Focus and Element Length for Book and Wikipedia 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. In this paper we describe our participation in INEX 2010 in the Ad Hoc Track and the Book Track. In the Ad Hoc track we investigate the impact of propagated anchor-text on article level precision and the impact of an element length prior on the within-document precision and recall. Using the article ranking of an document level run for both document and focused retrieval techniques, we find that focused retrieval techniques clearly outperform document retrieval, especially for the Focused and Restricted Relevant in Context Tasks, which limit the amount of text than can be returned per topic and per article respectively. Somewhat surprisingly, an element length prior increases within-document precision even when we restrict the amount of retrieved text to only 1000 characters per topic. The query-independent evidence of the length prior can help locate elements with a large fraction of relevant text. For the Book Track we look at the relative impact of retrieval units based on whole books, individual pages and multiple pages.

1 Introduction

In this paper, we describe our participation in the INEX 2010 Ad Hoc and Book Tracks. Our aims for the Ad Hoc Track this year were to investigate the impact of an element length prior on the trade-off between within-document precision and recall. In previous years we merged article and element level runs—using the article ranking of the article run and the element run to select the text to retrieve in those articles—and found that this can improve performance compared to individual article and element retrieval runs. But how much text should we retrieve per article?

For the Book Track we look at the relative impact of books, individual pages, and multiple pages as units of retrieval for the Best Books and Prove It Tasks.

The rest of the paper is organised as follows. Then, in Section 2, we report our runs and results for the Ad Hoc Track. Section 3 briefly discusses our Book Track experiments. Finally, in Section 4, we discuss our findings and draw preliminary conclusions.

2 Ad Hoc Track

For the INEX 2010 Ad Hoc Track we aim to investigate:

– The effectiveness of anchor-text for focused ad hoc retrieval. Anchor-text can improve early precision in Web retrieval [10], which might be beneficial for focused retrieval in Wikipedia as well. The new Focused and Restricted Relevant in Context Tasks put large emphasis on (early) precision.
– The relation between element length and within-document precision and recall. With the new tasks restricting systems to return only a limited number of characters per article (Restricted Relevant in Context Task) or per topic (Focused Task), an element length prior might be less effective, as it increases the chances of retrieving irrelevant text.

We will first describe our indexing and retrieval approach, then the official runs, and finally per task, we present and discuss our results.

2.1 Indexing

In this section we describe the index that is used for our runs in the ad hoc track. We used Indri [16] for indexing and retrieval. Our indexing approach is based on earlier work [1, 4, 5, 13–15].

– *Section index*: We used the `<section>` element to cut up each article in sections and indexed each section as a retrievable unit. Some articles have a leading paragraph not contained in any `<section>` element. These leading paragraphs, contained in `<p>` elements are also indexed as retrievable units. The resulting index contains no overlapping elements.
– *Article index*: We also build an index containing all full-text articles (i.e., all wikipages) as is standard in IR.
– *Anchor text index*: For this index we concatenated all propagated anchor text of an article as a single anchor text representation for that article.

For all indexes, stop-words were removed, and terms were stemmed using the Krovetz stemmer. Queries are processed similar to the documents. This year we only used the CO queries for the official runs.

2.2 Category Evidence

Based on previous experiments, we used category distance scores as extra evidence for ranking [11]. We determine two target categories for a query based on the top 20 results. We select the two most frequent categories to which the top 20 results are assigned and compute a category distance score using parsimonious language models of each category.

This technique was successfully employed on the INEX 2007 Ad hoc topics by Kaptein et al. [8] and on the larger INEX 2009 collection [12] with two sets of category labels [11]; one based on the *Wikipedia* category structure and one based on the *WordNet* category labels. Koolen et al. [11] found that the labels of the original Wikipedia category structure are more effective for ad hoc retrieval. In our experiments, we use the original Wikipedia category labels.

The category distance scores are computed as follows. For each target category we estimate the distances to the categories assigned to retrieved document, similar to what is done in Vercoustre et al. [17]. The distance between two categories is estimated according to the category titles. In previous experiments we also experimented with a binary distance, and a distance between category contents, but we found the distance estimated using category titles the most efficient and at the same time effective method.

To estimate title distance, we need to calculate the probability of a term occurring in a category title. To avoid a division by zero, we smooth the probabilities of a term occurring in a category title with the background collection:

$$P(t_1, ..., t_n|C) = \sum_{i=1}^{n} \lambda P(t_i|C) + (1 - \lambda)P(t_i|D)$$

where C is the category title and D is the entire wikipedia document collection, which is used to estimate background probabilities. We estimate $P(t|C)$ with a parsimonious model [2] that uses an iterative EM algorithm as follows:

$$\text{E-step:} \qquad e_t = tf_{t,C} \cdot \frac{\alpha P(t|C)}{\alpha P(t|C) + (1 - \alpha)P(t|D)}$$

$$\text{M-step:} \qquad P(t|C) = \frac{e_t}{\sum_t e_t}, \text{i.e. normalize the model}$$

The initial probability $P(t|C)$ is estimated using maximum likelihood estimation. We use KL-divergence to calculate distances, and calculate a category score that is high when the distance is small as follows:

$$S_{cat}(C_d|C_t) = -D_{KL}(C_d|C_t) = -\sum_{t \in D} \left(P(t|C_t) * \log \left(\frac{P(t|C_t)}{P(t|C_d)} \right) \right)$$

where d is a retrieved document, C_t is a target category and C_d a category assigned to a document. The score for a document in relation to a target category $S(d|C_t)$ is the highest score, or shortest distance from any of the document's categories to the target category. So if one of the categories of the document is a target category, the distance and also the category score for that target category is 0, no matter what other categories are assigned to the document. Finally, the score for a document in relation to a query topic $S(d|QT)$ is the sum of the scores of all target categories:

$$S_{cat}(d|QT) = \sum_{C_t \in QT} \operatorname*{argmax}_{C_d \in d} S(C_d|C_t)$$

Besides the category score, we also need a query score for each document. This score is calculated using a language model with Jelinek-Mercer smoothing with a linear length prior:

$$P(q_1, ..., q_n|d) = P(d) \sum_{i=1}^{n} \lambda P(q_i|d) + (1 - \lambda)P(q_i|D)$$

where P(d) is proportional to the document length, computed as:

$$P(d) = \frac{|d|}{|D|}$$

Finally, we combine our query score and the category score through a linear combination. For our official runs both scores are calculated in the log space, and then a weighted addition is made.

$$S(d|QT) = \mu P(q|d) + (1 - \mu) S_{cat}(d|QT)$$

Based on previous experiments [8], we use $\mu = 0.9$.

2.3 Runs

Combining the methods described in the previous section with our baseline runs leads to the following official runs.

Article an article index run with length prior ($\lambda = 0.85$ and $\beta = 1$).
ArticleRF an article index run with length prior ($\lambda = 0.85$ and $\beta = 1$) and relevance feedback (top 50 terms from top 10 results).
Anchor anchor text index run without length prior ($\lambda = 0.85$ and $\beta = 0$).
AnchorLen anchor text index run with length prior ($\lambda = 0.85$ and $\beta = 1$).
Sec a section index run without length prior ($\lambda = 0.85$ and $\beta = 0$).
SecLen a section index run with length prior ($\lambda = 0.85$ and $\beta = 1$).

From these initial runs we have constructed our baseline runs:

Base the ArticleRF combined with the category scores based on the 2 most frequent categories of the top 20 results.
Base Sec the Baseline run where an article is replaced by the sections of that article retrieved by the Sec run. If no sections for that article are retrieved, the full article is used.
Fusion a linear combination of the ArticleRF and the AnchorLen runs with weight $S(d) = 0.7 ArticleRF(d) + 0.3 AnchorLen(d)$. The combined run is used to compute category scores based on the 2 most frequent categories of the top 20 results, which are then combined with the merged article and anchor text scores.
Fusion Sec the Fusion run where an article is replaced by the sections of that article retrieved by the Sec run. If no sections for that article are retrieved, the full article is used.

For the Focused Task, systems are restricted to return no more than 1000 characters per topic. We submitted two runs:

Base Sec F1000 Topic : The Base Sec run with only the first 1000 characters retrieved for each topic.
Base Sec F100 Article : The Base Sec run with only the first 100 characters retrieved per article, cut-off after 1000 characters retrieved for each topic.

With the first 1000 characters retrieved, we expect to return only very few documents per topic. With a restriction of at most N characters per document, we can control the minimum number of documents returned, thereby increasing the possible number of relevant documents returned. Both runs have the retrieved sections grouped per article, with the sections ordered according to the retrieval score of the Sec run. That is, if sections s1, s2 and s3 of document d1 are retrieved by the Sec run in the order (s2, s3, s1), then after grouping, s2 is still returned first, then s3 and then s1. The first 1000 characters retrieved will come mostly from a single document (the highest ranked document). With a limit of 100 characters per article, the first 1000 characters will come from at least 10 documents. Although precision among the first 10 documents will probably be lower than precision at rank 1, the larger number of retrieved documents might give the user access to more relevant documents. We will look at the set-based precision of the 1000 characters retrieved as well as the article-based precision and the number of retrieved and relevant retrieved articles.

For the Relevant in Context Task, systems are restricted to returning no more than 1500 results. We submitted two runs:

Base SecLen : the baseline run Base SecLen described above, cut off after the first 1500 results.

Fusion Sec : the baseline run Fusion Sec described above, cut off after the first 1500 results.

The Base and Fusion runs will allow us to see the impact of using propagated anchor-text for early precision.

For the Restricted Relevant in Context Task, systems are restricted to returning no more than 500 characters per result. We submitted two runs:

Base F500 Article : the Base run reduced to the first 500 characters per retrieved article, and cut off after the first 1500 results.

Base Sec F500 Article : the Base Sec run reduced to the first 500 characters per retrieved article, and cut off after the first 1500 results.

Article retrieval is a competitive alternative to element retrieval when it comes to focused retrieval in Wikipedia [4, 6]. The full per-article recall of article retrieval makes up for its lack in focus. However, for the Restricted Relevant in Context Task, the amount of text retrieved per article is limited to 500 characters, which reduces the high impact of full within-document recall and puts more emphasis on achieving high precision. Relevant articles tend to have relevant text near the start of the article [7], which could give fair precision with the first 500 characters of an article. On the other hand, using the more focused evidence of the section index on the same article ranking, we can select the first 500 characters of the most promising elements of the article. With a restricted number of characters per article, and therefore restricted recall, we expect to a see a clearer advantage in using focused retrieval techniques.

We discovered an error in the baseline runs, which caused our official runs to have very low scores. In the next sections, we show results for both the officially submitted runs and the corrected runs.

Table 1: Interpolated precision scores of the baseline runs (runs in italics are official submissions, runs with an asteriks are the corrected versions)

Run id	MAiP	iP[0.00]	iP[0.01]	iP[0.05]	iP[0.10]
Base	**0.2139**	0.4398	0.4219	0.3810	0.3577
Fusion	0.1823	0.4001	0.3894	0.3370	0.3189
Base Sec	0.1555	**0.5669**	**0.5130**	0.4039	0.3600
*Base SecLen	0.1702	0.5507	0.5100	**0.4162**	**0.3784**
*Fusion Sec	0.1317	0.5447	0.4632	0.3439	0.2967
Base SecLen	0.0723	0.3308	0.2910	0.2184	0.1944
Fusion Sec	0.0678	0.3027	0.2694	0.2110	0.1906

2.4 Thorough Evaluation

We first look at the performance of the baseline runs using the Thorough interpolated precision measure. Results can be found in Table 1. The Fusion run is less effective than the Base run. The anchor text does not help early precision, although we did not range over all possible weighted combinations. Perhaps a lower weight on the anchor text might be beneficial. The length prior on the sections increases recall for the cost of a slight drop in early precision. The focused runs have a lower MAiP but a higher early precision than the article level runs. The article level runs have a much higher recall, and thereby score better on average precision. But the focused runs retrieve less irrelevant text and score better on early precision.

2.5 Focused Task

We have no overlapping elements in our indexes, so no overlap filtering is done. However, systems can return no more than 1000 characters per topic. This means the result list needs to be cut-off. Articles tend to be longer than 1000 characters, so for the article-level runs, some part of an article needs to be selected to be returned. Section elements might also be longer, in which case a part of a section needs to be selected. We choose to select the first 1000 characters returned in retrieval order, based on the idea that precision tends to drop over ranks. For example, if the highest ranked section has 500 characters and the second ranked section has 600 characters, we return the highest ranked section and the first 500 characters of the second ranked section. If the highest ranked section or article is longer than 1000 characters, we select the first 1000 characters.

Because relevant text tends to start near the beginning of XML elements [3], we also looked at a method that cuts off results after 100 characters, so that the initial text from multiple high-ranked results is returned.

The mean length of Sec results is 936, although most results have far fewer (median is 397) characters. The first 1000 characters often corresponds to more than one section. The SecLen and Base results are longer, with a median length of 2,227 and 6,098 characters respectively (mean lengths are 5,763 and 13,383). For most topics, the first returned result needs to be cut-off after the first 1000 characters.

Table 2: Results for the Ad Hoc Track Focused Task (runs in italics are official submissions, runs with an asterix are the corrected versions)

Run id	**# ret.**	**# rel. ret.**	$P_{article}$	P_{char}	**iP[0.00]**	**iP[0.01]**
Base Sec F1000 Topic	1.25	0.38	0.3301	0.1232	0.1694	0.0386
Base Sec F100 Article	10.17	3.17	0.3105	0.1162	0.2468	0.0338
Sec F1000 Topic	3.94	2.13	0.5196	0.2340	0.3372	0.1087
SecLen F1000 Topic	1.56	0.90	0.5667	0.2975	0.3188	0.1261
*Base Sec F1000 Topic	1.29	0.81	**0.6250**	0.3490	0.4012	0.1376
Base SecLen F1000 Topic	1.29	0.81	0.6186	**0.3526**	0.3903	**0.1518**
Base F1000 Topic	1.10	0.69	**0.6250**	0.2806	0.2828	0.0737
Sec F100 Article	**10.63**	4.87	0.4576	0.2127	0.4117	0.0842
SecLen F100 Article	10.08	4.85	0.4804	0.1981	0.4403	0.0934
*Base Sec F100 Article	10.06	**5.27**	0.5229	0.2445	0.4626	0.1140
Base SecLen F100 Article	10.06	**5.27**	0.5229	0.2677	**0.5015**	0.1226
Base F100 Article	10.00	5.23	0.5231	0.1415	0.2623	0.0340

How does precision among the first 1000 characters correspond to older Focused evaluation measures, such as interpolated precision at the first percentage of recall (iP[0.01])? The median number of characters per topic is 129,440 (mean 339,252). For most topics, precision of the first 1000 retrieved characters is somewhere between iP[0.00] and iP[0.01].

Table 2 shows the results for the Focused Task, where $P_{article}$ is the article-level precision and P_{char} is the character-level precision. Article-level precision is based on binary relevance judgements. If the returned (part of the) article contains relevant text, the article is considered relevant at this level. For the character level precision, the set-based precision of all returned characters is computed as the number of returned highlighted characters divided by the total number of returned characters. As expected, the P_{char} for all the runs is somewhere between iP[0.00] and iP[0.01]. The first 1000 retrieved characters (denoted F1000 Topic) gives higher precision than first 100 per article up to 1000 characters (denoted F100 Article). But by restricting each article to 100 characters, many more articles, including relevant articles, are retrieved. Thus, although the set-based precision of the F100 Article runs is lower, they do give direct access to many more relevant documents. The focused runs Base Sec and Base SecLen have a higher set-based character precision than the article level Base run. The length prior on the section index has a positive impact on the precision of the first 1000 characters. The Base Sec and Base SecLen runs have the same number of retrieved articles and retrieved relevant articles, but the Base SecLen run has more relevant text in the first 1000 characters. The query-independent length prior helps locate elements with a larger proportion of relevant text.

We note that, although the F100 Article runs have a lower P_{char} than the F1000 Topic runs, they have a higher iP[0.00] score. This is because they return smaller and therefore more results. Relevant text tends to be concentrated near the start of XML elements [3], so it is often beneficial to return text from the start of an element. In the case of the F1000 Topic runs, the first retrieved result is usually 1000 characters long, so iP[0.00], which is determined after the first

Table 3: Results for the Ad Hoc Track Relevant in Context Task (runs in italics are official submissions, runs with an asteriks are the corrected versions)

Run id	**MAgP**	**gP[5]**	**gP[10]**	**gP[25]**	**gP[50]**
Base Sec Len	0.0597	0.1492	0.1330	0.1080	0.1031
Fusion Sec	0.0563	0.1207	0.1068	0.1008	0.0963
Base	0.1613	0.2900	0.2619	0.2123	0.1766
Base Sec	0.1615	0.3026	0.2657	0.2112	0.1763
*Base Sec Len	**0.1646**	**0.3149**	**0.2790**	**0.2213**	**0.1817**
Fusion	0.1344	0.2849	0.2399	0.1945	0.1547
*Fusion Sec	0.1294	0.2840	0.2427	0.1917	0.1548

Table 4: Results for the Ad Hoc Track Restricted Relevant in Context Task (runs in italics are official submissions, runs with an asteriks are the corrected versions)

Run id	**MAgP**	**gP[5]**	**gP[10]**	**gP[25]**	**gP[50]**
Base F500 Article	0.0576	0.1439	0.1191	0.1053	0.0980
Base Sec F500 Article	0.0566	0.1375	0.1199	0.1040	0.0952
*Base F500 Article	0.1358	0.2516	0.2186	0.1696	0.1473
*Base Sec F500 Article	0.1503	0.2592	0.2288	0.1887	0.1624
Base SecLen F500 Article	0.1545	0.2666	0.2368	0.1868	0.1570

relevant retrieved result, is based on the first 1000 characters of an element, be it article or section. For the F100 Article runs, iP[0.00] will often be based on a smaller number of characters. If the first 500 characters of the highest ranked section or article are relevant, taking the first 1000 characters from that section or article gives an iP[0.00] of 0.5, while taking the first 100 characters gives an iP[0.00] of 1.

2.6 Relevant in Context Task

For the Relevant in Context Task, we group results per article. Table 3 shows the results for the Relevant in Context Task. We make the following observations:

- The difference between the Base and Fusion runs is small.
- The length prior on the section index results in higher early and average precision.

2.7 Restricted Relevant in Context Task

The aim of the Restricted Relevant in Context task is to return relevant results grouped per article, with a restriction to return no more than 500 characters per article. Table 4 shows the results for the Best in Context Task. We make the following observations:

- Similar to the normal Relevant in Context task, the focused run Base Sec F500 Article has somewhat better precision than the run based on the full articles.

– A length prior over the element lengths (Base SecLen F500 Article) leads to a further improvement in precision. Thus, longer elements give higher precision in the first 500 characters.

In summary, with the restrictions on the amount of text per article and per topic that can be retrieved, focused retrieval techniques clearly outperform standard document retrieval. What is somewhat surprising is that a length prior on the section index is effective even when we use the article ranking of an article level run. The query-independent length prior helps locate elements with a large fraction and amount of relevant text.

3 Book Track

In the INEX 2010 Book Track we participated in the Best Book and Prove It tasks. Continuing our efforts of last year, we aim to find the appropriate level of granularity for focused book search. The BookML markup has XML elements on the page level. In the assessments of last year, relevant passages often cover multiple pages [9]. With larger relevant passages, query terms might be spread over multiple pages, making it hard for a page level retrieval model to assess the relevance of individual pages.

Can we better locate relevant passages by considering larger book parts as retrievable units? One simple option is to divide the whole book in sequences of n pages. Another approach would be to use the logical structure of a book to determine the retrievable units. The INEX Book corpus has no explicit XML elements for the various logical units of the books, so as a first approach we divide each book in sequences of pages.

Book index : each whole book is indexed as a retrievable unit. This index contains 50,239 books.

1-Page index : each individual page is indexed as a retrievable unit. This index contains 16,105,669 1-page units.

5-Page index : each sequence of 5 pages is indexed as a retrievable unit. That is, pages 1-5, 6-10, etc., are treated as text units. Note that a book with 53 pages is divided into 11 units: 10 units with 5 pages and 1 unit with 3 pages. Most books have more than 300 pages, so the number of units with less than 5 pages is small; less than 2% of units in the index, in fact. The average number of pages per unit is 4.97. This index contains 3,241,347 5-page units.

For the 2010 Book Track, there are 83 topics in total. Each of these topics consists of a factual statement derived from a book in the collection, as well as a query with the most important words in the factual statement. Of these 83 topics, 21 have relevance assessments for pages pooled from the official runs. For the Best Book task, the aim is to return the best books in the collection on the general topic of the statement. For the Prove It task, the aim is to return pages that either refute or confirm the factual statement. We submitted six runs in total: two for the Best Book (BB) task and four for the Prove It (PI) task.

Table 5: Page level statistics for the 2010 Prove It runs

Run	# pages	# books	# pages/book
1-page	985	464	2.1
1-page RF	1000	444	2.3
5-page	988	112	8.8
5-page RF	1000	105	9.5

Table 6: Results for the INEX 2010 Best Book Task

Run	MAP	P@5	ndcg@5	P@10	ndcg@10
Book	**0.3286**	**0.4952**	**0.4436**	**0.4429**	**0.4151**
Book-RF	0.3087	0.4667	0.3948	0.4286	0.3869

Book : a standard Book index run. Up to 100 results are returned per topic.

Book RF : a Book index run with Relevance Feedback (RF). The initial queries are expanded with 50 terms from the top 10 results.

Page : a standard Page index run.

Page RF : a Page index run with Relevance Feedback (RF). The initial queries are expanded with 50 terms from the top 10 results.

5-page : a standard 5-Page index run. Because the returned results need to be individual pages, each 5-page unit is segmented into its individual pages. The highest ranked 5-page unit is split into 5 pages with each page receiving the same score as the 5-page unit. Thus, the first N retrieved units are presented as $5N$ results.

5-Page RF : a 5-Page index run with Relevance Feedback (RF). The initial queries are expanded with 50 terms from the top 10 results.

3.1 Prove It Run Analysis

With the single page index, we get an average of 2.1 pages per book, while the 5-page index gives us 8.8 pages per book. With an average 4.97 pages per unit, that is 1.77 units per book. The 5-page index gives us less units per book than the single page index, but we get more pages from a single book. The impact of relevance feedback is a slight increase in the number of pages from the same book. Expanding the query with terms from the top ranked book pages probably results in retrieving more pages from those books because similar terms are used throughout the same book.

3.2 Best Book Results

In Table 6 we see the results for the Best Book Task. The standard Book retrieval system performs better than the system with pseudo relevance feedback. PRF expands the topic of the query somewhat, which is unnecessary for the task of finding only the best books on a topic.

Books are have graded relevance judgements, with labels *excellent*, *good*, *marginal*, *irrelevant* and *cannot judge*. The latter is used for instance when the

Fig. 1: Success rate over ranks 1 to 10 in the 2010 Best Books task

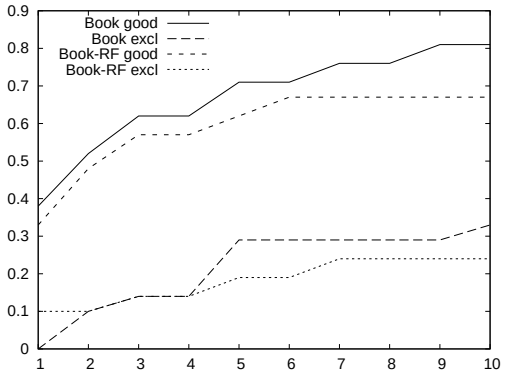


Table 7: Mapping of the Prove It relevance labels to graded relevance values

Label	Official	Extra
Irrelevant (0)	0	0
Relevant (1)	1	1
Refute (2) or Confirm (3)	2	10

scanned image of a book is unreadable. For the Best Book task we are mainly interested in books labelled *good* or *excellent*. In Figure 1 we see the success rates over the top 10 for retrieving at least one book labelled *good* or higher, and for retrieving at least one *excellent* book. We see that the Book run has a success rate for books labelled *good* or higher of almost 0.4 at rank 1 going up to 0.81 at rank 10. The Book-RF runs has a slightly lower success rate. For the books labelled *excellent*, the success rates are much lower. The Book run returns an excellent book in the top 10 for 33% of the topics, the Book-RF run for 24% of the topics.

3.3 Prove It Results

For the Prove It task, the goal is to return individual book pages that either confirm or refute a factual statement. The same set of topics is used as for the Best Book task, but here, participants could use the query field, as well as the factual statement. For our runs we only use the query field.

The assessed book pages could be assessed as either *irrelevant*, *relevant*, *refute* or *confirm*. Pages are labelled relevant when they discuss the topic of the statement, but neither refute or confirm that statement. These labels can be mapped to relevance values in multiple ways. We show results for the official mapping, and a mapping in which the refute/confirm pages are weighted extra. These mappings to graded relevance values are shown in Table 7. For the Extra mappings, the refute/confirm pages weight 10 times as much as pages labelled relevant.

Table 8: Evaluation results for the INEX 2010 Prove It task, nDCG@10 with official weights 0-1-2 (4th column) and Extra weights 0-1-10 (5th column)

Run ID	MAP	P@10	nDCG@10	
			Official	Extra
1-page	0.1216	0.3238	0.2795	**0.2338**
1-page RF	**0.1521**	**0.3524**	**0.2946**	0.2322
5-page	0.1209	0.2619	0.2182	0.1714
5-page RF	0.1163	0.2143	0.1703	0.1371

The nDCG@10 measure uses the graded relevance scores, and we regard the nDCG@10 as the official measure. The results are given in Table 8. The 1-page index gives better results than the 5-page index in terms of early precision. For the 1-page index, feedback improves precision, while for the 5-page index, feedback hurts performance. The top single page results are probably more focussed on the topic, so the expansion terms are also more related to the topic. The 5-page units might have only one or two pages on the topic, with the surrounding pages slowly drifting to other topics, which causes further topic drift through pseudo relevance feedback.

If we put extra weight on the confirm/retute pages, the nDCG@10 scores of all four runs drop. However, they drop slightly more for the 1-page RF run than for the other runs. Although feedback improves precision of the 1-page run, it apparently finds more relevant pages, but fewer confirm/refute pages. This could again be caused by topic drift introduced through pseudo relevance feedback.

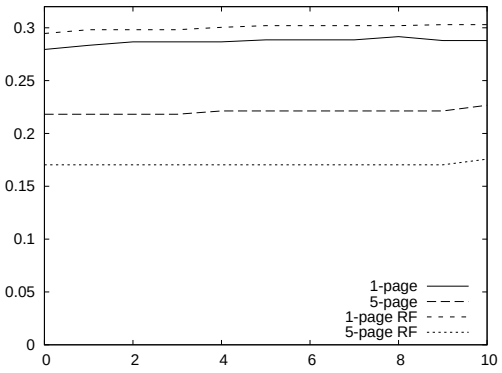
Longer units seem to provide no benefit to finding confirm/refute pages or relevant pages. At least not when treating each page within the 5-page units as individual results. Of course, one could still score a whole 5-page unit if any one of the 5 pages is relevant or confirms/refutes the factual statement. However, the relevance judgements treat each page as a separate document, so scoring larger units based on these page level judgements would give systems returning larger units an unfair advantage, as they can return more pages within the same number of ranks as systems returning only individual pages.

We have also looked at *near-misses*, that is, retrieved pages that are not relevant themselves, but are next to a relevant pages in the book. Because pages typically do not coincide with the logical of a book—section and paragraphs can start or end anywhere on the page and be split over multiple pages—a topic might be discussed over multiple pages. Although the information requested by a user might be on one page, the page preceding or succeeding it might still be closely related, and contain clues for the reader that the actual relevant information is nearby.

4 Conclusion

In this paper we discussed our participation in the INEX 2010 Ad Hoc and Book Tracks.

Fig. 2: Impact of near misses on nDCG@10 performance in the 2010 Prove It task



For the Ad Hoc Track we found that, with the restrictions on the amount of text per article and per topic that can be retrieved, focused retrieval techniques clearly outperform standard document retrieval. What is somewhat surprising is that a length prior on the section index is effective even when we use the article ranking of an article level run. The query-independent length prior helps locate elements with a large fraction and amount of relevant text.

For the Book Track, we found that pseudo relevance feedback is effective in combination with an index of individual pages as units. With larger units, feedback hurts precision. Given that the topics consist of specific factual statements, feedback only works when the top ranked results are highly focused on the topic of these statements. Although larger units might make it easier to find relevant material—early precision of the book-level runs is higher than that of the page-level runs—they also provide more off-topic text with which feedback terms introduce topic drift.

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 grants # 639.072.601 and 640.001.501.

Bibliography

[1] K. N. Fachry, J. Kamps, M. Koolen, and J. Zhang. Using and detecting links in Wikipedia. In *Focused access to XML documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2007)*, volume 4862 of *LNCS*, pages 388–403. Springer Verlag, Heidelberg, 2008.

[2] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In *Proceedings of the 27th Annual International*

ACM SIGIR Conference on Research and Development in Information Retrieval, pages 178–185. ACM Press, New York NY, 2004.

[3] J. Kamps and M. Koolen. On the relation between relevant passages and XML document structure. In A. Trotman, S. Geva, and J. Kamps, editors, *SIGIR 2007 Workshop on Focused Retrieval*, pages 28–32. University of Otago, Dunedin New Zealand, 2007.

[4] J. Kamps and M. Koolen. The impact of document level ranking on focused retrieval. In *Advances in Focused Retrieval: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2008)*, volume 5631 of *LNCS*. Springer Verlag, Berlin, Heidelberg, 2009.

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

[6] J. Kamps, M. Koolen, and M. Lalmas. Locating relevant text within XML documents. In *Proceedings of the 31th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 847–849. ACM Press, New York NY, USA, 2008.

[7] J. Kamps, S. Geva, A. Trotman, A. Woodley, and M. Koolen. Overview of the INEX 2008 ad hoc track. In S. Geva, J. Kamps, and A. Trotman, editors, *Advances in Focused Retrieval: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2008)*, volume 5631 of *LNCS*, pages 1–28. Springer Verlag, Berlin, Heidelberg, 2009.

[8] R. Kaptein, M. Koolen, and J. Kamps. Using Wikipedia categories for ad hoc search. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York NY, USA, 2009.

[9] G. Kazai, N. Milic-Frayling, and J. Costello. Towards methods for the collective gathering and quality control of relevance assessments. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 452–459, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: http://doi.acm.org/10.1145/1571941.1572019.

[10] M. Koolen and J. Kamps. The importance of anchor-text for ad hoc search revisited. In H.-H. Chen, E. N. Efthimiadis, J. Savoy, F. Crestani, and S. Marchand-Maillet, editors, *Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 122–129. ACM Press, New York NY, USA, 2010.

[11] M. Koolen, R. Kaptein, and J. Kamps. Focused search in books and Wikipedia: Categories, links and relevance feedback. In S. Geva, J. Kamps, and A. Trotman, editors, *Focused Retrieval and Evaluation: 8th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2009)*, volume 6203 of *LNCS*, pages 273–291, 2010.

[12] R. Schenkel, F. Suchanek, and G. Kasneci. Yawn: A semantically annotated wikipedia xml corpus, 2007. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.140.5501.

[13] B. Sigurbjörnsson and J. Kamps. The effect of structured queries and selective indexing on XML retrieval. In *Advances in XML Information Retrieval and Evaluation: INEX 2005*, volume 3977 of *LNCS*, pages 104–118, 2006.

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

[15] 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.

[16] 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.

[17] A.-M. Vercoustre, J. Pehcevski, and J. A. Thom. Using Wikipedia categories and links in entity ranking. In *Focused Access to XML Documents*, pages 321–335, 2007.