The Impact of Document Level Ranking on Focused Retrieval

Jaap Kamps[1,2] and Marijn Koolen[1]

[1] Archives and Information Studies, Faculty of Humanities, University of Amsterdam
[2] ISLA, Faculty of Science, University of Amsterdam

Abstract. Document retrieval techniques have proven to be competitive methods in the evaluation of focused retrieval. Although focused approaches such as XML element retrieval and passage retrieval allow for locating the relevant text within a document, using the larger context of the whole document often leads to superior document level ranking. In this paper we investigate the impact of using the document retrieval ranking in two collections used in the INEX 2008 Ad hoc and Book Tracks; the relatively short documents of the Wikipedia collection and the much longer books in the Book Track collection. We experiment with several methods of combining document and element retrieval approaches. Our findings are that 1) we can get the best of both worlds and improve upon both individual retrieval strategies by retaining the document ranking of the document retrieval approach and replacing the documents by the retrieved elements of the element retrieval approach, and 2) using document level ranking has a positive impact on focused retrieval in Wikipedia, but has more impact on the much longer books in the Book Track collection.

1 Introduction

In this paper we investigate the impact of document ranking for focused retrieval by comparing standard document retrieval systems to element retrieval approaches. In the evaluation of focused retrieval as studied in INEX, document retrieval techniques have proven to be competitive methods when compared with sub-document level retrieval techniques[3]. Although focused approaches such as XML element retrieval and passage retrieval allow for locating the relevant text within a document, using the larger context of the whole document often leads to better document ranking [7]. Our aim is to investigate the relative effectiveness of both approaches and experiment with combining the two approaches to get the best of both worlds. That is, we want to exploit the better document ranking performance of a document retrieval strateties and the higher within-document precision of an element retrieval strategy. To study the impact of using the document retrieval ranking we perform our experiments on the two collections used in the INEX 2008 Ad hoc and Book Tracks; the relatively short documents of the Wikipedia collection and the much longer books in the Book Track collection.

The paper is structured as follows. First, in Section 2, we report the results for the Ad Hoc Track. Then Section 3 presents our retrieval approach in the Book Track. Finally, in Section 4, we discuss our findings and draw some conclusions.

2 Ad Hoc Track

For the INEX 2008 Ad Hoc Track we investigate several methods of combining article retrieval and element retrieval approaches. We will first describe our indexing approach, then the run combination methods we adopted, the retrieval framework, and finally per task, we present and discuss our results.

The document collection for the Ad hoc track is based on the English Wikipedia [14]. The collection has been converted from the wiki-syntax to an XML format [1]. The XML collection has more than 650,000 documents and over 50,000,000 elements using 1,241 different tag names. However, of these, 779 tags occur only once, and only 120 of them occur more than 10 times in the entire collection. On average, documents have almost 80 elements, with an average depth of 4.82.

2.1 Retrieval Model and Indexing

Our retrieval system is based on the Lucene engine with a number of home-grown extensions [5, 9]. For the Ad Hoc Track, we use a language model where the score for a element e given a query q is calculated as:

$$P(e|q) = P(e) \cdot P(q|e) \tag{1}$$

where $P(q|e)$ can be viewed as a query generation process—what is the chance that the query is derived from this element—and $P(e)$ an element prior that provides an elegant way to incorporate query independent evidence [4].

We estimate $P(q|e)$ using Jelinek-Mercer smoothing against the whole collection, i.e., for a collection D, element e and query q:

$$P(q|e) = \prod_{t \in q} \left((1 - \lambda) \cdot P(t|D) + \lambda \cdot P(t|e) \right), \tag{2}$$

where $P(t|e) = \frac{\text{freq}(t,e)}{|e|}$ and $P(t|D) = \frac{\text{freq}(t,D)}{\sum_{e' \in D} |e|}$.

Finally, we assign a prior probability to an element e relative to its length in the following manner:

$$P(e) = \frac{|e|^{\beta}}{\sum_{e} |e|^{\beta}}, \tag{3}$$

where $|e|$ is the size of an element e. The β parameter introduces a length bias which is proportional to the element length with $\beta = 1$ (the default setting). For a more thorough description of our retrieval approach we refer to [12]. For comprehensive experiments on the earlier INEX data, see [10].

Our indexing approach is based on our earlier work [2, 6].

- *Element index*: Our main index contains all retrievable elements, where we index all textual content of the element including the textual content of their descendants. This results in the "traditional" overlapping element index in the same way as we have done in the previous years [11].
- *Article index*: We also build an index containing all full-text articles (i.e., all wiki-pages) as is standard in IR.

For all indexes, stop-words were removed, but no morphological normalization such as stemming was applied. Queries are processed similar to the documents, we use either the CO query or the CAS query, and remove query operators (if present) from the CO query and the about-functions in the CAS query.

2.2 Combining Article and Element Retrieval

Our experiments with combining runs all use the same two base runs:

- *Article*: a run using the Article index; and
- *Element*: a run using the element index.

Both runs use default parameters for the language model ($\lambda = 0.15, \beta = 1.0$). As shown by Kamps et al. [7], article retrieval leads to a better document ranking, whereas element retrieval fares better at retrieving relevant text within documents. For the Ad hoc Focused task, where the retrieved elements of different documents maybe interleaved in the ranking, we would expect that element retrieval achieves high early precision, while document retrieval, given that it will return whole documents which are often not relevant in their entirety, will have lower early precision. On the other hand, we expect that a document retrieval approach will have relatively little difficulty identifying long articles that have large a fraction of text highlighted as relevant, and therefore return them in the top ranks. The first few returned documents will thus contain a relatively large fraction of all the highlighted text with good within-document precision, resulting in a fairly slow drop in precision across the first recall percentages. For the Relevant in Context task, where retrieved elements have to be grouped by document, and introducing a document ranking score and a within-document retrieval score, we expect the document retrieval approach to rank the documents better and with a perfect within-document recall (due to it retrieving all text in the document) have a reasonable within-document score. With element retrieval, we expect the within-document precision to be better than that of the document retrieval approach, but it will have less recall and a worse document ranking. We therefore assume that a combined approach, using the document ranking of an article level run with the within document element ranking of an element level run, outperforms both runs on the "in context" tasks.

We experiment with three methods of combining the article and element results.

1. *ArtRank*: retain the article ranking, replacing each article by its elements retrieved in the element run. If no elements are retrieved, use the full article.
2. *Multiplication*: multiply element score with article score of the article it belongs to. If an element's corresponding article is not retrieved in the top 1,000 results of the article run, use only the element score.
3. *CombSUM*: normalise retrieval scores (by dividing by highest score in the results list) and add the article score to each element score (if article is not in top 1,000 results for that topic, only element score is used). Thus elements get a boost if the full article is retrieved in the top 1,000 results of the article run.

Our Focused and Relevant in Context submissions are all based on the following base "Thorough" runs:

Table 1: Results for the Ad Hoc Track Focused Task (runs in emphatic are not official submissions)

Run	iP[0.00]	iP[0.01]	iP[0.05]	iP[0.10]	MAiP
Article	0.5712	0.5635	0.5189	0.4522	**0.2308**
Element	**0.6627**	0.5535	0.4586	0.4062	0.1710
ArtRank	0.6320	**0.6025**	0.5054	0.4569	0.1991
CombSUM	0.6556	0.5901	0.4983	0.4553	0.1989
Multiplication	0.6508	0.5614	0.4547	0.4117	0.1815
Element CAS	0.6196	0.5607	0.4941	0.4396	0.2000
ArtRank CAS	0.6096	0.5891	**0.5361**	**0.4629**	0.2140
CombSUM CAS	0.6038	0.5811	0.5158	0.4506	0.2044
Multiplication CAS	0.6077	0.5855	0.5328	0.4601	0.2126

- ArtRank: submitted as inex08_art_B1_loc_in_100_and_el_B1_T
- CombSUM: submittedinex08_art_B1_loc_in_100_comb_sum_el_B1_T
- Multiplication: inex08_art_B1_loc_in_100_x_el_B1_T

We also made CAS versions of these Thorough runs, using the same filtering method as last year [2]. That is, we pool all the target elements of all topics in the 2008 topic set, and filter all runs by removing any element type that is not in this pool of target elements. Our official runs for all three tasks are based on these Thorough runs. Because of the lengthy names of the runs, and to increase clarity and consistency of presentation, we denote the official runs by the methods used, instead of the official run names we used for submission.

2.3 Focused Task

To ensure the Focused run has no overlap, it is post-processed by a straightforward list-based removal strategy. We traverse the list top-down, and simply remove any element that is an ancestor or descendant of an element seen earlier in the list. For example, if the first result from an article is the article itself, we will not include any further element from this article. In the case of the CAS runs, we first apply the CAS filter and then remove overlap. Doing this the other way around, we would first remove possibly relevant target elements if some overlapping non-target elements receive a higher score.

Table 1 shows the results for the Focused Task. Somewhat surprisingly, the Article run outperforms the Element run on the official Focused measure iP[0.01], although the Element run fares much better at the earliest precision level iP[0.00]. Thus, already after 1% recall, document retrieval has a higher precision than element retrieval. A possible explanation is that, given the encyclopedic nature of the collection, for many of the Ad hoc topics there will be a Wikipedia entry that is almost entirely relevant and form more than 1% of the total relevant text. As mentioned earlier, it seems plausible that a document retrieval approach finds these pages relatively easy, and 1% recall is often achieved with the first one or two results, thus with reasonable precision. Both CombSUM and Multiplication attain higher scores for iP[0.00] than ArtRank, but the latter keeps higher precision at further recall levels. The Multiplication method loses much more precision than the other two methods. Compared to the baseline runs Article

Table 2: Results for the Ad Hoc Track Relevant in Context Task (runs in emphatic are not official submissions)

Run	gP[5]	gP[10]	gP[25]	gP[50]	MAgP
Article	0.3376	0.2807	0.2107	0.1605	0.1634
Element	0.2784	0.2407	0.1879	0.1471	0.1484
ArtRank	0.3406	0.2820	0.2120	0.1627	0.1692
CombSUM	0.3281	0.2693	0.2099	0.1615	0.1665
Multiplication	0.3295	0.2827	0.2136	0.1654	0.1695
Element CAS	0.3378	0.2837	**0.2236**	0.1719	0.1703
ArtRank CAS	0.3437	0.2897	0.2207	0.1712	0.1734
CombSUM CAS	0.3481	**0.2991**	0.2200	**0.1726**	**0.1752**
Multiplication CAS	**0.3482**	0.2888	0.2198	0.1724	0.1748

and Element, the combination methods ArtRank and CombSUM lead to substantial improvements at iP[0.01], where the Multiplication method performs slightly worse than the Article run. However, the standard Article run clearly outperforms all other runs when looking at overall precision.

Looking at the CAS runs, we see that the differences are small, with ArtRank leading to the highest iP[0.01] and MAiP scores. The CAS filtering method leads to improvements in overall precision—all MAiP scores go up compared to the non CAS variants—but has a negative effect for early precision as both iP[0.00] and iP[0.01] scores go down, except for the Multiplication run, where the iP[0.01] score goes up. Also, the CAS version of the Multiplication run does improve upon the Article run for precision up to 10% recall.

2.4 Relevant in Context Task

For the Relevant in Context task, we use the Focused runs and cluster all elements belonging to the same article together, and order the article clusters by the highest scoring element. Table 2 shows the results for the Relevant in Context Task. The Article run is better than the Element across the ranking, which is to be expected, given the results reported in [7]. It has a superior article ranking compared to the Element run, and as we saw in the previous section, it even outperformed the Element run on the official measure for the Focused task. However, this time, the combination methods ArtRank and Multiplication do better than the Article run on all reported measures, except for the Multiplication run on gP[5]. Since they use the same article ranking as the Article run, the higher precision scores of the ArtRank and Multiplication show that the elements retrieved in the Element run can improve the precision of the Article run. The CombSUM method, while not far behind, fails to improve upon the Article run on early precision levels (cutoffs 5, 10, and 25). Through the weighted combination of article and element scores, its article ranking is somewhat different from the article ranking of the Article run (and the ArtRank and Multiplication runs).

The CAS filtering method leads to further improvements. The Element CAS run outperforms the standard Article run, and the combination methods show higher precision scores than their non CAS counterparts at all rank cutoffs. This time, the CombSUM method benefits most from the CAS filter. Whereas it was well behind on per-

Table 3: Results for the Ad Hoc Track Best in Context Task (runs in emphatic are not official submissions)

Run	gP[5]	gP[10]	gP[25]	gP[50]	MAgP
Element	0.2372	0.2213	0.1778	0.1384	0.1394
Article	**0.3447**	**0.2870**	**0.2203**	**0.1681**	**0.1693**
Article offset 190	0.2462	0.2042	0.1581	0.1204	0.1228
ArtRank	0.2954	0.2495	0.1849	0.1456	0.1580
CombSUM	0.2720	0.2255	0.1872	0.1487	0.1560
Multiplication	0.2782	0.2399	0.1866	0.1496	0.1577
Element CAS	0.2758	0.2410	0.1929	0.1517	0.1487
ArtRank CAS	0.3101	0.2616	0.1952	0.1539	0.1587
CombSUM CAS	0.3081	0.2547	0.1942	0.1532	0.1581
Multiplication CAS	0.3098	0.2595	0.1944	0.1545	0.1596

formance compared to the other two combination methods, its CAS version has the highest scores for gP[10], gP[50] and MAgP. Perhaps surprisingly, the Element CAS run is even on par with the combined runs. For the Focused task, the Element CAS run scored well below the combined runs at later rank cutoffs, but when grouped by article, the differences at the later cutoff levels are very small. In fact, the Element CAS run has the highest score at gP[25]. The CAS filter could have an effect on the document ranking of the Element run.

2.5 Best in Context Task

The aim of the Best in Context task is to return a single result per article, which gives best access to the relevant elements. We experimented with three methods of selecting the best entry point:

- *Highest Scoring Element*: the highest scoring element (HSE) returned for each article. We use this on the ArtRank combined run;
- *offset 0*: the start of each returned article; and
- *offset 190*: the median distance from the start of the article of the best entry points in the 2007 assessments.

Table 3 shows the results for the Best in Context Task.

The Article run is far superior to the Element run for the Best in Context Task, at all rank cutoffs and in MAgP. In fact, the Article run outperforms all combined runs and CAS runs. The combined ArtRank run does better than the pure article run with BEPs at offset 190. Note that both these two runs have the same article ranking as the standard Article run. The highest scoring element is thus a better estimation of the BEP than the median BEP offset over a large number of topics. However, using the start of the element clearly outperforms both other runs. Of the three run combination methods, ArtRank gets better scores at early precision levels (cutoffs 5 and 10), but is overtaken by the Multiplication method at further cutoff levels. All three combinations do outperform the Element run and the article run with fixed offset of 190.

The CAS runs again improve upon their non CAS variants, showing that our filtering method is robust over tasks, retrieval approaches and combination methods. As for the

non CAS variants, ArtRank gives the best early precision, but the Multiplication gets better precision at later cutoff levels.

The combination methods consistently improve upon the Element retrieval approach, but are far behind the standard Article run. This means that our focused retrieval techniques fail to improve upon an article retrieval approach when it comes to selecting the best point to start reading a document. A closer look at the distribution of BEPs might explain the big difference between the standard Article run and the other runs. The median BEP offset for the 2008 topics is 14 and 49% of all BEPs is at the first character. This shows that choosing the start of the article will in most cases result in a much better document score than any offset further in the document.

2.6 Findings

To sum up, the combination methods seem to be effective in improving early precision. For the official Focused measure, iP[0.01], they lead to improvements over both the Article run and the Element run. The ArtRank method gives the best results for the official measure. Although the Element run scores slightly better at iP[0.00], the combination methods show a good trade off between the good overall precision of the Article run and the good early precision of the Element run. Combining them with the CAS filter improves their overall precision but hurts early precision.

For the Relevant in Context task, all three methods improve upon the Article and Element runs for MAgP. The ArtRank method shows improvement across all cutoff levels. The Multiplication method leads to the highest MAgP scores of the three methods. The CAS filter further improves their effectiveness, although the differences are small for the ArtRank method. Here, the combined runs show the best of both worlds: the good article ranking of the Article run and the more precise retrieval of relevant text within the article of the Element run.

In the Best in Context task, of the three combination methods ArtRank scores better on early precision, while the other two methods do better at later cutoff levels. However, no focused retrieval method comes close to the effectiveness of the pure Article run. With most of the BEPs at, or *very* close to, the start of the article, there seems to be little need for focused access methods for the Wikipedia collection. This result might be explained by the nature of the collection. The Wikipedia collection contains many short articles, where the entire article easily fits on a computer screen, and are all focused on very specific topics. If any text in such a short article is relevant, it usually makes sense to start reading at the beginning of the article.

Finally, the CAS filtering method shows to be robust over all tasks and focused retrieval methods used here, leading to consistent and substantial improvements upon the non CAS filtered variants.

3 Book Track

For the Book Track we investigate the effectiveness of using book level evidence for page level retrieval, and experiment with using Wikipedia as a rich resource for topical descriptions of the knowledge found in books, to mediate between user queries

and books in the INEX Book Track collection. We use Indri [13] for our retrieval experiments, with default settings for all parameters. We made one index for both book and page level, using the Krovetz stemmer, no stopword removal, and created two base runs, one at the *book* level and one at the *page* level. The INEX Book Track collection contains 50,239 out-of-copyright books. The books have on average 321 pages and just over 100,000 words. An average page has 323 words. An important difference with the Wikipedia collection, apart from document length, is the difference in structural information in the form of XML markup. In the Wikipedia articles, the markup is based on the layout, containing markup for sections, paragraphs, tables, lists, figures, etc. The books contain only minimal markup, based on the individually scanned pages. That is, there is no layer of elements about sections or chapters in between the page level and book level. Although there is information about the start of chapters and sections in the attributes of `<marker>` elements, they provide no information about where these chapters and sections end. To make use of this information for retrieval, it would require either substantial changes to our indexing approach or a pre-processing step to adjust the XML markup by introducing actual chapter and section elements.

Before we analyse the impact of book level ranking on the retrieval of individual pages, we will discuss the various book level runs we submitted for the Book Retrieval Task.

3.1 Book Retrieval Task

Koolen et al. [8] have used Wikipedia as an intermediary between search queries and books in the INEX Book collection. They experimented with using the link distance between so called *query* pages—Wikipedia pages with titles exactly matching the queries—and *book* pages—each book in the collection is associated with one or more Wikipedia pages based on document similarity—as external evidence to improve retrieval performance. We adopt this approach with the aim to investigate its effectiveness on queries that have no exact matching Wikipedia page.

We obtained the *query* pages by sending each query to the online version of Wikipedia and choosing the first returned result. If the query exactly matches a Wikipedia page, Wikipedia automatically returns that page. Otherwise, Wikipedia returns a results list, and we pick the top result. The idea is that most search topics have a dedicated page on Wikipedia. With the 70 topics of the 2008 collection, we found dedicated Wikipedia pages for 23 queries (38.6%). The *book* pages are obtained by taking the top 100 $tf.idf$ terms of each book (w.r.t. the whole collection) as a query to an Indri index of all Wikipedia pages.[1] Next, we computed the link distance between *query* pages and *book* pages by applying a random walk model on the Wikipedia link graph to obtain a measure of closeness between these pages. Books associated with Wikipedia pages closer in the link graph to the *query* page have a higher probability of being relevant [8]. We then combine these closeness scores with the retrieval scores from an Indri run.

The probability of going from node j at step s from the *query* node to node k is computed as:

$$P_{s+1|s}(k|j) = P_{s|s-1}(j) * \frac{l_{jk}}{l_j} \tag{4}$$

[1] This is based on the Wikipedia dump of 12 March, 2008.

Table 4: Results for the Book Retrieval Task (the Closeness ordered run is not an official submission

Run	MAP	P(0.0)	P(0.1)	P5	P10
$Book$	0.0899	0.4051	0.2801	0.1760	0.1320
$Book^2 * Closeness$	0.0714	0.2771	0.2230	0.1520	0.1200
$Closeness$	0.0085	0.1058	0.0406	0.0320	0.0200
Closeness ordered	0.0302	0.2163	0.0978	0.0960	0.0600

where l_{jk} is the number of links from node j to node k, l_j is the total number of links from node j and $P_{s|s-1}(j)$ is the probability of being at node j after step s. Experimentally, using the INEX 2007 Book Track data, we found that the best closeness scores for the books are obtained by simply adding the closeness scores of the top 8 Wikipedia pages retrieved for that book.

We submitted the following runs:

- $Book$: a baseline book level Indri run (submitted as 6_BST08_B_clean_trec)
- $Closeness$: a run using only the closeness scores (submitted as 6_inex08_BST_book_sim100_top8_forward_trec)
- $Book^2 * Closeness$: a combination of the baseline Indri and the closeness scores, computed as $Indri(q,b)^2 * closeness(q,b)$ for a book b and topic q (submitted as 6_BST08_B_square_times_sim100_top8_fw_trec)

Table 4 shows the results for our submitted runs based on the first release of the relevance judgements, containing judgements for 25 topics. The number of judgements per topic varies greatly. Some topics have only one or two judged books, while others have hundreds of judged books. We have to be careful in drawing conclusions from these results. The standard run performs best on all measures. The official run based on closeness scores alone performs very poorly, based on a simple error. Only 1,000 results per topic were allowed to be submitted. In generating a run from the closeness scores, the first 1,000 scores for each topic were used. However, the closeness scores were not ordered, the first 1,000 were not the highest scores. Therefore, we add the results of on unofficial run – *Closeness ordered* – based on the 1,000 highest closeness scores per topic. Although still well below the baseline run, it is clearly much better than the erroneous official run. As the baseline run is clearly the best of these runs, and the Page in Context runs submitted to the INEX 2008 Book Track are derived from this baseline, we will only use this book level run in the following section.

3.2 Page in Context

As in the Ad Hoc Track (Section 2), we experiment with methods of re-ranking the *page* level runs using the ranking of a *book* level run. Because Indri scores are always negative (the log of a probability, i.e. ranging from $-\infty$ to 0), combining scores can lead to unwanted effects (page score + book score is lower than page score alone). We therefore transform all scores back to probabilities by taking the exponents of the scores.

We experimented with the following three methods.

1. *CombSum*: add exponents of page score and book score (if the book is not retrieved, use only page score. Submitted as 6_BST08.P_plus_B.xml).
2. *Multiplication*: multiply exponents of page and book scores (if book is not retrieved, discard page. Submitted as 6_BST08.P_times_B.xml).
3. *BookRank*: retain the book ranking, replacing each book by its pages retrieved in the $Page$ run. If no pages are retrieved, use the whole book.

The official evaluation measures and results of the Page in Context are not yet released, but the relevance judgements are available. To allow a direct comparison of our methods on the Ad hoc and Book Tracks, we evaluated our Page in Context runs using the Focused and Relevant in Context measures of the Ad hoc track.

We transformed the Book Track assessments into FOL format in the following way. First, we computed the number of pages of each book and the length of the actual text, and the average page length. Giving all pages in a book this same average length, we then compute the page offsets of judged pages and retrieved pages by multiplying the average page length by $1-$ the page number. That is, a book with 500 pages and 1 million characters has an average page length of 2,000 characters. Thus, page 54 in that book has length 2,000 and starts at offset $(54-1)*2,000 = 106,000$. For the Focused task, we rank the individual pages on their scores, without grouping them per book. For the Relevant in Context evaluation, which requires results from the same document to be grouped, we use the officially submitted Page in Context runs, where the book ranking is based on the highest scoring pages of each book.

The results of the Page in Context runs evaluated using the Ad hoc measures for the Focused task (see Section 2.3) are shown in Table 5. We see that for overall precision, the $Book$ run has low precision scores compared to the Book Retrieval Task, because it is penalised for retrieving the whole books instead of only the relevant pages. However, the more focused page level runs have even lower precision scores (except for the earliest precision score of the $BookRank$ run). This somewhat surprising result can be explained by the fact that the original $Page$ run contains only a very small portion – 141 out of 6,477 – of the relevant pages. The early precision is comparable to that of the $Book$ run, but rapidly drops. The reason for the very low precision scores after 5% recall is that our runs contain up to 1,000 retrieved pages, which is not enough for most topics to reach even 5% recall.

Among the page level runs, the $BookRank$ run clearly outperform the standard $Page$ run and $CombSUM$, showing that the book level ranking helps. Boosting pages from highly ranked books leads to substantial improvements in precision across the ranking. Apart from that, retaining the whole book when no individual pages for that have been retrieved has a big impact on recall. Especially further down the results list, the $Book$ run finds relevant books that are not found by the $Page$ run. The $BookRank$ run also improves upon the $Book$ run at iP[0.00], showing that focused methods can indeed locate the relevant text within books. The big difference between the $Page$ and $BookRank$ runs, as well as the low precision of the $Page$ run by itself show that page level evidence is of limited use without the wider context of the whole book.

The results of the Page in Context runs evaluated using the Ad hoc measures for the Relevant in Context task (see Section 2.4) are shown in Table 6. We see a similar pattern as with the Focused Task results. The $Book$ run receives low scores because

Table 5: Results for the Book Track Page in Context Task (using Focused measures)

Run	iP[0.00]	iP[0.01]	iP[0.05]	iP[0.10]	MAiP
Book	0.1690	**0.1690**	**0.0999**	**0.0957**	**0.0393**
Page	0.1559	0.1002	0.0030	0.0017	0.0037
BookRank	**0.2650**	0.1618	0.0838	0.0838	0.0280
CombSUM	0.1666	0.1045	0.0095	0.0083	0.0054
Multiplication	0.0349	0.0247	0.0035	0.0030	0.0015

Table 6: Results for the Book Track Page in Context Task (using Relevant in Context measures)

Run	gP[5]	gP[10]	gP[25]	gP[50]	MAgP
Book	0.0567	0.0309	**0.0147**	**0.0087**	0.0254
Page	0.0242	0.0164	0.0098	0.0058	0.0088
BookRank	**0.0581**	**0.0315**	0.0147	0.0082	**0.0273**
CombSUM	0.0231	0.0158	0.0090	0.0064	0.0102
Multiplication	0.0061	0.0031	0.0027	0.0015	0.0047

it retrieves a lot of irrelevant text. Focused techniques should be able to achieve much better precision. Again, only the $BookRank$ run can compete with the $Book$ run, and improves upon it with early and overall precision. The fact that the $BookRank$ run has lower precision than the $Book$ run further down the ranking shows that at these lower ranks, the whole books do better than the individually retrieved pages of these books. Although this might partly be caused by the low number of pages retrieved, the low precision scores for the Focused evaluation show that the content of individual pages is not very effective for locating the relevant information in books containing hundreds of pages.

4 Discussion and Conclusions

For the *Ad Hoc Track*, we investigated the effectiveness of combining article and element retrieval methods and found that the ArtRank method, where the article run determines the article ranking, and the element run determines which part(s) of the text is returned, gives the best results for the Focused Task. For the Relevant in Context Task, the Multiplication method is slightly better than ArtRank and CombSUM, but for the CAS runs, where we filter on a pool of target elements based on the entire topic set, the CombSUM method gives the best performance overall. The combination methods are not effective for the Best in Context Task. The standard article retrieval run is far superior to any focused retrieval run. With many short articles in the collection, all focused on very specific topics, it makes sense to start reading at the beginning of the article, making it hard for focused retrieval techniques to improve upon traditional document retrieval. The CAS pool filtering method is effective for all three tasks as well, showing consistent improvement upon the non CAS variants for all measures.

For the *Book Track*, we experimented with the same run combination methods as in the Ad Hoc Track. As for the Ad hoc Track using the Wikipedia collection, we see that for the Book Track, a document retrieval approach is a non-trivial baseline.

However, for the long documents in the Book Track collection, where an individual pages forms only a small part in a much wider context, the impact of the document level ranking on focused retrieval techniques is much bigger than for the short documents in the Wikipedia collection. Using only page level evidence, the precision is very low, indicating that the content of individual pages seems not very effective in locating all the relevant text spread over multiple pages in a book. By using the ranking of the book level run, and replacing the whole content of a book only when individual pages of that book are retrieved, the combination can improve upon standard document level retrieval.

Acknowledgments Jaap Kamps was supported by the Netherlands Organization for Scientific Research (NWO, grants # 612.066.513, 639.072.601, and 640.001.501). Marijn Koolen was supported by NWO under grant # 640.001.501. # 639.072.601.

Bibliography

[1] L. Denoyer and P. Gallinari. The Wikipedia XML Corpus. *SIGIR Forum*, 40:64–69, 2006.

[2] K. N. Fachry, J. Kamps, M. Koolen, and J. Zhang. Using and detecting links in wikipedia. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 388–403. Springer Verlag, Heidelberg, 2008.

[3] N. Fuhr, J. Kamps, M. Lalmas, S. Malik, and A. Trotman. Overview of the INEX 2007 ad hoc track. In N. Fuhr, M. Lalmas, and A. Trotman, editors, *Pre-Proceedings of INEX 2007*, pages 1–22, 2007.

[4] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, Center for Telematics and Information Technology, University of Twente, 2001.

[5] ILPS. The ILPS extension of the Lucene search engine, 2008. http://ilps.science.uva.nl/Resources/.

[6] J. Kamps, M. Koolen, and B. Sigurbjörnsson. Filtering and clustering XML retrieval results. In *Comparative Evaluation of XML Information Retrieval Systems (INEX 2006)*, volume 4518 of *LNCS*, pages 121–136. Springer Verlag, Heidelberg, 2007.

[7] J. Kamps, M. Koolen, and M. Lalmas. Locating relevant text within XML documents. In *Proceedings SIGIR 2008*, pages 847–849. ACM Press, New York NY, USA, 2008.

[8] M. Koolen, G. Kazai, and N. Craswell. Wikipedia Pages as Entry Points for Book Search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM 2009)*. ACM Press, New York NY, USA, 2009.

[9] Lucene. The Lucene search engine, 2008. http://lucene.apache.org/.

[10] B. Sigurbjörnsson. *Focused Information Access using XML Element Retrieval*. SIKS dissertation series 2006-28, University of Amsterdam, 2006.

[11] B. Sigurbjörnsson, J. Kamps, and M. de Rijke. An Element-Based Approach to XML Retrieval. In *INEX 2003 Workshop Proceedings*, pages 19–26, 2004.

[12] B. Sigurbjörnsson, J. Kamps, and M. de Rijke. Mixture models, overlap, and structural hints in XML element retreival. In *Advances in XML Information Retrieval: INEX 2004*, volume 3493 of *LNCS 3493*, pages 196–210, 2005.

[13] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: a language-model based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, 2005.

[14] Wikipedia. The free encyclopedia, 2008. http://en.wikipedia.org/.