NTCIR-5 CLIR Experiments at QUT

David Lu, Shlomo Geva, Yue Xu, and Yuefeng Li
School of Software Engineering and Data Communications
Queensland University of Technology
Brisbane, QLD 4001, Australia
c4.lu@student.qut.edu.au, {s.geva, yue.xu, y2.li}@qut.edu.au

Abstract

The Information Retrieval and Web Intelligent (IR-WI) research group is a research team at the Faculty of Information Technology, QUT, Brisbane, Australia. The IR-WI group participated in the NTCIR 5 for the first time. This paper focuses on our participation in the CLIR task. For this track, we experiment our XML search engine within the NTCIR English document collection. Our results indicate that in general, our XML search engine is not suitable for non-structured document-level information retrieval due to the difference of document structure.

1 Introduction

Information Retrieval (IR) is one of the most influential, challenging and aspired fields in today's world. QUT's Information Retrieval and Web Intelligent (IR-WI) group is a team of researchers investigating IR and other associated technologies such as data mining, web intelligence and recommendation systems. In previous years, our group has participated in the Initiative for the Evaluation of XML Retrieval (INEX). We have developed a XML document search engine – GPX for the INEX task. GPX produces results that are comparable with the best alternatives at INEX. This year we participated in the NTCIR CLIR task and tested all the 50 topics of the English collections. The main purpose is to discover the difference between document-level information retrieval and the XML information retrieval. The As the NTCIR collection contains well formatted XML like documents, we can use our indexer and search engine with only a little change.

2 Indexing

The documents were indexed using the word-base indexing approach. The indexer was originally developed for INEX and it is basically used for indexing XML documents. When indexing, the indexer will record the term, the term position in the context (context position), the term position in the article (global position), the context name (XPATH) and also the article ID. The words were stemmed using porter stemmer and stop words were removed from the index to reduce the size of index file.

3 Searching

Because word-base indexing could not capture phrase, when searching query includes phrase, we used context position for phrase retrieval. Only words in the same context and their positions are in conjunction would be treated as term. Exact phrase will be given highest score. As an enhanced, context position can be used for deciding term proximity. If words appear in the same context but are not in conjunction we will treat the words as partial phrase and will be given higher score. And words appear in the same document but in different context will have lowest score.

4 Ranking Scheme

In our scheme, XML leaf and branch elements are treated differently. In the NTCIR collection, leaf elements are usually with <P> tags or sometimes, only with <TEXT> tags. Our inverted list mostly stores information about leaf elements. A leaf element is considered a candidate for retrieval if it contains at least one query term. A branch node is candidate for retrieval if it contains a relevant child element. Once an element (either leaf or branch) is deemed to be a candidate for retrieval its relevancy judgment score is calculated. A heuristically derived formula is used to calculate the relevance judgment score of leaf elements which is given below. The score is determined from query terms contained in the element. It penalizes elements which contain query terms appearing frequently in the collection, and it rewards elements with evenly distributed query term frequencies within the elements.

$$L = K^{n-1} \sum_{i=1}^{n} t_i$$

Here n is the number of unique query terms contained within the leaf element, K is a small integer (we used K=5). The term K^{n-1} scales up the score of elements having multiple distinct query terms. The system is not sensitive to the value of K – we experimented with K=3 to 10 with little difference in results. The sum is over all terms where it is the frequency of the ith query term in the leaf element. This sum rewards the repeat occurrence of query terms, but uncommon terms contribute more than common terms. Once the relevance judgment scores of leaf elements have been calculated, they can be used to calculate the relevance judgment score of branch elements. In the NTCIR collection, branch elements are usually the root of the documents. Thus we just sum up all the leaf scores for the document. Our ranking scheme is quite simple but with high performance. Our experiments on INEX 2004 indicate that our search engine performed best in INEX04 conference.

5 Results

Due to the limited time, we only did experiments on English collections this year. The results for the official submission are performed poorly. In most of the cases, our results are sitting at the bottom of all the participants' results. Our MAP is only 0.228 on official runs. By analysing the results we found that there are a few reasons that we have such low results. We believe that the low MAP is caused by the difference between XML IR and document level IR. In XML IR, we retrieve the document components (XML nodes) instead of whole documents. In our ranking strategy, if we have more words matching at the same XML node, the node will have higher score. This strategy works perfectly in the INEX collection. In the INEX collection, the leaf nodes of a document are usually a few sentences. Therefore we wound not have to think about the proximity in the score calculation. However, XML nodes in the NTCIR collection are very large. In the NTCIR collection, the leaf nodes are usually a large piece of text thus we have to consider the proximity in the score calculation. One of the good examples is topic 13, details as the table showed below.

Position	P5	P10	P15	P20	P30	P100	P200	P500	P1000
Precision	0.4	0.2	0.13	0.1	0.1	0.09	0.065	0.034	0.036
Documents	2	2	2	2	3	9	13	17	36

Table 1: Precision of topic 13

As we can see, our search engine cannot really find relevant document until P100. The query for topic 13 is "Taliban, Destroy Buddhism". Our search engine will give higher rank for the document that have more matching in the word "Taliban". As "Taliban" has much higher frequency than "Destroy Buddhism" in the collection. Also we are not using any idf*tdf in the calculation of the score, the documents with a lot of "Taliban" will rank very high even they do not have the words "Destroy Buddhism". Therefore all the relevant documents are appeared at the bottom of our rank list.

Although our results are not good, we still have some topics that yield the best or very close to the best results such as topic 23, 24 and 40. It seems that if the search keywords are common words, our search engine outperformed other search engines. However, further study is needed.

6 Query Expansion

In our experiment, two methods of query expansion were investigated: plurals/singular expansion and Porter stemming. Plural and singulars were added using lexical-based heuristics to determine the plural form of a singular term (and vice-versa) Porter stemmer was performed on the existing query terms to derive each of their stems. The results are shown as table 2.

	No expansion	Plural and singulars	Porter stemming
MAP	0.1031	0.2581	0.2640
P10	0.2300	0.3857	0.3939

Table2. MAP, P10 Results on expansion

As we can see, plural/singulars and porter stemming provide similar performance. Porter stemmer does not show much higher benefit for the performance. In most cases, porter stemmer expansion provides the same performance as plural/singulars expansion does. We believed that although the porter stemmer can expand the query with more words, it also brings noise into the query in some cases. Therefore its overall performance does not improve much. For example, in topic 39, we are looking for "Windows, Linux, competition". Competition will be expanded as "competitors competiting competition competitions competitive competitively, competitiveness, competitivity". Some of those words are not relevant to our topic. Therefore we see a 10% performance drop. The only topic we can see performance yield is topic 28. The search words for topic 28 are "Bubka, human bird, retirement". As the search engine received the benefit from expending the word "retirement". Words with same stem such as "Retire retired retirement retirements retires retiring retirement" are expended. All those expanded words are relevant to the search topic. Thus we see the precision is 0.1974 for plural/singulars expansion and 0.3872 in porter stemming expansion.

7 Conclusion

We conducted a series of experiments on existing XML document information retrieval system. We completed two sets of experiments with two different query expansions. Our results indicate that our XML document search engine is not suitable for NTCIR document level retrieval. Although our search engine performed best in XML retrieval, it performed worst in the NTCIR task due to the difference structure of documents. Our experiments also used three different sets of inputs: a standard NTCIR title and expanded queries. Our results indicate that query expansion can improve precision over 100%. Although porter stemming can expand more words for the query, it provide similar performance as plural/singulars expansion. We will continue to research on GPX search engine and will participate in the CLIR task. of next NTCIR workshop.