the weights of query terms $w_i^{(q)}$ similarly.

2. We measure the probabilistic correlations between every term of the initial query and documents terms (the terms contained in the RDs of the initial query, the most related queries to it and their RDs). The expression of the probabilistic correlation used is as follow:

$$P(w_j^{(d)} / w_i^{(q)}) = \sum_{\forall D \in S} P(w_j^{(d)} \mid D) \times P(D \mid w_i^{(q)}) \quad (11)$$

Where $P(w_j^{(d)} \mid D)$ represents the conditional probability of occurrence of the j^{th} document term if the document D is a relevant, and $P(D \mid w_i^{(q)})$ represents the conditional probability of the document D being relevent when the i^{th} query term appears in the user query.

These two conditional probabilities can be determined as follow:

$$P(w_j^{(d)} \mid D) = \frac{w_j^{(d)}}{\max_{\forall t_k \in D} w_k^{(d)}} \quad (12)$$

$$P(D \mid w_j^{(q)}) = \frac{f_i^{(q)}(w_j^{(q)}, D)}{f^{(q)}(w_j^{(q)})} \quad (13)$$

Where $f_i^{(q)}(w_j^{(q)}, D)$ Represents the number of queries in which the term $t_i^{(q)}$ appear and for which the document D is relevant, and $f^{(q)}(w_j^{(q)})$ represents the number of queries witch contain the term $t_i^{(q)}$.

3. By combining the correlation of a candidate term with all the terms of the initial query, we can calculate the cohesion weight of this document term, which represent the relationship between the document term and the whole query. The cohesion weight of a document term $t_j^{(d)}$ for a user query Q is measured by the formula:

$$CoWeight_Q(w_j^{(d)}) = \ln\left(\prod_{t_i^{(q)} \in Q} (P(w_j^{(d)} / w_i^{(q)}) + 1)\right) \quad (14)$$

This final step returns a classified list of weighted terms. The top-ranked terms are selected as expansion terms for the initial user query.

3. Experimental Results

In our experiments, we realized a java application to apply the Query Recommendation Algorithm and the Probabilistic Query Expansion Method. As a collection of test we used a database from the standard collection Smart: database CISI (Collection of documents abstracts in library science and related areas). This collection provides the user past queries and their ground of truth (a list of relevant documents to each query). Table 1 lists statistics about the database CISI.

# document	# queries
1460	111

Table 1. Statistics about the Database CISI

The input queries that we used are the four short queries (contain less than 5 terms) and the four long queries (contain more than 5 terms) that have the highest numbers of relevant documents in the database CISI (table 2).

N°Query	20	27	30	34	15	24	32	44
Terms number	4	5	4	5	10	11	12	27
Number of RDs	144	115	134	38	82	52	117	155

Table 2. Input queries

We used the information contained in this database, first to found the recommended queries to each input query using the Query Recommendation Algorithm that we presented in Section 2 (2.3).

3.1 Query recommendation for short and long queries

At the first, we built a document-weighted vector and a term-weighted vector for each query using the function LTC, and we calculated the scores of all past user queries with each input query using the Query Recommendation Algorithm.

Then, to show the difference between the influence of each score (the score based on terms vectors and the one based on documents vectors) on the global score (the score based on documents and terms vectors), we built a bar graphs (Figure 2 and Figure 3) using the scores of the best recommended query for every input query (The scores values are displayed on the table 3 and table 4).

Table 3, shows the three different scores for the best recommended queries of the four short input queries. We considered the global score to select the best recommended query of every input query.

N° Query	**N° Best Recommended Queries**	**Rank based on document vectors**	**Rank based on term vectors**	**Global score**
20	**104**	0,015	0,532	0,547
27	**28**	0,074	0,701	0,775
30	**25**	0,013	0,203	0,216
34	**27**	0,153	0,587	0,769

Table 3. The different scores for the best recommended query of the short queries

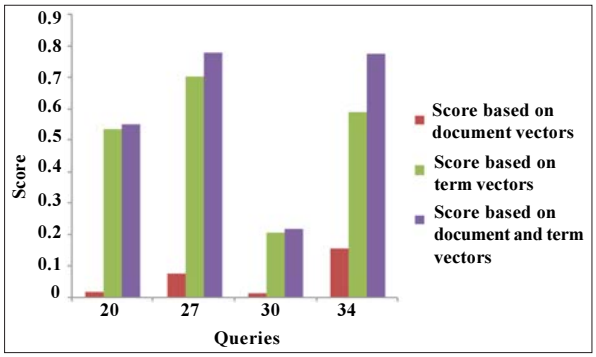

Figure 2. The comparison of the scores of the best recommended query for each short query

We notice from Figure 2, that the score based on terms influence the global score more than the score based on documents, because their values are very close, while the score based on documents have very minimal values.

In contrast, table 4 shows the three different scores for the best recommended queries of the four long input queries.

In the case of long queries, Figure 3 shows that the score based on terms influence the global score most of the time except for

the query number 32 whose score based on documents vectors is more dominant in the global score, that can be explained by the fact that the queries 32 and 46 may have a lot of common relevant documents and that influenced the value of the association between them even if they don't have many common terms.

We note that the score based on term vectors is the most dominant for most of the queries whether short or long.

N° Query	N° Best Recommended Queries	Rank based on document vectors	Rank based on term vectors	Global score
15	**25**	0,008	0,644	0,652
24	**21**	0,136	0,800	0,936
32	**46**	0,341	0,033	0,375
44	**45**	0,051	0,220	0,271

Table 4. The different scores for the best recommended query of long queries

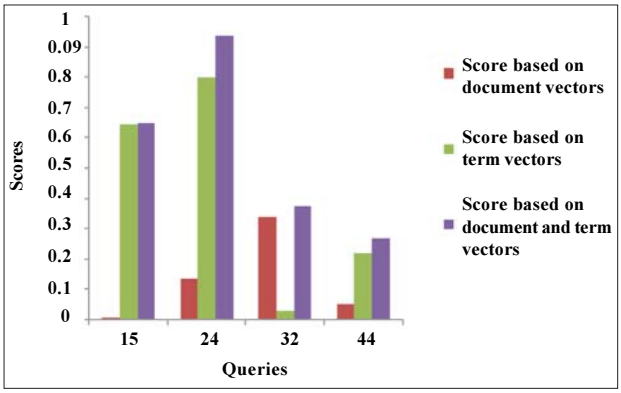

Figure 3. The comparison of the scores of the best recommended query for each long query

3.2 Query Expansion: Variation of the number of term expansion

In the second step, we applied the Probabilistic Query Expansion Method by using as inputs: the initial query (each query of the Table 2), its 5 top-ranked relevant documents, its best recommended query (RQ) and its 5 top-ranked relevant documents as well.

In this step, we varied the number of term expansion from five to thirty, in purpose to find the number of term expansion that gives in result a new query whose precision is the highest one, and to found a maximum threshold for expansion.

Before using the probabilistic method to expand the input queries, we had to do a pretreatment of the terms contained on the queries and the relevant documents used in the process of expansion. This pretreatment is divided on two steps, the first one is the elimination of the stop words and the second one is the stemming of the terms. To evaluate the performance of the retrieval system, we used the Interpolated Average Precision (IAP). We did a series of experiments to investigate the effects of the variation of the number of the expansion terms used to expand the input queries on the interpolated average precision. We expanded each of the eight input queries by adding the five terms that have the highest Coweight relatively to these queries. In the same way we added more and more terms to the initial queries (10, 15, 20, 25 and 30), until we had a resulted queries that was expanded with thirty terms. Figure 4 and Figure 5 show the impact of the number of expansion terms used on the IAP for short and long queries.

Figure 4, shows that for most of the input queries the IRS reached the best values of IAP by expanding the queries using fifteen terms. In the other hand, we notice in Figure 5 that the IAP reach its maximum threshold when we add twenty terms of expansion in the three first long queries and that have between 10 and 12 initial terms (Table 2), but the query 44 that contain 27 terms attained the highest value of IAP by adding just five terms, and it has retained this value when we added five other terms (10 terms of expansion in total).

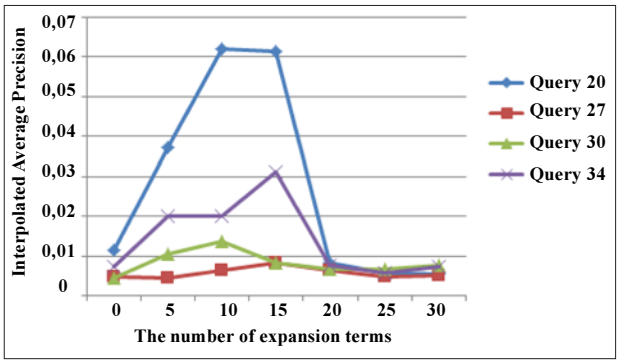

Figure 4. Variation of the number of expansion terms for short queries

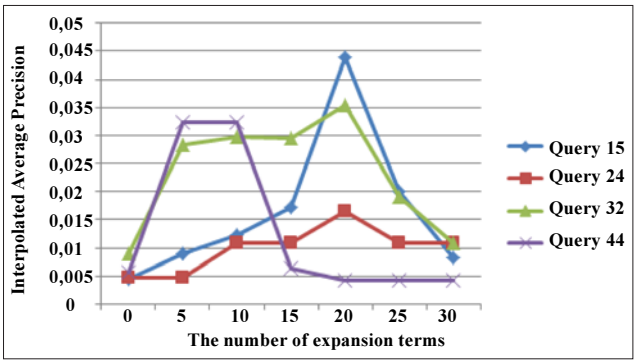

Figure 5. Variation of the number of expansion terms for long queries

To standardize the number of expansion terms used in the next steps of experimentation, we choose to add fifteen terms, whatever the initial length of the query is.

3.3 Query Expansion : Variation of the number of relevant documents

The third step of experimentation concerned the evaluation of our approach of expansion while the number of the top-ranked documents used is increasing.

For the expansion of each one of the eight initial queries, we used the fifteen terms of expansion that have the best values of Coweight according to the result of the previous step of experimentation, and we used the best recommended query and its relevant documents.

First, we used in our process of expansion the five top-ranked documents of the input query, its best recommended query, and the five top-ranked documents of the RQ. Then, in every new series of experiments we added five other top-ranked documents of the input query, and we also added five other top-ranked documents of its best RQ, until we had twenty relevant documents for the query and for its best RQ.

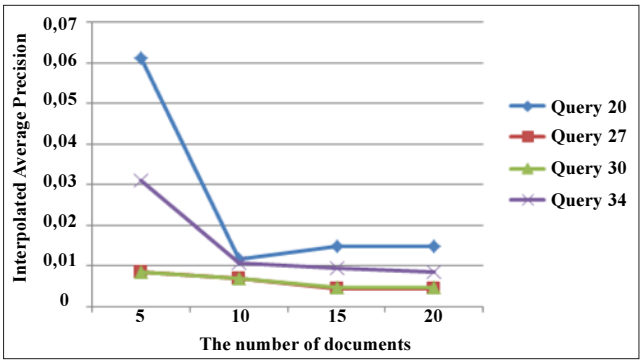

Figure 6. Variation of the number of documents used in the expansion process for short queries

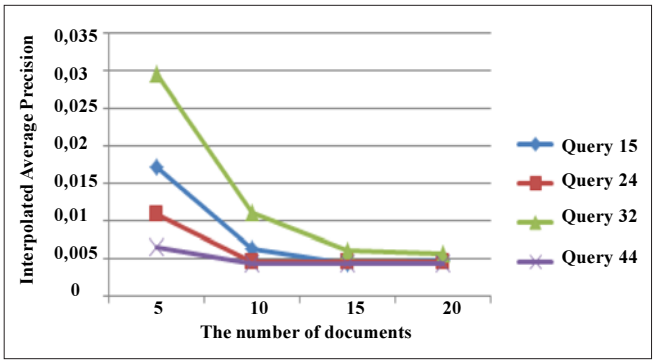

Figure 7. Variation of the number of documents used in the expansion process for long queries

Figure 6 and Figure 7show that while adding more and more relevant documents (RDs) to the process of expansion the value of the Interpolated Average Precision in decreasing, and that the best values are given by the use of the five top-ranked documents of the user query, its best RQ, and the five top-ranked documents of it.

In what follow, we used the results of this step of experimentations (Five documents for each query).

3.4 Query Expansion: Variation of the number of recommended queries

In this step, we search about the four best recommended queries for each input query to use them in the Probabilistic Query Expansion Method. The table 5 shows the input queries and their four best recommended queries.

N° Query	N° Queries (four Best Recommended)
Short queries	
20	104, 84, 65, 37
27	28, 34, 6, 100
30	25, 13, 15, 44
34	27, 19, 4, 28
Long queries	
15	25, 13, 35, 26
24	21, 66, 31, 7
32	46, 8, 97, 1
44	45, 42, 92, 7

Table 5. The input queries and their four best recommended queries

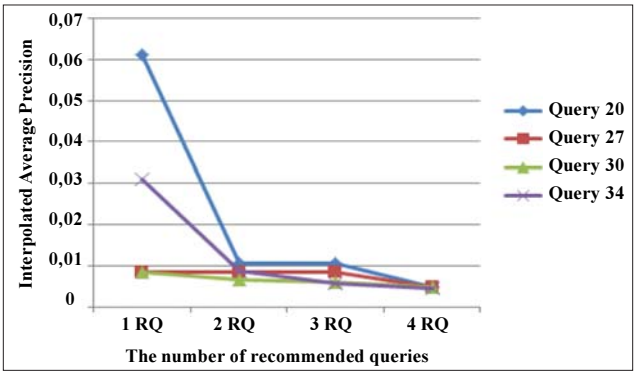

Figure 8. Variation of the number of recommended queries used for the expansion of short queries

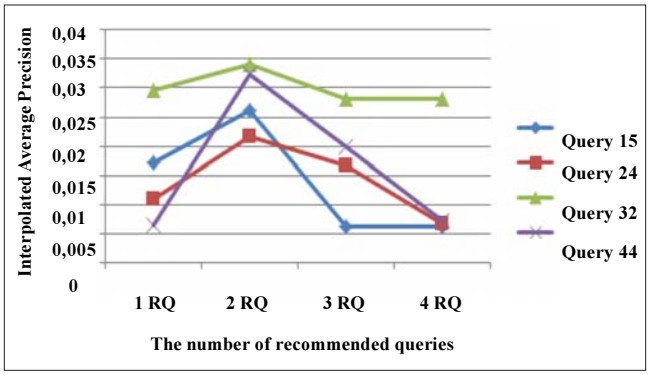

Figure 9. Variation of the number of recommended queries used for the expansion of long queries

The experiments illustrated in Figure 8 show that, in the case of short input queries, the IAP is decreasing while we use more recommended queries and their five top-ranked documents.

In contrast, the Figure 9 shows that for long queries the highest values of IAP are reached by using two recommended queries and their five top-ranked documents in the process of expansion.

From these results, we can assume that the precision of the Information Retrieval System is at its top when we use one recommended query and its five top-ranked documents in the process of expansion of queries that have five or less terms. While, the user queries that contain more than five terms, need to be expanded using the second recommended query and its five top-ranked documents also.

3.5 Query Expansion: Comparison with a Pseudo-Relevance Feedback approach

The final step of experimentations is a comparison between our Probabilistic Query expansion approach and the Rocchio Algorithm which is a Pseudo-Relevance Feedback method. We applied Rocchio also using the relevant documents of the input query, its Recommended Queries (RQ) and the relevant documents of each RQ used.

In this experiment, we expanded the eight queries by adding fifteen terms, we used the five top-ranked documents of each query and we varied the number of recommended queries used in the two processes of expansion (our approach and the Rocchio Algorithm). In the Figure 10 we illustrate the average of the value of IAP of the four short queries, and in and Figure 11 those of the four long queries.

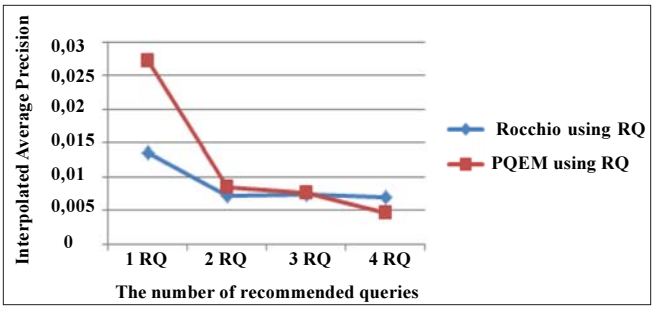

Figure 10. The comparison between the Probabilistic Query Expansion Method (PQEM) and the expansion by Rocchio using short input queries

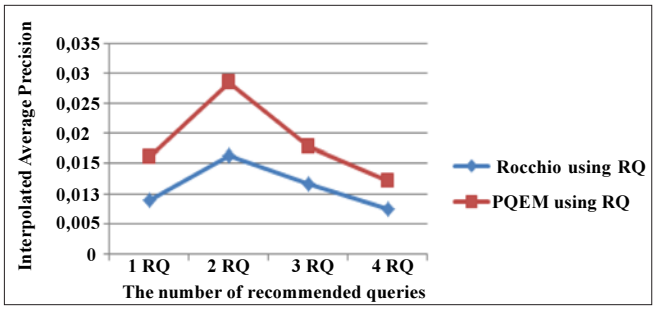

Figure 11. The comparison between the Probabilistic Query Expansion Method (PQEM) and the expansion by Rocchio using long input queries

The Figure 10 shows that our Probabilistic Query Expansion approach gives better precision that the Pseudo-Relevance Feedback method used while using one or two recommended queries. By using the three RQs the value of IAP for our approach is slightly higher and tends to be equal. However, in the case of the use of four RQs the precision of the expanded queries by Rocchio exceeds the precision of the expanded queries by our approach.

In contrast, we notice in the Figure 11 that our approach gives better values of IAP than Rocchio all the time with a significant difference.

4. Conclusion and Future work

In this work we have presented a query recommendation algorithm and a probabilistic query expansion method, based on the relevant documents and the terms contained in the user past queries. We did the experimentations with the four short queries and the four long queries that have the largest number of relevant documents in the database CISI of the Collection SMART. The results show that most of the time the score of a recommended query based on terms is more important than the score based on documents, and that the information retrieval system reach the best precision when we expand the user query with fifteen terms and we use the five top ranked documents of the user query and we add to it the first best recommended query and its five top-ranked documents in the case of short queries. While for long queries, we have to add the first and second recommended queries to the user query and the five top-ranked documents of every added recommended query. The experiments that we did also show, that our Probabilistic Query Expansion Method, gives better precision to the IRS than the Pseudo-Relevance Feedback Method Rocchio.

As future work, we propose to improve the approach developed in this work by adding to it the notion of semantic information retrieval and by triying to use a classification method on the past user queries. Concerning the experimentations, we intend to apply our approach in a data set of past user queries extracted from a real search engine.

References

[1] Wen, J., Nie, J., Zhang, H. (2001). Clustering User Queries of a Search Engine. *In*: Proceedings of WWW10, Hong Kong, May, p. 1-5.
[2] Baziz, M. (2005). Indexation conceptuelle guide par ontologie pour la recherche d'information. PhD thesis, Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier de Toulouse, December 14.
[3] Saint Requier, A., Dupong, G., Adam, S., Lecourtier, Y. (2010). Evaluation d'outils de reformulation interactive de requêtes. *In*: Proceedings of 7th Conférence en Recherche d'information et Applications, CORIA 2010, Sousse, Tunisie.
[4] Wei, J., Bressan, S., Ooi, B. C. (2000). Mining term association rules for automatic global query expansion: methodology and preliminary results. *In*: Proceedings of the First International Conference on Web Information Systems Engineering (WISE'00), 1, 366 - 373.
[5] Xu, J., Croft, B. (1996). Query Expansion Using Local and Global Document Analysis. *In*: Proceedings of SIGIR'96, Zurich, Switzerland.

[6] Zahera, H. M., El Hady, G. F., Abd El-Wahed, W. F. (2010). Query Recommendation for Improving Search Engine Results. *In*: Proceedings of the World Congress on Engineering and Computer Science (WCECS 2010), San Francisco, USA, October, 1, 20-22.
[7] Cui, H., Wen, J., Nie, J., Ma, W. (2002). Probabilistic Query Expansion Using Query Logs. *In*: Proceedings of WWW2002, Honolulu, Hawaii, USA, May, p. 7-11.
[8] Kunpeng, Z., Xiaolong, W., Yuanchao, L. (2009). A new query expansion method based on query logs mining. *International Journal on Asian Language Processing*, 19 (1) 1-12, China.
[9] Baeza-Yates, R., Hurtado, C., Mendoza, M. (2004). Query Recommendation Using Query Logs in Search Engines. LNCS 3268, Springer-Verlag, p. 588-596, Berlin Heidelberg.
[10] El Qadi, A., El Ghali, B., Aboutajdine, D. (2011). *Query Expansion Based Past User Queries*. *In*: Proceedings of the 2th Conference on Extraction et Gestion des Connaissances, EGC 2011, p. 249-258, November 23-25, Ensa-Tanger, Maroc.
[11] Tebri, T. (2004). Formalisation et spécification d'un système de filtrage incrémental d'information. PhD Thesis, Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier de Toulouse, December 15.
[12] Kjersti, A., Eikvil, L. (1999). Text Categorisation: A survey. Technical Report, Norwegian Computing Center, Norway, June.
[13] Slimani, T., Ben Yaghlane, B., Mellouli, K. (2007). Une extension de mesure de similarité entre les concepts d'une ontologie. *In*: Proceedings of SETIT 2007 (4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications), TUNISIA, March 25-29.
[14] Ganguly, D., Leveling, J., Jones, G. J. F. (2011). Query expansion for language modeling using sentence similarities. *In*: The 2nd Information Retrieval Facility (IRF) Conference, 6th June, Vienna, Austria
[15] Bai, J., Nie, J. (2004). Using Language Models for Text Classification, Conference'04, Month 11, Washington D.C., U.S.A.
[16] Chen, S. F., Goodman, J. (1998). An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Science Group*, *Harvard University*, Cambridge, Massachusetts.
[17] Cao, G., Nie, J., Bai, J. (2005). Integrating Word Relationships into Language Models. *In*: Proceedings of SIGIR'05, Salvador, Brazil, August 15-19.
[18] Zhai, C. (2009). Book: Statistical Language Models for Information Retrieval.