QUERY EXPANSION WITH MATRIX CORRELATION TECHNIQUES
A Systematic Approach

Claudio Biancalana, Antonello Lapolla and Alessandro Micarelli
Department of Computer Science and Automation, Artificial Intelligence Laboratory
Roma Tre University, Via Della Vasca Navale, 79, 000146 Rome, Italy

Keywords: Information Retrieval, Query Expansion, Personalization.

Abstract: This paper presents an Information Retrieval system that employs techniques based on Personalization and Query Expansion (QE). The system was developed in an incremental and iterative way, starting from a simpler system and reaching a more complex one, to the point that it is possible to talk about several systems each based on a specific, deeply analyzed approach: four systems sharing the concept of term co-occurrence. Starting from a simple system based on bigrams, we moved onto a system based on term proximity, through an approach known in the literature as Hyperspace Analogue to Language (HAL), and eventually developing a solution based on co-occurrence at page level. The latter presents a hybrid approach based on term proximity. This novel architecture is shown here for the first time to our knowledge.

1 INTRODUCTION

The considerable quantitative increase in the amount of documents on the World Wide Web has led to a scenario in which disorganization gained the upper hand, due to the many different languages composing the documents, typically drafted by a huge number of authors on the Web. This fact leads to the need of supporting the user more efficiently in retrieving information on the web. Users easily find problems in retrieving information by means of a simple Boolean search to check the presence of the searched-for term in the web texts (Jansen et al., 2000). Indeed, some texts, consisting of terms that are often synonyms, or related to similar topics only, do not allow to conduct a proper search, and only take into consideration a few terms, which could be input by a user who is likely to have no or little experience in on-line searches. The Query Expansion (QE) technique fits in this disordered scenario to support the user in his/her search and allow to widen the search domain, to include sets of words that are somehow linked to the frequency of the term the user specified in his/her query (Bai et al., 2005). These may be simple synonyms or terms that are apparently not connected to syntactic meaning, but nevertheless linked to a context that is similar or identical to the one expressed by the original search provided by the user (Burgess et al., 1999). Such information may be obtained in several ways, the main difference being the source used to obtain further information, which can be retrieved through the preferences explicitly indicated by the user, through the user's interaction with the system (Gasparetti and Micarelli, 2007), through the incremental collection of information that links the query terms to document terms (Gao et al., 2004)(for instance the search session logs (Anick, 2003)) or by means of a simple syntactic analysis of the phrase forms that compose the documents (Teevan et al., 2005),(Radlinski and Joachims, 2005).

This paper is organized as follows. Section 2 is an overview of the implemented systems. In section 3 we introduce the general architecture of the developed systems. Section 4 presents our experimental setup and gives a detailed description of the results. Finally, in 5 we illustrate our conclusions.

2 THE SYSTEMS

The systems we present in this work are based on the same approach: the query input by the user into the search engine is expanded through terms linked with

Biancalana C., Lapolla A. and Micarelli A. (2008).
QUERY EXPANSION WITH MATRIX CORRELATION TECHNIQUES - A Systematic Approach.
In *Proceedings of the Fourth International Conference on Web Information Systems and Technologies*, pages 34-41
DOI: 10.5220/0001520000340041

the content of the previously visited web pages, hence pertaining to all the user's information needs. Thus, the QE process goes with a mechanism that builds the user model. As for the QE process, it is an approach based on the automatic expansion of the query, based on implicit feedback, formed by the pages previously visited by the user. Besides, the developed systems are referable to global analysis techniques, since nor preliminary search is done through the original query.

The personalization process and the construction of data structures for the user model are thus incremental, so as to dynamically adapt themselves to the user changing interests. All the developed systems build the user model following the concept of term co-occurrence (Schütze and Pedersen, 1997). By co-occurrence we mean the extent of which two terms tend to appear simultaneously in the same context.

In this research we implemented and rated four systems, each one characterized by two or more different versions. The four systems differ in the way they define the co-occurrence between two terms.

- **System I**: this is the most straightforward system among the implemented ones, both conceptually and computationally. The user model is built around the concept of bigrams, namely a pair consisting of two adjacent terms in the text of a web page. Two terms are considered co-occurring only if adjacent. The context of a term is thus exclusively limited to the term that is directly next to it, either to the left or to the right;
- **System II**: this system is based on the Hyperspace Analogue to Language approach, in which the context of a term is expanded to a window of N adjacent terms. Given a window of N terms, that can be scrolled inside a page text, two terms are considered co-occurring only if they are within such window. The co-occurrence value will be inversely proportional to the distance between the two terms within the window;
- **System III**: within this system, the context of a term is expanded to the entire page considered. Two terms are then deemed co-occurring only if they are both present, simultaneously, in the same page;
- **System IV**: this is a hybrid system, where the co-occurring of two terms is not just the two terms' frequency in the same document, but also the distance between the two terms within the considered text. It is therefore a system which tries to find a compromise between the approaches used in the previous three systems.

3 GENERAL ARCHITECTURE OF THE DEVELOPED SYSTEMS

The four developed systems feature the same architecture to create the user model and to execute the QE. These systems differ in that they adopt different methods to create the co-occurrence matrix. The main architecture elements are the following:

3.1 Creation of the User Model

1. For every training link, the corresponding html page is obtained, and the textual information is taken from it through a parser, the purpose of which is to get rid of html tags;
2. the extracted textual information is analyzed with a part of speech (POS) tagger, MontyLingua [1], which can make a semantic analysis of the text. The text is then broken down into sentences, where nominal and verbal phrases are tracked down; each term is then tagged as adjective, noun, proper noun or preposition;
3. the terms included in the stop word list (a list of words that, owing to their high occurrence within a text document are considered irrelevant for Information Retrieval) are removed from the textual information analyzed by the POS tagger;
4. the textual information previously analyzed by the POS tagger and cleansed of stop words now undergoes stemming, by means of Porter's algorithm (Porter, 1997). The stemming algorithm allows to trace terms with the same root back to the same word;
5. the textual information is analyzed according to the chosen system, taking into consideration or not the extra semantic information gathered by the POS tagger. In all the implemented systems the user model consists of a co-occurrence matrix where, depending on the cases, such co-occurrence is to be seen as a simple bigram for sentences, nominal phrases or documents;

In all the implemented systems a co-occurrence matrix is taken from each page. The lines of these matrices are normalized so as to obtain comparable values for each line. The matrices of the single pages are added incrementally, in order to form one single matrix of co-occurrences for the entire corpus.

[1] http://web.media.mit.edu/hugo/montylingua

3.2 Execution of the Expansion

1. Given query Q, consisting of n terms q_i, $i = 1,\ldots,n$, for each of the q terms, the corresponding stemmed term q'_i is calculated, hence obtaining the new query Q' to be represented in vectors as $\langle q'_1,\ldots,q'_n\rangle$, $i = 1,\ldots,n$;
2. for each of the terms q'_i, $i = 1,\ldots,n$, belonging to query Q', the vector corresponding to $\overrightarrow{cv_{q'_i}} = \langle c_1,\ldots,c_m\rangle$ is taken from the co-occurrence matrix, where m stands for the number of stemmed terms found in the training corpus, a value corresponding to the size of the co-occurrence matrix;
3. given the stemmed term of query q'_i,$i = 1,\ldots,n$, and the corresponding co-occurrence vector $\overrightarrow{cv_{q'_i}} = \langle c_1,\ldots,c_m\rangle$, it is possible to calculate the weighting the latter by means of a further co-occurrence measure, indicated as $c-index$. Given the term $q'j$, $j = 1,\ldots,n$, and the corresponding vector $\overrightarrow{cv_{q'_j}} = \langle c_1,\ldots,c_m\rangle$, each component of the latter c_i,$i = 1,\ldots,m$, is replaced by the value $c_i \times c-index(t_i,q'_j)$, with t_i standing for the term corresponding to the co-occurrence measure c_i. The two terms, a and b give:

$$c-index(a,b) = \frac{n_{ab}}{(n_a + n_b - n_{ab})}$$

   where n_{ab} stands for the number of documents in the training corpus, in which words a and b are both present, while n_a and n_b indicate the number of documents in the training corpus in which word a and word b are present, respectively. The $c-index$ measure referring to two words therefore increases according to the frequency with which the two words appear together in the document rather than alone. Hence given a term of the query and the corresponding co-occurrence vector, the use of the $c-index$ tends to consolidate co-occurrences with words that usually appear together with the query term, and not much alone.
4. the vectors of co-occurrence with the query terms, to be possibly weighted with the $c-index$, are added up to obtain a single vector representing the terms that mostly co-occur with all the query terms;
5. once obtained the vector $\overrightarrow{cv_Q} = \langle c_1,\ldots,c_m\rangle$, referring to query Q, it is possible to weigh each co-occurrence value c_i, $i = 1,\ldots,m$, with the Inverse Document Frequency (IDF) (Salton et al., 1975) of the corresponding t_i term. The IDF of each term was calculated beforehand, considering the entire corpus.
6. the components of vector $\overrightarrow{cv_Q} = \langle c_1,\ldots,c_m\rangle$, referring to Q are ordered according to the decreasing co-occurrence values, removing the terms for which the co-occurrence value is 0.0. Starting from vector $\overrightarrow{cv_Q} = \langle c_1,\ldots,c_m\rangle$, we get the vector of ordered pairs $\overrightarrow{cv\_Pair_Q} = \langle (t_1,c_1),\ldots,(t_s,c_s)\rangle$ where s stands for the number of terms co-occurring with the query terms;
7. the stemmed terms present in the vector of ordered pairs are replaced by the original terms through the stemming table, provided that they are not already present in the starting query Q. The original terms are given the co-occurrence value of the corresponding stemmed term;
8. the expansion query relating to query Q is obtained by taking the first n terms of the vector of ordered pairs $\overrightarrow{cv\_Pari\_ex_Q}$. The original query Q terms are then given a co-occurrence value of 1.0 and added to the first n terms. Each query term is weighted according to its co-occurrence value.
9. the obtained query is then input in the search engine, which searches for the pages that mostly pertain to the query. The use of co-occurrence values in the query allows to assign a greater weight to the words with higher co-occurrence values.

It is very interesting to notice that given two terms t_1 and t_2, their similarity value is directly taken from the co-occurrence matrix, considering the element (t_1,t_2) or (t_2,t_1), given the symmetry of the matrix itself.

What follows is the description of the main development characteristics of the single systems implemented.

3.3 Bigram-based System (System I)

This is the simplest QE system among the implemented ones. It is based on a very simple approach, that of limiting the context of a word to its two adjacent words, to the left and to the right. Each word thus forms two pairs, one with the word to the right, and one with the word to the left. Given a document, the terms contained in it are stemmed, and all the pairs of adjacent words, known as bigrams, are searched for. The pair (a,b) was considered to be equal to the pair (b,a), while pairs such as (a,a), where the two terms are identical, were left out. The final co-occurrence value will be equal to the number of times the two words are adjacent in the document.For each document, a co-occurrence matrix is then built, whose lines are normalized. Finally, all the matrices in the training documents are summed up to obtain the co-occurrence matrix.

3.4 HAL-based System (System II)

Considering the limits of the bigram-based approach, with reference to the small size of a term's context, exclusively associated with the adjacent terms, we decided to expand this context to a window of N terms, using the Hyperspace Analogue to Language approach,(Bruza and Song, 2002). The co-occurrence matrix is generated as follows: once a term is given, its co-occurrence is calculated with the N terms to its right (or to its left). In particular, given a term t and considered the window of N terms to its right (or left) $f_t = \{w_1, \ldots, w_n\}$, we get $co-oc(t, w_i) = \frac{w_i}{i}$, $i = 1, \ldots, N$. During the testing phase, N was given a value of 10. As in the bigram-based approach, pair (a,b) is equal to pair (b,a): hence even in this case the co-occurrence matrix is symmetrical. For each one of the training documents a co-occurrence matrix is generated, whose lines are then normalized. The matrices of the single documents are then summed up, generating one single co-occurrence matrix representing the entire training corpus. The text is broken down into nominal expressions, as before, but instead of gathering all the terms in one single document, the breakdown is maintained intact, in nominal expressions. This is when the *HAL* algorithm is implemented separately on the single nominal expressions of the document. We want to ascertain if and how much the addition of semantic information, such as the breakdown into nominal expressions, can help enhance performance, still implemented the weighting system of co-occurrences based on the joint use of IDF and $c-index$.

3.5 System based on Co-occurrence at Page Level (System III)

The systems implemented so far base the construction of the co-occurrence matrix on the proximity of words: in the case of bigrams, co-occurrence is limited to two adjacent words, while co-occurrence in the HAL-based approach is extended to a window of N terms. Both methods take advantage of the concept of word proximity: the more the two words are closer in the text, the higher the probability they will be semantically linked. In the approach we are about to describe, we have decided to pursue a totally different method in building the co-occurrence matrix, which allows to overcome the limit of considering two co-occurring terms only if they are close to each other in the text. Indeed, we tried to implement a system which exploits co-occurrence at a page level, namely trying to track down the pairs of words that usually co-occur within the same training document, regardless of the distance between them; each term in a document is considered co-occurring with all the other terms in that very document. The number of times the term appears in the document is counted, and the vector $\overrightarrow{o_v} = \langle (t_1, tf_{t_1}), \ldots, (t_n, tf_{t_n}) \rangle$ is generated, where N stands for the number of different stemmed terms within the training document under discussion. Such vector consists of pairs (t_1, tf_{t_i}), $i = 1, \ldots, N$, where t_i stands for a term present in the document, and tf_{t_i} the number of times it appears in the document. The benefits of this weighting mechanism is evident when the QE is done. As seen before, for each query term, the co-occurrence vector is calculated. These vectors are then summed up. The weighting mechanism makes the contribution of the co-occurrence of the hardly relevant terms of the query in the document corpus less important compared to the co-occurrence of relevant terms. Hence, for each training document, a co-occurrence matrix is generated. These matrices are then summed up so as to form one single matrix of co-occurrences, which is used for the QE. Once the textual information is obtained from the training links, the POS tagger extracts the nouns, proper nouns and adjectives. Not all these terms are selected, only the first k are used, following an order based on $tf \times idf$. Co-occurrences at a page level are then calculated exclusively using these first k keywords where k is a fixed parameter of the system, which is the same for any page to be analyzed.

3.6 System based on Co-occurrence at Page Level and Term Proximity (System IV)

System II is exclusively based on the concept of co-occurrence at page level: it attempts to track down the terms that are usually present simultaneously in the same pages, without even considering the distance between the words within the text. Ignoring term proximity within the same document can lead to a considerable loss of information, since two words that are close to each other are more likely to be correlated, from a semantic viewpoint too. That's why we decided to use a hybrid approach, that doesn't use page-level co-occurrence only, but that also considers term proximity, as the bigram-based and HAL-based systems do. Following this idea we implemented and tested a hybrid approach, starting from the extraction of nominal expressions and exclusively considering nouns, proper nouns and adjectives. Moreover, the weighting mechanism based on *IDF* and $c-index$ is used. In order to carry out the QE, the two vectors of co-occurrence with the query terms are obtained separately, following the HAL-based approach and the ap-

proach based on co-occurrence at page level, without extracting the keywords. Such vectors are therefore the same ones obtained from systems I and II respectively. Each vector contains a different type of information: the co-occurrence at page level vector consists of the terms that are usually present in the same documents in which the query terms appear, while the HAL-type co-occurrence vector contains the terms that are usually present in the documents, in proximity of the query terms. The blending of the two types of information is done by introducing a new element which is known as proximity matrix. This matrix is basically an extension of the HAL-based approach. Whilst forming the co-occurrence matrix the HAL-based approach, given a term t, considers t co-occurring with the adjacent N terms, associating a greater co-occurrence value to the terms that are closer to t. The HAL method thus envisages the use of a window of N adjacent terms. We therefore asked ourselves how the use of a preset-size window can entail a loss of information, and we decided to employ a method that allows to consider the proximity of term t with all the other terms in the document. Let us see how the proximity matrix P is built. A matrix P is constructed, having size M, namely the number of stemmed terms present in the training corpus. Each matrix box contains two values, v_1 and v_2, which we initialize at 0.0 and 0, respectively. For each document d of the training documents corpus and for each term t present in d, all terms to the right of t are considered, and distance i from t is measured. Assume that term t' is at distance i from t, we extract from matrix P the pair of values (v_1, v_2) in box (t, t'), and we increase v_1 by $\frac{1.0}{i}$ and v_2 by 1. As for the construction of the query, it is done following the same method adopted in system III. The only difference is to be seen in the proximity measuring when co-occurrence at page level values of the query terms are extracted. Given term q' belonging to query Q, and having extracted the corresponding co-occurrence at page level vector $\overrightarrow{cv_q}$, the co-occurrence value of each term t' belonging to vector $\overrightarrow{cv_q}$ is multiplied by $\frac{v1}{v2}$, where v_1 and v_2 are the values present in the proximity matrix corresponding to the pair of terms (q', t') based on co-occurrence at page level and term proximity, through which it is possible to understand the assets and weak points of each term, also with reference to the systems based on different approaches.

3.7 Pseudo-coding of the Seminal Algorithms System III and System IV

In this subsection we show the pseudo-coding of the algorithm which calculates the co-occurrence matrix starting from the set of training documents (see Algorithm 1), and the pseudo-coding of the algorithm for the actual execution of the QE, starting from the data contained in this matrix (see Algorithm 2). With reference to the algorithm calculating co-occurrence at page level with extraction of the first k keywords, we notice that:

- the co-occurrence matrix is represented by a map of maps. In this way, we avoid initializing a square matrix the size of which is the overall number of different terms in the training documents set. Using this matrix would entail a huge waste of memory, since the majority of its elements have a value equal to zero, considering how sparse the matrix is. By using a map of maps on the other hand, it is possible to input the co-occurrence values between the pairs of co-occurring terms in the training documents when such pairs are present;
- the `keys()` method of a map yields the list of keys;
- the `items()` method of a map yields the list of pairs `(key, value)`;
- the $IDF(t)$ method yields the inverse document frequency of term t, calculated previously according to the documents belonging to the third level of the DMOZ directory (see section 4.1).

As for the algorithm calculating the QE through the matrix of co-occurrence at page level, we notice that:

- the sum method `(coocMap1, coocMap2)` yields a new map, given from the union of pairs `(key, value)` present in the two maps, `coocMap1` and `coocMap2`; should a key be present in both maps, the corresponding value is given by the sum of values contained in the starting maps;
- the stemming method (Q) yields a new query, obtained by stemming the terms of query Q;
- the expand method (t) yields the list of non-stemmed terms, found in the training documents, whose root is equal to term t.

Algorithm 1: Pseudo-coding algorithm of the co-occurrence-based system at page level (System III).

```
begin
    Δ co-occurrence global matrix
    initialization, represented by a map of maps
    M ← Map([])
    Δ training documents analysis
    for doc in D do
        Δ term occurrence map initialization
        contained in a single document
        tf ← Map([])
        Δ term frequency calculation
        for t in doc do
            if not t in tf.keys() then
                tf[t] = 0
            else
                tf[t] = tf[t] + 1
        Δ tf * idf calculation for every terms in
        the document
        for t in tf.keys() do
            tf[t] = tf[t] × idf(t)
        Δ get a list ordered by tf * idf of k
        couples (t, tf * idf_t)
        tfidf_list = map_to_ord_list(tf)[0 : K]
        Δ normalize tf * idf values
        tfidf_list = normalize(tfidf_list)
        Δ transform the list into map
        tfidf_map = Map(tfidf_list)
        Δ get unique document-terms list
        term_list = tfidf_map.keys()
        Δ update global co-occurence matrix
        for t1, tfidf in tfidf_map.items() do
            if not t1 in M.keys() then
                M[t1] ← Map([])
            for t2 in term_list do
                if not t2 in M[t1].keys() then
                    M[t1][t2] = 0.0
                else
                    M[t1][t2] = M[t1][t2] +
                    tfidf × tfidf_map[t2]
end
```

Algorithm 2: Pseudo-coding algorithm of the co-occurrence-based system at page level and term proximity (System IV).

```
begin
    Δ co-occurrence global matrix
    initialization, represented by a map of maps
    Q ← [q1, q2, ..., qn]
    Δ co-occurrence map initialization
    cooc_map ← Map([])
    Δ stemming
    Q ← stemming(Q)
    Δ co-occurrence map update for every
    terms in the query
    for q in Q do
        Δ update co-occurrence map A
        cooc_map = sum(cooc_map, M[q])
    Δ get an ordered list of couples
    (t, cooc_val), sorted by co-occurrence
    values
    cooc_list = map_to_ord_list(cooc_map)
    Δ transform the list into map
    cooc_map = Map(cooc_list)
    Δ query initialization
    exp_query = []
    Δ query expansion
    for term, cooc_val in cooc_map.items() do
        Δ get a non-stemmed term list
        associated to term
        exp_list = expand(term)
        Δ expand terms
        for orig_term in exp_list do
            exp_query.append((orig_term, cooc_val))
    Δ limit query expansion to first k
    co-occurrence terms
    exp_query = exp_query[0 : K]
    Δ merge original terms
    for q in Q do
        exp_query.append((q, 1.0))
    return exp_query
end
```

4 EXPERIMENTATION

For each of the adopted approaches, the results of the corresponding systems are presented. The comparison between different systems is made by using comparative performance values obtained from the system under examination, on one single topic or the entire benchmark. Such performances are expressed in F1-measures, so as to summarize, in one single measure, precision and recall values. As for the performance measures taken into consideration, we have precision, recall and f1-measure:

$$precision(t) = \frac{n_t}{50} \qquad recall(t) = \frac{n_t}{N_t}$$

$$F1 - measure = \frac{2 \times precision \times recall}{precision + recall}$$

where n_t stands for the number of returned links belonging to topic t, and N_t the overall number of test links belonging to topic t present in the index.

4.1 The Employed Benchmark: The Open Directory Project

The Open Directory Project (ODP), also known as DMOZ[2], is a multilanguage directory of links belonging to the World Wide Web, namely a system to collect and classify links. The Open Directory Project has a hierarchic structure: the links are grouped into categories, also known as topics, and subcategories. It is therefore possible to identify a level-based organization within the hierarchy. Given the large quantity of links contained in ODP, we decided to consider only Level III links. The pages corresponding to such links were downloaded from the World Wide Web, by using a parser; the textual information was taken from it, and then it was indexed by means of the Lucene indexing system [3]. Ten topics were then chosen from the Level III topics, five of which corresponding to the user's information needs, and five whose function was exclusively to generate noise in the creation of the user model. Each topic's links were then subdivided in a training set, corresponding to 25% of the links, and set of tests, corresponding to 75% of the links (see table 1).

4.2 Experimentation Methods

Once the user model is generated, it is possible to carry out real tests as follows. A query is built for each topic belonging to the user's information needs. The terms of the query are simply the terms that form the topic name. This query is then expanded according to the user model, and used to search for web pages within the created index, starting from all third-level links. The pages belonging to the training set of the considered topic are removed from the returned pages; only the first fifty are taken into consideration, which include the number of pages belonging to the topic under consideration. The index obtained with Lucene, starting from the third-level link of ODP, consists of 131,394 links belonging to 5,888 topics.

Table 2 compares the performances of the four systems. It is possible to notice that the system based on co-occurrence at page level (system III) clearly achieves better results compared with other systems based on term proximity (system IV), such as bigrams (system I) and HAL (system II). Indeed, both

[2]http://dmoz.org

[3]http://lucene.apache.org

Table 1: The employed benchmark: statistics.

Topic	Test links	Training links	Information Needs
S/C/H_P_V	15	5	yes
C/H_A/P_and_M	27	7	yes
B/M_and_D/C	74	18	yes
G/R/D_and_P	52	14	yes
B/A_and_F/F	100	27	yes
S/C/P	35	7	no
A/P_A/M	25	6	no
S/P/M_and_E	26	7	no
S/S_S/L	13	5	no
R/G/R	15	5	no
Tot.	382	101	

Table 2: Comparative F1-Measurement on implemented Systems.

Topic	System I	System II	System III	System IV
C/H_A/P_and_M	0.00	0.00	0.16	0.16
S/C/H_P_V	0.13	0.06	0.09	0.06
G/R/D_and_P	0.14	0.12	0.18	0.16
B/M_and_D/C	0.21	0.27	0.19	0.21
B/A_and_F/F	0.34	0.36	0.57	0.48
Average	**F1**	**F1**	**F1**	**F1**
	0.16	0.16	**0.24**	0.21

Table 3: Comparative F1-Measurement.

Topic	no QE	RF	System III
C/H_A/P_and_M	0.05	0.08	0.16
S/C/H_P_V	0.09	0.13	0.09
G/R/D_and_P	0.10	0.18	0.18
B/M_and_D/C	0.19	0.14	0.19
B/A_and_F/F	0.05	0.14	0.57
Average	**F1**	**F1**	**F1**
	0.10	0.13	**0.24**

are based on the concept of term proximity, and more specifically they imply a correlation between two terms when they are close to each other in the text. This approach, however, entails the loss of information linked to terms that usually co-occur in the documents themselves, but which are not always close to each other in the text. The approach based on co-occurrence at page level hence steers away from the term proximity concept, and tries to track down the terms that are usually simultaneously present in the same documents. The experimentation results show that page-level correlations are stronger than those based exclusively on term proximity within the text: given a page relating to a particular topic, this will feature correlated terms, not because of their proximity, but because they refer to the same topic. We also notice that the system based on keyword extraction is the one that offered the best performance among the pre-

sented ones: to calculate the correlation values, this system takes into consideration only the most relevant terms in the text, thus preventing a large quantity of noise from impairing the performance.

Table 3 shows the results obtained by a system based on a traditional content-based user-modeling approach, where documents are represented in the Vector Space Model (VSM) and without Query Expansion, in comparative terms. This system particularly focuses on the update of the user model by means of Relevance Feedback (RF) techniques (Salton and Buckley, 1997), applied to the training pages content: for each category, the first ten keywords are taken from the corresponding training pages. The keywords are obtained in terms of $tf \times idf$, and are then used to expand the query.

5 CONCLUSIONS

In this research we implemented and analyzed an Information Retrieval system, based on QE and Personalization, that may help the user search for information on the Web, with reference to his/her information needs. The four systems implemented are based on the following approaches: bigrams, Hyperspace Analogue to Language (HAL), co-occurrence at page level and co-occurrence at page level with term proximity. Among the developed systems, the one based on co-occurrence at page level and keyword extraction stood out. Indeed, this system, based on an algorithm calculating co-occurrences, obtained the best results, in terms of performance, on the reference benchmark.

REFERENCES

Anick, P. (2003). Using terminological feedback for web search refinement: a log-based study. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 88–95, New York, NY, USA. ACM Press.

Bai, J., Song, D., Bruza, P., Nie, J.-Y., and Cao, G. (2005). Query expansion using term relationships in language models for information retrieval. In *CIKM*, pages 688–695.

Bruza, P. D. and Song, D. (2002). Inferring query models by computing information flow. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 260–269, New York, NY, USA. ACM Press.

Burgess, C., Livesay, K., and Lund, K. (1999). Exploration in Context Space: Words, Sentences, Discourse. *Discourse Processes*, 25(2&3):211–257.

Gao, J., Nie, J.-Y., Wu, G., and Cao, G. (2004). Dependence language model for information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177, New York, NY, USA. ACM Press.

Gasparetti, F. and Micarelli, A. (2007). Personalized search based on a memory retrieval theory. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI): Special Issue on Personalization Techniques for Recommender Systems and Intelligent User Interfaces*, 21(2):207–224.

Jansen, B. J., Spink, A., and Saracevic, T. (2000). Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, 36(2):207–227.

Porter, M. F. (1997). An algorithm for suffix stripping. pages 313–316.

Radlinski, F. and Joachims, T. (2005). Query chains: Learning to rank from implicit feedback.

Salton, G. and Buckley, C. (1997). Improving retrieval performance by relevance feedback. pages 355–364.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Schütze, H. and Pedersen, J. O. (1997). A cooccurrence-based thesaurus and two applications to information retrieval. *Inf. Process. Manage.*, 33(3):307–318.

Teevan, J., Dumais, S. T., and Horvitz, E. (2005). Personalizing search via automated analysis of interests and activities. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456, New York, NY, USA. ACM Press.