Compact Representations in XML Retrieval

Fang Huang, Stuart Watt, David Harper, and Malcolm Clark

School of Computing, The Robert Gordon University, Scotland
{fah,sw,djh,mc}@comp.rgu.ac.uk

Abstract. This paper describes the participation of the Information Retrieval and Interaction group of Robert Gordon University in the INEX 2006 ad hoc track. We focused on two questions: "What potential evidence do human assessors use to identify relevant XML elements?" and "How can this evidence be used by computers for the same task?". Our main strategy was to investigate evidence taken not only from the content, but also from the shallow features of how texts were displayed. We employed the vector space model and the language model combining estimates based on element full-text and the compact representation of the element. We analyzed a range of non-content priors to boost retrieval effectiveness.

1 Introduction

In this paper, we describe our participation in the INEX 2006 ad hoc track. We conducted experiments to retrieve XML elements that have similar features judged to be relevant by human assessors. The criteria used by a human assessor for judging relevance involve a complex variety of individual factors. However, it is evident that not every word of a given document catches the reader's eye. In most cases, people judge a document's relevance by skimming over the titles, section titles, figures, tables, and words emphasized in bold or larger fonts. We proposed, in this paper, to extract and put together all those most representative words to build a compact form of a document (or an XML element). We employed retrieval models that emphasized the importance of the compact form in identifying the relevance of an XML element. We also conducted preliminary experiments to investigate the potential features of human-retrieved elements regardless of their content, and introduced a range of priors to the language model to enhance retrieval effectiveness.

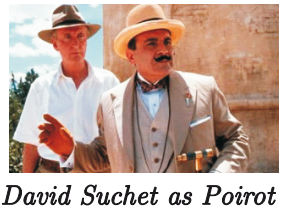
The remainder of this paper is organized as follows: Section 2 outlines our ideas for building a compact representation of an XML element. Section 3 describes the vector space model we used. The mixture language model is presented in section 4. Section 5 offers a detailed description and discussion of our INEX experiments and results. The final part, section 6, concludes with a discussion and possible directions for future work.

N. Fuhr, M. Lalmas, and A. Trotman (Eds.): INEX 2006, LNCS 4518, pp. 64–72, 2007.

2 Compact Representation of an XML Element

Even a brief look at the procedure adopted by a human assessor when judging a document's relevance to a query, shows quite clearly that not all the details of the document are taken fully into consideration. Some words attract more or less of the reader's attention. It is our contention that words appearing in titles, figure captions, or printed in bold, italics, larger fonts, and different colors are frequently more representative of the document's relevance. Figure 1 shows a sample text taken from a document named "Hercule Poirot" from the Wikipedia collection. Extracting words from the figure caption, and words that are underlined, or displayed in bold or larger size, we get a list of words containing "Hercule Poirot fiction character Belgium England World War I Private detective Arthur Hastings". This list of words provides a clue to the meaning and content of the original text. We therefore believe it can be used to enhance retrieval effectiveness. Furthermore, the XML structure of the INEX collection allows automatic locating of these words displayed in certain formats as they are marked-up in specific tags. Based on this consideration, we proposed to extract

Hercule Poirot is a fiction character, the primary detective of Agatha Christies novels. He appears in over 30 books. The character was born in Belgium, and has worked as a Belgian police officer, but moved to England during World War I and started a second career as a private detective. Poirotis remarkable for his small stature and egg-shaped head, his meticulous moustache, his dandified dressing habits, his absolute obsession with order and neatness, and his disdain for detective methods that include crawling on hands and knees and looking for clues. He prefers to examine the psychology of a crime, once even betting his best friend and sometime partner, Arthur Hastings, that he could solve a case simply by sitting in an easy chair and using his "little grey cells."



David Suchet as Poirot

Fig. 1. A sample text

these representative words from the original text to build a compact form of a text and emphasize its importance in identifying the relevance of the text. In our experiments, retrieval units were XML elements. Consequently, the method was adapted to XML element-based (the whole document can be considered as

an XML element as well), i.e., for each XML element, we built a compact representation of it by extracting words from titles, subtitles, figure captions, and words printed in bold, italics or larger fonts from the text nested by the element. The compact form was then used in our retrieval experiments based on vector space model and mixture language models.

3 Vector Space Model

We used the vector space model based on the default similarity measure in Lucene[4]. For a collection C, the similarity between a document d and the query q is measured by:

$$sim(q,d) = \sum_{t \in q} \frac{tf_{t,q} \cdot idf_t}{norm_q} \cdot \frac{tf_{t,d} \cdot idf_t}{norm_d} \cdot coord_{q,d} \cdot weight_t \tag{1}$$

where

$$tf_{t,x} = \sqrt{freq(t,X)} \tag{2}$$

$$idf_t = 1 + log \frac{|C|}{freq(t,C)} \tag{3}$$

$$norm_q = \sqrt{\sum_{t \in q} tf_{t,q}.idf_t^2} \tag{4}$$

$$norm_d = \sqrt{|d|} \tag{5}$$

$$coord_{q,d} = \frac{|q \cap d|}{|q|} \tag{6}$$

$weight_t = 1$ for all term t. In our experiment, retrieval units were XML elements. An element's relevance was measured based on the element's full-text and the compact representation of the element, i.e.,

$$sim(q,e) = \frac{sim(q,e_{full}) + sim(q,e_{compact})}{2} \tag{7}$$

where e is an XML element, e_{full} is the full text nested in element e, and $e_{compact}$ is the compact form of element e.

4 Language Model

We present here a retrieval model based on the language model, i.e., an element's relevance to a query is estimated by

$$P(e|q) \propto P(e) \cdot P(q|e) \tag{8}$$

where e is an XML element; q is a query consisting of the terms t_1,...,t_k; the prior, $P(e)$, defines the probability of element e being relevant in absence of a query; $P(q|e)$ is the probability of the query q, given element e.

4.1 Probability of the Query

Assuming query terms to be independent, $P(q|e)$ can be calculated according to a mixture language model:

$$P(q|e) = \prod_{i=1}^{k} (\lambda \cdot P(t_i|C) + (1 - \lambda) \cdot P(t_i|e)) \tag{9}$$

where λ is the so-called smoothing parameter; C represents the whole collection. $P(t_i|C)$ is the estimate based on the collection used to avoid sparse data problem.

$$P(t_i|C) = \frac{doc\_freq(t_i, e)}{\sum_{t' \in C} doc\_freq(t', C)} \tag{10}$$

The element language model, $P(t_i|e)$, defines where our method differs from other language models. In our language model, $P(t_i|e)$ is estimated by a linear combination of two parts:

$$P(t_i|e) = \lambda_1 \cdot P(t_i|e_{full}) + (1 - \lambda - \lambda_1) \cdot P(t_i|e_{compact}) \tag{11}$$

where λ_1 is a mixture parameter; $P(t_i|e_{full})$ is a language model for the full-text of element e; $P(t_i|e_{compact})$ is the estimate based on the compact representation of element e. Parameter λ and λ_1 play important roles in our model. Previous experiments[1,8] suggested that there was a correlation between the value of the smoothing parameter and the size of the retrieval elements. Smaller average sizes of retrieved elements require more smoothing than larger ones. In our experiments, the retrieval units, which are XML elements, are relatively small. Consequently, we set a large smoothing parameter $\lambda = 0.3$ and used equal weights for the full text and the compact representation, i.e., $\lambda_1 = 0.35$.

4.2 Element Priors

The Prior $P(e)$ defines the probability that the user selects an element e without a query. Elements are not equally important even though their contents are ignored. Several previous studies[1,6] reported that a successful element retrieval approach should be biased toward retrieving large elements. Furthermore, we believe relevant elements are more likely to appear in certain parts of a document, e.g., the title, the first paragraph, the first section, etc.

We conducted a preliminary experiment to investigate potential non-content features that might be used to boost retrieval effectiveness. The features considered included size, type, location, and the path length of an element. Location was defined as the local order of an element ignoring its path. The path length of an element equals to the number of elements which nest it. For example, for an element /article[1]/p[1] (the first paragraph in the document), type of this element is 'paragraph', location is represented as '1' (the first paragraph), and the path length is 1. The main objective of our experiment was to find out the

distribution of the above features among the relevant elements. Two human assessors were asked to search the Wikipedia collection, retrieve relevant XML elements, and analyze retrieved results. Details of the procedure were: i) query creation: we created 216 queries. A query was a list of keywords or a one-sentence description of the information need which was written in natural language and without regard to retrieval system capabilities or document collection peculiarities. ii) element retrieval: in this step, each query created in the previous stage was used to explore the Wikipedia collection. The TopX[7] XML search engine, which is provided through the INEX website, was used for this task. Human assessors judged the top 100 results retrieved by TopX for each query, assessed the relevance of each of the retrieved elements, and recorded the path for each of the relevant elements. iii) feature distribution analysis: paths for relevant elements were analyzed automatically by a computer program. Results are shown in Table 1. Part (a) of the table shows that the total number of relevant elements is 9142. Among these elements, most of them are articles, sections, and paragraphs. The total number of elements of these three types is 8522, which accounts for 93.2% of the total amount. Part (b) shows the relevant elements tend to appear in the beginning parts of the text. The whole documents are excluded in this analysis, as the location feature is not applicable for the whole documents. The total number of the elements excluding whole documents is 3316. Elements with location value of '1', '2', '3' account for 88.1%(2920 out of 3316) of the total amount. Part (c) shows relevant elements are not likely to be nested in depth. Again, whole documents are excluded. Elements that only nested by the whole article (path-length=1, e.g., /article/section, /article/p, etc.) constitute the majority (2089 out of 3316, i.e., 63.0%). Only 8.4% (280 out of 3316) of relevant elements are of path length longer than 3.Our preliminary experiments indicated that

Table 1. Distribution of element shallow features

(*a*)		(*b*)		(*c*)	
type	number	location-value	number	path-length	number
article	5826	1	1588	1	2089
section	2098	2	789	2	835
paragraph	598	3	543	3	112
others	620	≥ 4	396	≥ 4	280
total	9142	total	3316	total	3316

relevant elements had some non-content features which could be used to boost retrieval effectiveness. We did not analyze the size of the elements in our experiment, because some studies[1,6] have already concluded that a successful element retrieval approach should be biased toward retrieving large elements.

Based on the above observation, consider non-content feature set F={$|e|$, $|e_{path}|$, $e_{location}$}, where $|e|$ is the size of element e measured in characters; $|e_{path}|$ is the path length of e; and $e_{location}$ is the location value of e. Assuming features are independent, we calculated the prior $P(e)$ by the following equation:

$$P(e) = \prod_{i=1}^{3} P(e|F_i) \tag{12}$$

where F_i is i^{th} feature in set F. In the experiments, we chose a uniform length filter to ensure the retrieval of larger sized XML elements. The threshold used to filter out short elements was set to 120 characters, i.e.,

$$P(e|\,|e|) = \begin{cases} 1 & |e| \geq 120 \\ 0 & otherwise \end{cases} \tag{13}$$

The decision to measure in characters instead of words was based on the consideration that smaller segments such as "I like it." contains little information, while a sentence with three longer words tends to be more informative.

$P(e||e_{path}|)$, the prior based on e_{path} in our experiments was calculated by:

$$P(e|\,|e_{path}|) = \frac{1}{1 + |e_{path}|} \tag{14}$$

$P(e|e_{location})$, the prior based on the location value was calculated by:

$$P(e|e_{location}) = \frac{1}{e_{location}} \tag{15}$$

5 INEX Experiments

In this section, we present our INEX experiments in participating the Thorough task.

5.1 Index

We created inverted indexes of the collection using Lucene[4]. Indexes were word-based. All texts were lower-cased, stop-words removed using a stop-word list, but no stemming. For each XML element, all text nested inside it was indexed. In addition to this, we added an extra field which corresponded to the compact representation of the element. The indexing units could be any types of XML elements. However, due to the time restrictions placed on our experiments, we only indexed three types of elements: article, section, and paragraph.

5.2 Query Processing

Our queries were created using terms only in the <title> or <description> parts of topics. Like the index, queries were word-based. The text was lower-cased and stop-words were removed, but no stemming was applied. '+', '-' and quoters in queries were simply removed. We processed the <description> part of topics by identifying and extracting noun phrases[5] to form queries.

5.3 Runs and Results

We submitted a total of six runs to the Thorough task.

1. VSM and nl-VSM: runs using vector space model based on full-text and compact representation of elements. For VSM, queries were created using terms in the <title> field. And queries for nl-VSM were created from the <description> parts.
2. LM-2 and nl-LM-2: runs created using mixture language model based on full-text and compact representation of elements. Queries for the runs were created from <title> and <description> fields, respectively.
3. LM-1 and nl-LM-1: runs created using language model based on compact representation of elements only, i.e., equation (11) in section 4 is replaced by

$$P(t_i|e) = (1 - \lambda) \cdot P(t_i|e_{compact}) \tag{16}$$

where $\lambda = 0.3$. Queries for the runs were created from <title> and <description> fields, respectively.

Table 2 lists our results in the INEX official evaluation: the system-oriented MAep measure and the ranks among all submitted runs. Details of the evaluation metrics can be found in[3]. We were ranked at 39, 42, 66, 67, 68, and 69 out of 106 submissions. The vector space model based on combination of full-text and compact representation outperformed the language models. For runs using language models, estimates based on compact representation only(i.e., LM-1 and nl-LM-1) achieved comparable (and slightly better) performances with estimates based on the combination of full-text and compact representation (i.e., LM-2 and nl-LM-2). This confirmed our hypothesis that the compact representation generated by extracting words from the original text is effective for element retrieval. Due to the pressure of time, we did not submit baseline runs for the retrieval models based on full-text solely for comparison.

Results of each pair of runs using the same retrieval method (e.g., VSM and nl-VSM) show no significant difference. This prompts us that natural language queries work quite well after some shallow pre-processing.

Table 2. Thorough results using filtered assessments

RunID	MAep	Rank
VSM	0.0294	39/106
nl-VSM	0.0290	42/106
LM-1	0.0180	66/106
nl-LM-1	0.0174	68/106
LM-2	0.0178	67/106
nl-LM-2	0.0172	69/106

6 Conclusions and Future Work

We have presented, in this paper, our experiments for the INEX 2006 evaluation campaign. We assumed important words could be identified according to the ways they were displayed in the text. We proposed to generate a compact representation of an XML element by extracting words appearing in titles, section titles, figure captions, tables, and words underlined or emphasized in bold, italics or larger fonts from the text the element nesting. Our retrieval methods emphasized the importance of these words in identifying relevance. Results of the Thorough task showed that estimates based solely on compact representation performed comparably with estimates using combinations of full-text and compact representation. This indicated that compact representation provided clues to content of the original element, as we had assumed.

We also investigated a range of non-content priors. Our preliminary experiment indicated that relevant elements tended to be larger elements, such as whole articles, sections, paragraphs. Furthermore, relevant elements were more likely to appear in certain locations, such as the first element (e.g. first paragraph) of a document. And they were not likely to be deeply nested in the structure. We implemented priors in our language models, but the limited time at our disposal meant that we could not submit baseline runs for comparisons of how these priors work.

Our future work will focus on refining the retrieval models. Currently, the compact representation of an element is generated by words from certain parts of the text. However, the effectiveness of this method depends on the type of the documents. For example, in scientific articles, section titles (such as introduction, conclusion, etc) are not very useful for relevance judgement, whereas section titles in news reports are very informative. In the future, we will explore different patterns for generating compact representations depending on types of texts. This might involve genre identification techniques. We will investigate different priors' effectiveness and how different types of evidence can be combined to boost retrieval effectiveness.

Acknowledgments

We would like to thank Ulises Cervino Beresi for his help in indexing tasks.

References

1. Kamps, J., Marx, M., de Rijke, M., Sigurbjornsson, B.: XML retrieval: What to retrieve? In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2003)
2. Kamps, J., de Rijke, M., Sigurbjornsson, B.M.: Topic field selection and smoothing for XML retrieeval. In: Proceedings of the 4th Dutch-Belgian Information Retrieval Workshop (2003)
3. Kazai, G., Lalmas, M.: INEX 2005 evaluation metrics. In: Fuhr, N., Lalmas, M., Malik, S., Kazai, G. (eds.) INEX 2005. LNCS, vol. 3977, Springer, Heidelberg (2006)

4. Lucene. The Lucene search engine (2005) http://jakarta.apache.org/lucene
5. Ramshaw, L., Marcus, M.: Text chunking using transformation-based learning. In: Proceedings of the Third ACL Workshop on Very Large Corpora (1995)
6. Sigurbjornsson, B., Kamps, J., de Rijke, M.: An element-based approach to XML retrieval. In: INEX 2003 Workshop Proceedings (2004)
7. Theobald, M., Schenkel, R., Weikum, G.: An Efficient and Versatile Query Engine for TopX Search. In: Proceedings of the 31th International Conference on Very Large Databases (VLDB), Trondheim, Norway (2005)
8. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2001)