A CONTENT-BASED APPROACH TO RELEVANCE FEEDBACK IN XML-IR FOR CONTENT AND STRUCTURE QUERIES

Luis M. de Campos, Juan M. Fernández-Luna, Juan F. Huete and Carlos Martín-Dancausa
Departamento de Ciencias de la Computación e Inteligencia Artificial
E.T.S.I. Informática y de Telecomunicación, CITIC-UGR, Universidad de Granada, 18071 Granada, Spain

Keywords: Relevance Feedback, Information Retrieval, XML.

Abstract: The use of structured documents following XML representation allows us to create content and structure (CAS) queries which are more specific for the user's needs. In this paper we are going to study how to enrich this kind of queries with the user feedback in order to get results closer to their needs. More formally, we are considering how to perform Relevance Feedback (RF) for CAS queries in XML Information Retrieval. Our approach maintains the same structural constraints but expands the content of the queries by adding new keywords to the original CAS query. These new terms are selected by considering their presence/absence in the jugded units. This RF method is integrated in a XML-based search engine and evaluated with the INEX 2006 and INEX 2007 collections.

1 INTRODUCTION

Relevance Feedback (RF) is a very well-known iterative technique to enhance the retrieval performance of a search engine given a query formulated by a user (Ruthven and Lalmas, 2003). When a user formulates a query, the Information Retrieval System (IRS) gives as a result a ranking of documents sorted decreasingly by their relevance degree. Then the user inspects this list and makes assessments with respect to the relevance or non-relevance of the given documents considering her underlying information need. Then the IRS is able to formulate a new query automatically starting from the original query, integrating information from the content of these assessed texts. This new query, as it is based on the user's direct opinion, is supposed to be closer to the user's requirements and helps to enhance the performance of the system. Typically, this process is iterative and could be stopped when the user is satisfied with the material returned by the system in the different feedback steps.

XML-IR (Lalmas, 2009) is the IR field dealing with structured documents, i.e. those which are internally organized in a well defined structure. In an XML-IR search engine the internal structure of the documents in the collection is exploited to give a focused access, i.e. the system could return any type of element that it considers closer to the query (a paragraph, a section or even the whole document).

Also, the internal structure can be exploited when formulating the queries. In this case, the user is able to express which document parts she is interested in, and where the XML IRS has to search in order to find these relevant units. Broadly speaking, the user may represent her information needs in two different ways: by means of the classic keyword-based queries composed of a sequence of keywords, known as *Content Only queries (CO queries)*, or using *Content And Structure queries (CAS queries)*, which also take into account this internal organization giving structural hints about what XML elements to retrieve and where to look for.

In order to show the differences between CO and CAS queries we can consider the following queries:

- CO Query: 'I would like to read something about *the global warming in Asia.*' The output of the system will be the list of the *best* structural elements (which might include some sections, subsections, paragraphs or whole documents) satisfying this query.
- CAS Query: 'I would like to read *sections about Asia in articles dealing with global warming*'. The output of the system will be the list of sections (and only sections) matching this query.

Taking up again the RF subject, but now in the context of XML-IR, the user is presented a list of document elements sorted by their relevance degree. Then she makes the corresponding decisions about the relevance of some of them. The XML IRS may consider,

de Campos L., Fernández-Luna J., Huete J. and Martín-Dancausa C..
A CONTENT-BASED APPROACH TO RELEVANCE FEEDBACK IN XML-IR FOR CONTENT AND STRUCTURE QUERIES.
DOI: 10.5220/0003080104180427
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 418-427
ISBN: 978-989-8425-28-7

not only the content information of the judged elements, but also the type of element, in order to formulate a new query. Therefore, we could say that there are three main types of RF, according to the type of the original and resulting queries:

T1: *CO – CO*: The original query and the expanded query are only composed of keywords. New terms are usually added to the original query and term weights (re)computed.

T2: *CO – CAS*: The original query is only composed of keywords but the expanded query generated by the RF method contains (new) terms, and their corresponding weights, complemented with structural restrictions extracted from the analysis of the assessments.

T3: *CAS – CAS*: Both queries are composed of terms and structural restrictions. In this case, there are two possibilities:

T3.1: Not to modify the structural restrictions expressed in the original query but only adding new terms to the underlying CO queries.

T3.2: In addition to new terms, to modify the existing strutural restrictions or including new ones.

Although, both *CO–XX* and *CAS – CAS* RF have the same inputs, i.e. the triplet (Q,E,J) being Q the original query, E a list of documents' elements and J a set of relevance judgments given to E, there exist some particularities that make the CAS-based RF a different problem: Firstly, the way in which a system must deal with CAS queries differs and secondly the relevant judgments will have a different interpretation. Thus, whereas the relevant judgments are well defined for CO queries[1], this situation does not necessarily hold for CAS queries. For example, in the previous CAS query example, the user only judges a list of sections (the output of the system). Thus, when he/she considers a section as relevant we can assume that it satisfies all the user's requirements (it is about Asia in a article dealing with global warming). But, when the user judges a section as non-relevant it could be because only one of the requirements are not fulfilled (it might talk about Asia but the article does not deal with global warming) or because both of them are not relevant to the query.

RF of T1 and T2 types has been studied in the literature (see Section 3.4), but there exists few papers dealing with T3 type. In this paper we will present a content-oriented approach to solve the CAS–CAS RF problem, particularly the T3.1 case. In this approach we will study how the original query can be expanded, i.e. including new terms (expansion terms) in Q, in order to better capture the user information needs. Some problems will be considered as how many terms and in which parts of the query (structural restrictions) they might be included. Therefore, the main contribution of this paper is a content-oriented CAS–CAS RF method, where terms are selected for expansion not only from the assessed target elements but also for the context elements, which are not explicitly judged to the users.

In order to describe the RF technique, this paper is organized as follows: Next section introduces the NEXI query language and how it was modified to allow the inclusion of new terms in the RF process. Then, Section 3 presents our approach to RF when using CAS queries and also the related works. In order to evaluate the feasibility of our approach, Section 4 is devoted to the experimentation. Finally, Section 5 presents the concluding remarks and points out some future research tasks.

2 PRELIMINARIES

In order to give the corresponding background to understand the method presented in this paper, we shall briefly introduce the NEXI language to formulate CAS queries, as this is the starting point of our CAS – CAS RF method.

To allow queries combining content and structure (Content And Structure Queries, CAS queries) to be specified, the NEXI language (Trotman et al., 2005) was designed. It is a simplified XPath containing only the descendant operator ($//$) in a tag path and also an extended XPath containing the *about* function.

The kind of structured CAS query considered by NEXI can take two possible forms:

- $//C[D]$: Returns C units that fulfill the condition D.
- $//A[B]//C[D]$: Returns C descendants of A where A fulfills the condition B and C fulfills the condition D.

A and C are *paths* (sequences of elements or structural units), specifying structural restrictions, whereas B and D are *filters*, which specify content restrictions, and $//$ is the descendant operator ($A//C$, where C is a descendant, though not necessarily a direct one of A). C is the *target* path (the last structural unit in C is the one that we want to retrieve) and path A is the *context*. Each content restriction (as B or D) will include one or several *about* clauses, connected by either *and* or *or* operators; each *about* clause contains a text (a sequence of words or terms) together with a relative

[1]The user assesses whether the content of *this* element E is relevant (or not) to Q.

path, from the structural unit which is the container of the clause to the structural unit contained in it where this text should be located. The *about* clause is the IR counterpart of the classical *contains* clause used in XPath (which requires an exact matching between the textual content of the clause and a part of the text in the structural element being evaluated). However, *about* does not demand such a strict matching but states, vaguely, that a relevant element should satisfy the information need expressed by means of the text contained in the clause. The wildcard symbol '*' matches any tag and can be used to formulate keyword queries in NEXI. Note that when using '*' we are relegating to the search engine the decision about which structural elements might be retrieved. Examples of NEXI queries are:

- Find sections about *Asia* in articles dealing with *global warming*:

  $Q1$="`//article[about(.,global warming)//section[about(., Asia)]`"

- Find any structural unit that contains a paragraph dealing with *NEXI* in articles about *information retrieval*:

  $Q2$="`//article[about(.,information retrieval)//*[about(//p, NEXI)]`"

- Retrieve bibliography units containing $text3$, which must be in chapter units with a $text1$-related title and contain a section about $text2$:

  $Q3$ = "`//chapter[about(.//title, text1) and about(.//section, text2)]//bibliography[about(., text3)]`"

Measuring the Importance of the Terms in XML-RF After assessing the relevance status of the retrieved elements, a content-based XML-RF will select those terms used to expand the query. In order to select these terms it is necessary to have a measure of the relevance of each term to the query. For example, whenever a term t_i indexes an element judged relevant by the user, perhaps we should increase the belief supporting the relevance of t_i; similarly, if a term t_i only appears in units which are not relevant, we should decrease its relevance belief. Therefore, it seems natural that the relevance of a term will depend on its distribution in relevant and non-relevant units.

In order to include this information in the new query we propose to extend NEXI language with the capability of managing weighted terms: For each term in any about clause we include a weight in the interval $[0,1]$, expressing the importance of the term in the query, being 1 very important and 0 not important at all[2] (in Section 3.3 we will discuss how these weights are computed). For instance, the previous query including term weights could be the following

`//article[about(.,0.9*information 1.0*retrieval)//*[about(//p,0.9*NEXI)].`

3 THE CONTENT-ORIENTED RF METHOD FOR CAS QUERIES

In this section we are going to present our CAS–CAS RF approach. In general terms, our proposal focuses on creating a new expanded CAS query keeping the same structural restrictions as the original but expanding the different content sub-queries of the context and target parts (the content queries included in each about clause of the NEXI query). This is one of the main diferences between our proposal and those in the literature (see Section 3.4).

3.1 Managing CAS Queries in XML Systems

With the aim of offering a background to well understand the approach presented in this paper, we have to briefly outline a typical scheme to solve CAS queries in XML systems. Let us imagine that the previous $Q3$ query has been submitted to the system (query of type $//A[B]//C[D]$). In this query, the structural restrictions are represented by means of the target elements, which are the bibliography units and the context which are chapter units.

The basic idea is to decompose the original CAS query in several simple CAS sub-queries, i.e. a CAS query having only one structural restriction on the target element. The content of each query corresponds to the content part of the related about clauses. In our example, the sub-queries are:

Q_31 = `//chapter[about(.//title,text1)]`

Q_32 = `//chapter[about(.//section,text2)]`

Q_33 =`//chapter[about(.//bibliography, text3)]`

These sub-queries are submitted to the IRS independently and, for each one, a list of relevant elements satisfying the structural restrictions is retrieved, ordered by the Relevance Status Value (RSV). In the example, the system would retrieve //chapter//title, //chapter//section and //chapter//bibliography, for each respective sub-query.

[2]Note that a weight equal to zero does not mean that the structural units that contain the term have to be excluded.

The next step is to compute an aggregated RSV for structural units of the same type for the context and the target ($RSV((//chapter)_l, B)$ and $RSV((//chapter//bibliography)_l, D)$, respectively[3]. Finally, it is necessary to combine these units and their relevance values to obtain the final relevance degree of each bibliography unit in relation to the complete structured query, $RSV((//chapter//bibliography)_l|Q)$.

3.2 RF and CAS Queries

Once the whole CAS query is submitted to the IRS the user inspects the ranking and makes the corresponding relevance assessments (explicitly or implicitly). In our approach, the system will analyze the judged elements and extracts new terms which will be used to expand the query considering both, target and the context, structural restrictions.

In order to do that, the system must be able to refine the existing simple sub-queries independently on their location in the CAS query. When expanding the target sub-queries there is no problem because we know the relevance values of related elements, but this situation does not hold for the rest of the structural restrictions, i.e. context restrictions. This is because the units that might be considered to extract the terms for these sub-queries are usually different to the type of the target unit, and therefore the user did not directly assess relevance judgments on these units.

The problem is how to find those appropriate structural elements from where to select the expansion terms. In order to tackle this problem, in this paper we are going to consider the following assumptions:

- **Relevance Assumption:** *If a user judges a target element as relevant, then all the context elements used to generate the final RSV are considered also relevant.*
- **Non-Relevance Assumption (Hard):** *If a user judges a target element as non-relevant, then all the context elements used to generate the final RSV are considered also non-relevant.*

This last assumption can be considered very restrictive, so we propose also to consider a light version.

- **Non-Relevance Assumption (Soft):** *If a user judges a target element as non-relevant, then we can not infer any relevance value about the context units used.*

[3]With the subindex l we mean any of the units compatible with the context and targets XPaths.

By means of these assumptions we can infer the relevance value for other different units that the ones the user inspects. Therefore, to perform term expansion in CAS – CAS RF we have to keep track of the units used to compute the relevance degree for each element presented to the user. These units will be used to select the most useful terms to expand the content sub-queries, and to compute their weights representing their importance.

In order to illustrate this process we will consider the following example where the target is any section included in an article:

> Find those sections having a paragraph about *NEXI* in a article dealing with *information retrieval*.

```
//article[about(.,information
retrieval)//section[about(//p, NEXI)]
```

Left hand side of Figure 1 shows three results presented to the user and the assessed relevance judgments. According to the method sketched in Section 3.1 to solve CAS queries, these results have to be obtained by the combination of the different units retrieved after submitting to the system the set of simple subqueries. Thus, if we denote by $R(Q,k)$ the k^{th}-result for query Q, we have that

$$R(Q,k) = \otimes_{i=1}^{m} R_k(Q_i),$$

being $Q_1, \ldots, Q_m$ the set of simple subqueries and $R_k(Q_i)$ each one of the intermediate results. In our example, we have two simple CAS subqueries $Q1$ = `//article[about(., information retrieval)]` and $Q2$ = `//article//section[about(//p, NEXI)]`. The first one returns a set of articles and the second one a set of paragraphs in a section.

Let us focus on $R_k(Q_i)$ since they will have an special role in our CAS–CAS RF approach. In this case, it is also possible that there exist several units in the subquery playing a role in the creation of this result, and as consequence they are candidate units from where we might extract the expansion terms. We named these units the *set of generators,* $\mathcal{G}_k$, i.e. relevant units in a subquery satisfying at the same time the structural restrictions. Let us follow with our example by focusing on $R_1(Q2)$, i.e. the results of $Q2$ involved in the first judged result of Q. In this case it is possible to have several paragraphs including the term *NEXI* in the same section. Particularly, looking at right hand side of Figure 1 (the format is: RSV * unit XPath * XML file to which it belongs and the sub-query that they satisfy) we found that the paragraphs `p[2]` and `p[4]` in `/article[1]/chapter[3]/section[4]` satisfy

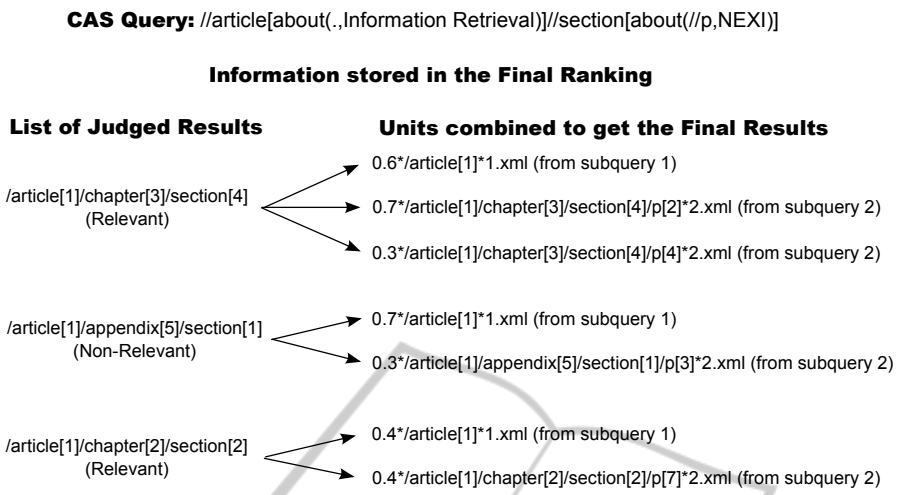

Figure 1: Assessed elements with their corresponding set of generators.

this property, and they are the set of generators for $R_1(Q2)$.

In order to speed up the query expansion process, for each set of generators only one unit (the one having the highest RSV) is considered as a candidate to extract the expansion terms. Thus,

$$R_k(Q_i) = \arg_{u \in \mathcal{G}_k} \max RSV(u, Q_i).$$

Then, the union of these elements for all the n judged results are the units satisfying the structural restrictions associated to each sub-query, named the *set of sub-query expansion units* (SEU):

$$SEU(Q_i) = \cup_{r=1}^{n} R(Q_i, r)$$

In our example we have the following set of subquery expansion units:

- $SEU(Q_1)$: `/article[1]`
- $SEU(Q_2)$:
  `/article[1]/chapter[3]/section[4]/p[2]`
  `/article[1]/appendix[5]/section[1]/p[3]`
  `/article[1]/chapter[2]/section[2]/p[7]`

3.3 Selecting the Expansion Terms

Each subquery Q_i will be expanded by selecting the terms from $SEU(Q_i)$. This is one of the differences from the diferent approaches for CO–CO RF where the terms were selected from those judged units. In this process there are two different, but related, problems: Which are the expansion terms and what are the weights associated to them.

In order to deal with the first problem in this paper we are going to assume that only those terms belonging to relevant elements in $SEU(Q_i)$ but not belonging to any non relevant elements in $SEU(Q_i)$ can be considered as candidates for the expansion of the subquery Q_i. In other words, a term in an non relevant element can not be an expansion term. This is because we are assuming that the content of these elements, which have a great literal similarity with the query, is not related to the query intention.

Therefore, by taking up again the assumptions in Section 3.2 the candidate terms for a subquery Q_i will be determined. Nevertheless, depending on the type of the *non-relevance assumption* (soft or hard) considered, the set of candidate terms might differ. In our example, focusing on the subquery Q_2, a term t_i that belongs to both, `/article[1]/chapter[3]/section[4]/p[2]` and `/article[1]/appendix[5]/section[1]/p[3]` is a valid candidate if we use the soft non-relevance assumption, whereas this term can not be a valid candidate when considering the hard version.

Once we know the candidate terms, the second step will be to select the best ones. In order to deal with this problem we will use a weight measuring the importance of each term. Then, those k candidate terms with the highest importance weights expand the corresponding subquery. Particularly, for a given subquery Q_i, we assume that the importance of a candidate term t depends on the number of relevant elements in $SEU(Q_i)$ in which it appears. Then it is computed according to the following expression:

$$w(t) = \frac{n_{tr}}{n_r}, \tag{1}$$

where n_{tr} denotes the number of relevant elements of the set $SEU(Q_i)$ that contain t, and n_r denotes the total number of relevant elements of that set.

Finally, the expanded CAS query can be formed by adding the k top-weighted terms to the original

query. In this new query the original query terms are weighted with 1.0 and the expanded terms with their corresponding importance weights.

3.4 Related Work

RF has been the objective of many researchers as a mean of improving retrieval effectiveness in XML-IR. Firstly, we can start studying the Rocchio algorithm (Rocchio, 1971) in which most of the works undertaken in content RF in structured IR are based on. It consists of extracting the expressive terms from the elements judged relevant by the user. Each element is seen as a retrieved unit, and elements are therefore considered to be independent from each others, even though they appear in the same document. Ruthven and Lalmas (Ruthven and Lalmas, 2003) have studied different RF techniques (automatic and interactive techniques), specific interfaces to RF systems and characteristics of searches that can affect the use and access of RF systems.

After that, we shall describe some works concentrated on query expansion based on the content of elements with known relevance: The works of Y. Mass (Mass and Mandelbrod, 2004) and C. Crouch (Crouch et al., 2005) are very similar, the only difference between them is in the way of evaluating the queries (authors use respectively distinct indexes with the vector space model or a relevance propagation method). In (Sigurbjörnsson et al., 2004), the authors investigate the effectiveness of blind ("pseudo") feedback and Pan et al. (Pan et al., 2004) apply user feedback to recompute similarities in the ontology used for query evaluation.

On the other hand, several papers have considered structural query expansion which is the objective of this paper: The first work is developed by Mihajlovic et al. (Mihajlovic et al., 2004) to extend their database approach. They assume that knowledge of component relevance provides "implicit structural hints" which may be used to improve performance. Their implementation is based first on "extracting the structural relevance" of the top-ranked elements and then restructuring the query and tuning the system based on RF information. They argue that the document names of the relevant components are used to model structural relevance because these documents are apt to contain similar information. Using the structural information and assessments associated with the relevant elements, the query is rewritten and evaluated.

The second approach, proposed by Schenkel and Theobald (Schenkel and Theobald, 2006), consists of extracting classes of features for each relevant element. These classes of features are: the ancestor class, the descendant class and the content class. For example, for a given relevant element section, article and body elements are added to the ancestor class, paragraph and subsection elements are added to the descendant class, and terms of the section element are added to the content class.

A third approach has been proposed in (de Campos et al., 2009). The main contribution of this approach is that the system automatically selects the best part of the document to be retrieved. Particularly, they consider the expected utility of retrieving each element or structural unit of the documents in a XML collection. Then, after analyzes the retrieved elements the expansion terms are obtained and a new query was submitted to the system.

Fourati et al. (Fourati et al., 2009) propose an approach where they take the original CAS query and the fragments judged as relevant by the user. Then, they create a representation of the original query and relevant fragments under a matrix form. After some processing and calculations on the obtained matrix and after some analysis they have been able to identify the most relevant nodes and their relationships that connect them. These are used to modify the structure of the new query but keeping the same content.

Finally, L. Hlaoua et al. (Hlaoua et al., 2006) propose to add structural constraints to the initial keyword query. Their approach first seeks to identify the generic structure shared by the largest number of relevant elements and then they use this information to incorporate in the expanded query.

4 EXPERIMENTATION

In this section, we shall describe the framework of the experimentation in order to validate our RF approach, as well as discuss the corresponding results. First, we have to highlight the fact that this CAS – CAS RF approach could be used to expand CAS queries for any XML IRS where terms could be weighted.

In general, a CAS query retrieval process can be implemented by means of several modules which are external to the base XML IRS in order to work with NEXI queries but in our case, and in order to evaluate its performance, it has been used on the top of the *Garnata IRS* (de Campos et al., 2006), an Information Retrieval System, designed to work with structured documents in XML.

Table 2: Results of the experiments for context expansion.

	Context Exp (Hard).				Context Exp (Soft).			
k	iP[0.01]	iP[0.05]	iP[0.10]	AiP	iP[0.01]	iP[0.05]	iP[0.10]	AiP
Without RF	0.3460	0.2589	0.2214	0.0831	0.3460	0.2589	0.2214	0.0831
1	**0.3539**	**0.2729**	**0.2427**	**0.0848**	**0,3323**-	0,2624-	0,2213-	0,0833-
2	0.3144	0.2652	0.2346	0.0822	0,3148+	0,2754+†	0,2326-	0,0842+
3	0.2998	0.2499	0.2292	0.0799	0,3044+	0,2709+	0,2376+	0,0848+
4	0.2990	0.2555	0.2319	0.0806	0,2907-	0,2667+	0,2373+	0,0838+
5	0.3131	0.2650	0.2382	0.0816	0,2939-	0,2717+	0,2444+†	0,0841+
6	0.3078	0.2639	0.2366	0.0802	0,3010-	0,2768+†	**0,2526**+†	**0,0849**+†
7	0.3147	0.2634	0.2338	0.0802	0,2956-	0,2719+	0,2484+†	0,0848+
8	0.3064	0.2642	0.2337	0.0794	0,3047-	**0,2793**+†	0,2369+	0,0826+
9	0.3143	0.2645	0.2315	0.0790	0,2920-	0,2666+	0,2326+	0,0832+
10	0.3167	0.2696	0.2287	0.0792	0,2948-	0,2617-	0,2267-	0,0825+

Table 1: Query statistics.

Year	$n(Q^t)$	$n(Q^c)$	$l(Q^t)$	$l(Q^c)$
2006	1.24	1.03	2.32	2.12
2007	1.39	1.12	1.73	1.94

4.1 Data Set and Evaluation Measures

We have performed several experiments with the collections and CAS queries at INEX 2006[4] and INEX 2007[5] in order to validate our proposal. The XML document collection considered in the experimentation is the one used in the last editions of the INEX Conference, namely Wikipedia (an XML version of the English Wikipedia), at the beginning of 2006 (Denoyer and Gallinari, 2006) with its 659388 articles (around 4600 Megabytes in size).

In terms of the queries (and the corresponding relevance assessments) used in our experiments, we have used a subset of the queries developed for INEX'2006 (34 CAS queries) and INEX'2007 (36 CAS queries). Particularly, we consider those queries having CAS restricctions. Table 1 shows some statistics of these queries. Second and third columns show the mean number of subqueries for the target and context component, and fourth and fifth present the mean lengths of the queries.

In those queries where there is no structural restrictions, as for example the case of "find any structural unit that contains a paragraph dealing with *term*"[6], we have considered the *focused task* of INEX Ad-hoc track (Kamps et al., 2008). The objective is to retrieve the most relevant parts of the documents, without overlapping. Therefore, if a chapter is retrieved, and therefore, considered the most appropriate type of element that matches the query, none of its sections should be retrieved.

The measures of retrieval effectiveness are those used in the focused task of the INEX'2007 ad hoc track, namely the interpolated precision (iP) at selected recall levels (iP[0.01], iP[0.05] and iP[0.10]) and the average interpolated precision (AiP), all of them averaged across the 70 queries.

4.2 Evaluation Methodology

The objective is to compare the results returned by the search engine for the original and expanded queries, which are generated after knowing the relevance of the first 10 structural units of the result list (this relevance information is obtained from the relevance assessments of INEX), so we can test if the expansion of CAS queries with our approach is interesting.

This comparison is not an easy task because the results from the expanded query could apparently be very positive due to the fact that the elements judged as relevant will probably appear in the first positions of the result list of the expanded query (this fact is called *ranking effect*). The use of this data for the *training* in the evaluation has an overfitting effect which artificially improves the results. For this reason, there are different techniques to evaluate RF methodologies like *residual collection* or *freezing* (Chang et al., 1971).

In our case, we will use the residual collection method, but it has been adapted for structured documents (because we must take into account that if one unit has been judged as relevant, then we have relevance information about other units which could appear in the result list being ancestors or descendants of it) and the *focused* task. Therefore, the method works as follows:

- *Original query*: We consider the original ranking

[4]http://inex.is.informatik.uni-duisburg.de/2006/

[5]http://inex.is.informatik.uni-duisburg.de/2007/

[6]In NEXI language, queries having the wildcard symbol * as for example "//*[about(//p, term)]".

of the system, after removing the first 10 judged structural units. This ranking will be considered the baseline.

- *Expanded query*: The system runs the new expanded query and then all the structural units overlaping with the first 10 judged elements are removed.

Then, these two retrieval lists could be compared in order to measure the impact of RF, computing a percentage of change in the performance measures.

4.3 Experimental Results

The first experiments that we have designed and run under this evaluation framework, are based on the expansion of the sub-queries using different number of terms and the expansion in different parts of the original query. With the first parameter, we would like to know the impact in the query performance when the number of terms increases. With respect to the part of the CAS query to be expanded, we are interested in knowing which part of the query is more appropriate for expansion. The number of terms used for the expansion is ranged from 1 to 10 and the expanded parts are:

- Full expansion: All the sub-queries of the structural query are expanded.
- Context expansion: The sub-queries of the context part are expanded. In this case we will consider both the soft and hard version of the non-relevance assumption.
- Target expansion: The sub-queries of the target part are expanded.

The results of these experiments are summarized in Tables 2 and 3. They display in the first row the corresponding effectiveness measures for the baseline (*Original query*) and for each one of the experimental settings.

Focusing on the context subqueries (Table 2) we use bold face to indicate the best results for each measure. Also, in those columns associated with the soft non-relevance assumption (right hand side of the table), we denote by the symblol $+$ (or $-$) when under the same number of expansion terms the soft version is better (worse) than the hard one and we use the symbol † to indicate that this option is better than the best results obtained using the best expansion for the hard version, i.e. to use only one expansion term.

From Table 2 some conclusions might be obtained, first that by expanding only the context subqueries we can obtain minors improvements with respect to the baseline (even in some cases the expansion of only the context part is not a good solution). Thus, the best results obtained for each measure achieve the following improvements: 2.30% for iP[0.01], 7.88% for iP[0.05], 14.09% for iP[0.10] and 2.16% for AiP. Second, me might say that the performance of the soft version outperforms the results of the hard version when, under the same circumstances, we compare 1-to-1, but soft version barely outperforms the best results of the hard version, obtained when adding only one expansion term. As general conclusion we can say that it is better to consider the hard version for the non-relevance assumption. This is because the soft version, which best results seem to be obtained using 6 expansion terms, does not get, in mean, better results than the best hard version. Therefore, in the rest of the paper we will only consider the hard version.

Focusing on Table 3, we present the results obtained by expanding only the target subqueries and also when expanding both subqueries, target and context (in its hard version), named full expansion. In the case of full expansion we have used the same number of expansion terms in all the subqueries. In this table we highlight with bold face the best results obtained for each measure.

The first conclusion that can be obtained is that the results obtained in this table are much better than the ones obtained using context expansion. Comparing full and target expansions, the first one, in general, obtains better results in all the measures with significant differences although it is smaller in the AiP measure. With respect to the number of expansion terms, using from 3 to 5 terms for every sub-query is the best solution because the improvements we can achieve go from 19.57% to 23.57% for the AiP measure in the full expansion and from 19.77% to 22% in the target expansion, but these improvements are higher if we consider other measures like the ip[0.10], where the percentages are ranging from a minimum of 39.92% to a maximum of 43.90% in the full expansion and from a minimum of 28.57% to a maximum of 33.48% in the target expansion.

We believe that this best performance for these numbers of terms is due to the short length of the subqueries (around two terms in average, with a maximum of four terms). So, if we consider fewer terms (1-2) we can also get some improvements but they are not so significant, and the use of more than 6 terms provokes the decreasing of performance because we could be introducing non representative terms in the expanded sub-queries, thus changing the focus of the original query and hence generating worse results.

Another important conclusion is that context and target expansion reinforce each other. In most of the

Table 3: Results of the experiments for full and target expansion.

	Full Exp.				Target Exp.			
k	iP[0.01]	iP[0.05]	iP[0.10]	AiP	iP[0.01]	iP[0.05]	iP[0.10]	AiP
Without RF	0.3460	0.2589	0.2214	0.0831	0.3460	0.2589	0.2214	0.0831
1	0.4239	0.3267	0.2848	0.0920	0.4080	0.3196	0.2716	0.0911
2	0.4361	0.3439	0.2923	0.0939	0.4107	0.3349	0.2744	0.0943
3	**0.4433**	0.3558	0.2984	0.0952	0.4177	0.3343	0.2730	0.0959
4	0.4356	**0.3648**	**0.3186**	**0.1028**	**0.4312**	0.3392	**0.2955**	**0.1013**
5	0.4407	0.3604	0.3116	0.0993	0.4294	**0.3437**	0.2895	0.0995
6	0.4137	0.3530	0.3098	0.0998	0.4063	0.3255	0.2847	0.0995
7	0.4245	0.3488	0.3111	0.0986	0.4147	0.3230	0.2784	0.0982
8	0.4158	0.3444	0.3106	0.0984	0.3907	0.3092	0.2790	0.0975
9	0.4188	0.3454	0.3059	0.0976	0.4008	0.3091	0.2733	0.0971
10	0.4283	0.3505	0.3078	0.1006	0.3938	0.3036	0.2707	0.0982

Table 4: Results of the experiments varying the number of expansion terms (k) in context sub-queries and fixed to 4 in target sub-queries.

k	iP[0.01]	iP[0.05]	iP[0.10]	AiP
Without RF	0.3460	0.2589	0.2214	0.0831
1	0.4355	0.3591	0.3121	0.1035
2	**0.4427**	**0.3700**	0.3164	**0.1040**
3	0.4349	0.3643	0.3159	0.1019
4	0.4356	0.3648	0.3186	0.1028
5	0.4401	0.3681	**0.3202**	0.1015
6	0.4326	0.3633	0.3196	0.1004
7	0.4345	0.3626	0.3178	0.0995
8	0.4313	0.3568	0.3160	0.0994
9	0.4229	0.3461	0.3067	0.0972
10	0.4269	0.3510	0.3122	0.0979

experiments we have that the measure's values of the full expansion are strictly greater than the ones obtained by expanding context or target isolately. Moreover, the improvement's percentages obtained with a full expansion are greater than the sum of the improvements obtained by context and target expansion.

Once we know that full expansion is the most appropriate approach to expand CAS queries, we could wonder how a different number of expansion terms in the context and targets sub-queries could affect the retrieval performance. In order to answer this question we have designed the following experiment: Firstly, we fix to 4 the number of expansion terms for the target subquery (we assume that a good performance is achieved with this configuration). Then, a new experiment has been run by varying the number of expansion terms in the context subqueries from 1 until 10. Table 4 shows the absolute results obtained (in boldface the best results for each measure).

Observing this table, we could conclude that the consideration of a different number of terms used to expand in both parts of the CAS query is relevant in terms of retrieval performance. In general, the best overall results are obtained by using a different number of terms: four in the target (fixed) and two terms in the context, although it represents a minor improvement (around 1%) with respect to using four terms also in the context. These results agree with the ones obtained expanding only context in the original query, as a large number of expansion terms in the context seems to worse the performance. It seems more effective to leave the original context sub-queries unchanged since, for retrieval purposes, they are less important than the target sub-queries.

5 CONCLUDING REMARKS AND FUTURE RESEARCH

We present in this paper a new relevance feedback framework for content and structure (CAS) queries. This approach modifies the original queries keeping the same structural restrictions but expanding the keyword sub-queries with a best top weighted terms. This framework has been evaluated with INEX 2006 and

INEX 2007 collections.

Our future work will be concentrated on determining different rules for the expansion of sub-queries studying different features of the original one with the objective of getting the best number of terms for the expansion of the sub-query. This process should be automatic adapting the query to the user's needs. Also it is interesting to study how to apply negative feddback in this framework, i.e. how to use the terms in the non relevant units in order to best capture the real intention of the user query.

Another interesting issue could be to modify structural constraints (only in the context) in the new expanded query. The objective part should not be changed because this indicates the structural unit which the user is interested in, so it must be the same in both original and expanded queries but it could be interesting to change, in some cases, the context restrictions.

ACKNOWLEDGEMENTS

This paper has been supported by the Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía and the Spanish Ministerio de Ciencia de Innovación by means of the projects P09-TIC-4526 and TIN2008-06566-C04-01, respectively.

REFERENCES

Chang, Y., Cirillo, C., and Razon, J. (1971). Evaluation of feedback retrieval using modified freezing, residual collection & test and control groups. *The SMART Retrieval System: Experiments in Automatic Document Processing, chapter 17*, pages 355–370.

Crouch, C. J., Mahajan, A., and Bellamkonda, A. (2005). Flexible XML retrieval based on the extended vector model. In N. Fuhr, M. Lalmas, S. M. and Szlávik, Z., editors, *INEX 2004. Lecture Notes in Computer Science*, volume 3493, pages 292–302. Springer, Heidelberg.

de Campos, L. M., Fernández-Luna, J. M., Huete, J., and Romero, A. (2006). Garnata: An information retrieval system for structured documents based on probabilistic graphical models. In *Proceedings of the IMPU'06 conference*, pages 1024–1031.

de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., and Martín-Dancausa, C. (2009). Content-oriented relevance feedback in XML-ir using the Garnata Information Retrieval system. *FQAS 2009, Lecture Notes in Artificial Intelligence*, 5822:617–628.

Denoyer, L. and Gallinari, P. (2006). The wikipedia XML corpus. *SIGIR Forum*, 40:64–69.

Fourati, I., Tmar, M., and Hamadou, A. (2009). Structural relevance feedback in XML retrieval. *FQAS 2009. Lecture Notes in Artificial Intelligence*, 5822:168–178.

Hlaoua, L., Sauvagnat, K., and Boughanem, M. (2006). Structure-oriented relevance feedback in XML retrieval. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 780–781. ACM.

Kamps, J., Pehcevski, J., Kazai, G., Lalmas, M., and Robertson, S. (2008). INEX 2007 evaluation measures. *INEX 2007. Lecture Notes in Computer Science*, 4862:24–33.

Lalmas, M. (2009). *XML Retrieval*. Morgan&Claypool.

Mass, Y. and Mandelbrod, M. (2004). Relevance feedback for XML retrieval. In *INEX 2004 Workshop*, pages 154–157.

Mihajlovic, V., Ramirez, G., de Vries, A. P., Hiemstra, D., and Blok, H. E. (2004). Tijah at inex 2004 modeling phrases and relevance feedback. pages 276–291. INEX 2004 Workshop Proceedings.

Pan, H., Theobald, A., and Schenkel, R. (2004). Query refinement by relevance feedback in an XML retrieval system. pages 854–855. ER 2004.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. *The SMART retrieval system-experiments in automatic document processing*, pages 313–323.

Ruthven, I. and Lalmas, M. (2003). A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(2):95–145.

Schenkel, R. and Theobald, M. (2006). Structural feedback for keyword-based XML retrieval. In Lalmas, M., editor, *In Advances in Information Retrieval. 28th European Conference on IR Research (ECIR 2006) Lecture Notes in Computer Science*, volume 3936, pages 326–337. Springer.

Sigurbjörnsson, B., Kamps, J., and de Rijke, M. (2004). The university of amsterdam at inex 2004. pages 104–109. INEX 2004 Workshop.

Trotman, A., Sigurbjörnsson, B., Fuhr, N., Lalmas, M., Malik, S., and Szlvik, Z. (2005). Narrowed extended XPath i (NEXI). In *Lecture Notes in Computer Science*, volume 3493, page 1640. Springer Verlag, Heidelberg.