

# Using Language Models and the HITS Algorithm for XML Retrieval\*

Benny Kimelfeld, Eitan Kovacs, Yehoshua Sagiv, and Dan Yahav

The Selim and Rachel Benin School of Engineering and Computer Science  
The Hebrew University of Jerusalem  
Edmond J. Safra Campus, Jerusalem 91904, Israel  
{bennyk,koveitan,sagiv,dyahav}@cs.huji.ac.il

**Abstract.** Our submission to the INEX 2006 Ad-hoc retrieval track is described. We study how to utilize the Wikipedia structure (XML documents with hyperlinks) by combining XML and Web retrieval. In particular, we experiment with different combinations of language models and the HITS algorithm. An important feature of our techniques is a filtering phase that identifies the relevant part of the corpus, prior to the processing of the actual XML elements. We analyze the effect of the above techniques based on the results of our runs in INEX 2006.

## 1 Introduction

The Ad-hoc track of INEX 2006 consists of four *tasks*, namely, *Thorough*, *Focused*, *All-in-Context* and *Best-in-Context*. This paper describes our participation in this track. In particular, we describe the different methods and techniques we used and the results of our runs.

In all of the Ad-hoc tasks, the goal is to appropriately estimate the relevance of *elements* in *XML documents*. One may consider this goal as being equivalent to estimating whole documents by considering each element as a stand-alone source. We, however, take a different approach: First, we apply some preliminary ranking to the documents and *filter out* those with low relevancy. We then rank each of the elements of the documents in the remaining corpus. Our estimation, hence, consists of two main components: a *document filter* and an *element ranker*. Furthermore, each of these components is itself associated with some method of estimation. Our runs and experimentations consist of several combinations of filters and rankers. In addition, the configuration of each run contains parameter settings and some additional techniques that we describe later in this section.

In our runs, we used two methods of estimation for both document filtering and element ranking. The first method is a linear interpolation of language models [1], namely, the corpus, document and element (in the case of element ranking). The graph nature of the Wikipedia collection, which contains hyperlinks between documents and elements (in the form of *XPointers* and *XLinks*), led us to believe that combining methods of XML and Web retrieval could be

---

\* This research was supported by The Israel Science Foundation (Grant 893/05).

a promising approach. Thus, the second method is based on applying the HITS algorithm [2] and is combined with the language-model approach.

We used additional, simple techniques that were highly useful when testing our methods on the results of the Ad-hoc track of INEX 2005. One example is *length cutoff* [3], that is, elimination of very short elements from the results. Another example is the elimination of elements that have relatively high rank, yet contain none of the query term (such cases can occur due to smoothing [4] and the application of HITS).

We obtained several insights on our methods from the results of the Ad-hoc track of INEX 2006. First, our methods generally provide a good estimation of relevancy. In the Thorough and Focused tasks, most of our runs were among the best. In the other two tasks, some of our runs were among the top 10. Second, better results were obtained when using HITS for filtering rather than for element ranking. Finally, the results did not show that either one of the two filtering methods (i.e., language models and HITS) was consistently better than the other one.

## 2 Document Filters and Element Rankers

The approach that we take for estimating relevancy of XML elements entails two main steps. The first step is to identify a relevant subset of the documents in the corpus, while the second is to rank each of the elements of the documents found in the first step. More particularly, each of our submissions to the Ad-hoc track is produced by the following two components:

1. The *document filter* chooses a set of documents that are relevant to the given query. We call this set the *filtered corpus*.
2. The *element ranker* ranks each of the elements in the documents of the filtered corpus.

Note that this two-step approach is efficient since, typically, it allows us to ignore a huge number of XML elements.

To implement each of the above two components, we used two different ranking methods. Thus, a specific estimation technique in our setting is essentially a combination of two ranking methods: one for filtering out irrelevant documents and the other for ranking elements.

## 3 Document Filters

The document filters basically apply a preliminary ranking to the documents and choose the top  $N$ , where  $N$  is a predefined parameter. In all our submissions to the Ad-hoc track, we used  $N = 500$ . Next, we describe each of the two filters we used.

### 3.1 Language Model

The first filter, denoted by  $F_{LM}$ , is based on statistical language modeling [1] combined with smoothing techniques [4]. More particularly, for a query  $q$ , filtering consists of the following three steps.

1. Estimate the likelihood of generating  $q$  in each document's language model. This results in a value  $P(q|D)$  for each document  $D$ .
2. Count the number of terms of  $q$  appearing in each document  $D$ . Denote this number by  $|q \cap D|$ .
3. Choose the top  $N$  documents, when sorted *lexicographically*, first by  $|q \cap \cdot|$  and then by  $P(q|\cdot)$ .

The estimation of  $P(q|D)$  is the standard document-language model [1], smoothed by the corpus model [4]:

$$P(q|D) = \prod_{i=1}^n \lambda P(t_i|D) + (1 - \lambda)P(t_i|\mathcal{C}).$$

In this formula,  $q$  is a list of *terms*  $t_1, \dots, t_n$  and  $\mathcal{C}$  denotes the corpus. The smoothing parameter  $\lambda$  was set to 0.9 in all our submissions. We estimate the probability  $P(q|X)$  of generating an individual term  $t_i$  in the language model of  $X$  (which is either  $D$  or  $\mathcal{C}$ ) as

$$P(t_i|X) = \frac{tf(t_i, X)}{\sum_{t \in X} tf(t, X)},$$

where  $tf(t, X)$  is the total number of occurrences of  $t$  in  $X$ .

**Utilizing Structural Hints.** Our CO+S runs take the query structure (formulated in NEXI [5]) into account in the following simple way. We transform a query  $q$  into two different CO queries: The *target query*,  $q_{tgd}$ , consists of the terms that appear in the target element of  $q$ . The *supporting query*,  $q_{sup}$ , consists of the rest of the terms, i.e., all the terms that appear in  $q$ , except for those appearing in the target element. In the second filtering step described above, the evaluation of  $P(q|D)$  is replaced with a linear interpolation of the probabilities associated with the two queries  $q_{tgd}$  and  $q_{sup}$ , that is,

$$P(q|D) = \alpha P(q_{tgd}|D) + (1 - \alpha)P(q_{sup}|D).$$

Here,  $P(q_{tgd}|D)$  and  $P(q_{sup}|D)$  are evaluated regularly. In our submissions, we used  $\alpha = 0.9$ .

### 3.2 HITS

The second filter, denoted by  $F_{HITS}$ , is based on an analysis of the links (given as either XPointers or XLinks) among the Wikipedia documents using the HITS

algorithm [2].<sup>1</sup> In particular, the result of this filter consists of the top- $N$  documents w.r.t. the score obtained by applying HITS.

To describe the application of HITS, we essentially need to define the graph over which HITS is applied. So, consider a query  $q$ . The nodes of the graph are the documents in the set  $S$  that is constructed by the following three-step process:

1. Construct the set  $S_q$  of all documents  $D$ , such that a link to  $D$  contains one or more terms of  $q$ .
2. Apply the filter  $F_{LM}$  to  $s_q$  and let  $S_q^f$  be the set of the top-5 documents of  $S_q$ .
3.  $S$  includes all the documents of  $S_q^f$ , every document that points to a document of  $S_q^f$ , and every document that is pointed to by a document of  $S_q^f$ .

In the graph, there is an edge from document  $D_1$  to document  $D_2$  if  $D_1$  contains a hyperlink to  $D_2$  (i.e., either an XLink to  $D_2$  or an XPointer to an element of  $D_2$ ).

## 4 Element Rankers

The element rankers are applied to the elements of the documents in the filtered corpus in order to obtain the final rank. Again, we use two element rankers. The first ranker,  $R_{LM}$ , is based on statistical language modeling. The second ranker,  $R_{HITS}$ , combines the first ranker with the rank that the document containing the element obtains when applying HITS.

We first describe the ranker  $R_{LM}$  in detail. This ranker uses the element model, smoothed by both the document and corpus models. More formally, given a query  $q$  and an element  $E$ , we define

$$R_{LM}(E) = \prod_{i=1}^n \lambda_1 P(t_i|E) + \lambda_2 P(t_i|D) + \lambda_3 P(t_i|\mathcal{C}) \quad (1)$$

where

- $D$  and  $\mathcal{C}$  are the document that contains  $E$  and the corpus, respectively;
- $t_1, \dots, t_n$  are the terms of  $q$ ; and
- $\lambda_1, \lambda_2, \lambda_3$  are nonnegative smoothing parameters with a total sum of 1.

In our submissions to the Ad-hoc track,  $\lambda_1 = 0.8$  and  $\lambda_2 = \lambda_3 = 0.1$ .

The ranker  $R_{HITS}$  is used when the filter is  $F_{HITS}$ . Recall that when  $F_{HITS}$  is used, the HITS algorithm assigns a rank to each document  $D$  in the filtered corpus. Let that rank be denoted as  $HITS(D)$ . Then,  $R_{HITS}$  simply multiplies the rank of  $R_{LM}$  by  $HITS(D)$ , i.e.,

$$R_{HITS}(E) = R_{LM}(E) \cdot HITS(D),$$

where  $D$  is the document that contains  $E$ .

The rankers have some additional features that are described next.

<sup>1</sup> We used the JUNG library [6] in order to apply HITS.

**Table 1.** Results in the Thorough task (106 submissions)

runID	method	co/s	MAep	rnk
18_08_02	$F_{LM}/R_{LM}$	co	0.0709	1
18_11_06	$F_{LM}/R_{LM}$	cos	0.0708	2
15_11_38	$F_{HITS}/R_{LM}$	cos	0.0696	4
14_11_03	$F_{HITS}/R_{LM}$	co	0.0692	5
16_03_32	$F_{HITS}/R_{HITS}$	cos	0.0664	6
13_09_11	$F_{HITS}/R_{HITS}$	co	0.0646	8

- **CO+S.** In the case of CO+S, the probability estimation (1) uses only the terms in the target of the query (i.e.,  $q_{tgd}$ ).
- **Length cutoff.** We filter out elements that are too short. That is, we pre-defined a *cutoff* value  $c$  and ignored all the elements shorter than  $c$ .
- **Quoted expressions.** In Equation (1), we consider each quoted expression as a single term. So, for example, the query “‘West coast’ musician” consists of two terms:  $t_1 = \text{“West coast”}$  and  $t_2 = \text{“musician.”}$  So, to evaluate the term frequency (in either the corpus or the document) of a term  $t_i$ , we counted the number of consecutive appearances of the keywords of  $t_i$  (but not necessarily in the original order). Furthermore, to evaluate  $P(t_i|E)$ , we used a more flexible measure, namely, we allowed the keywords of  $t_i$  to be sufficiently “close” to each other (i.e., reside in a small interval).

## 5 Submissions and Results

In this section, we describe the runs submitted to the INEX 2006 Ad-hoc track. In each task, several combinations of filters, rankers and parameters were used. When needed, additional processing was performed so that each submission fits the specific requirements of its task.

### 5.1 Thorough Task

The Thorough task does not require elimination of element overlap. The evaluation results of our runs in this task, using the *filtered assessments*, are shown

**Table 2.** Results in *Focused* task, with overlap on (85 submissions)

runID	method	co/s	nxCG[5]		nxCG[10]		nxCG[25]		nxCG[50]	
			Score	rnk	Score	rnk	Score	rnk	Score	rnk
15_12_28	$F_{HITS}/R_{LM}$	co	0.3659	5	0.3275	3	0.2678	5	0.2257	5
16_08_52	$F_{HITS}/R_{LM}$	cos	0.3460	9	0.3103	7	0.2663	6	0.2250	6
16_12_44	$F_{HITS}/R_{HITS}$	cos	0.3244	25	0.2891	15	0.2449	11	0.2077	12
18_09_38	$F_{LM}/R_{LM}$	co	0.3547	6	0.3247	4	0.2810	1	0.2450	2
18_12_32	$F_{LM}/R_{LM}$	cos	0.3366	15	0.3103	6	0.2736	2	0.2474	1

**Table 3.** Results in *Focused* task, with overlap off (85 submissions)

runID	method	co/s	nxCG[5]		nxCG[10]		nxCG[25]		nxCG[50]	
			Score	rnk	Score	rnk	Score	rnk	Score	rnk
15_12_28	$F_{HITS}/R_{LM}$	co	0.4066	4	0.3827	2	0.3312	1	0.2770	2
16_08_52	$F_{HITS}/R_{LM}$	cos	0.3890	6	0.3697	4	0.3302	2	0.2816	1
16_12_44	$F_{HITS}/R_{HITS}$	cos	0.3999	5	0.3626	8	0.3152	5	0.2660	3
18_09_38	$F_{LM}/R_{LM}$	co	0.3878	7	0.3670	5	0.3163	4	0.2620	5
18_12_32	$F_{LM}/R_{LM}$	cos	0.3684	12	0.3506	9	0.3081	7	0.2639	4

in Table 1. Later, we consider the results using the non-filtered assessments. For each run, the table specifies the run identifier (**runID**), the filter-ranker combination (**method**), whether the run is CO or CO+S (**co/s**) and the *rank* of the result (**rnk**), i.e., its position among the submissions of all the participants in this task. For example, the first line describes Run 18.08.02 with the following properties. The filter is  $F_{LM}$ , the ranker is  $R_{LM}$  and the query was considered as content-only (co). The MAep score of this run is 0.0709 and it is the best run in the Thorough task. Our runs on this task use the same parameters. In particular, the length cutoff is 20.

Table 1 shows that, in general, our submissions provide a good tradeoff of effort vs. recall-gain (compared to the other submissions). Furthermore, under this yardstick, the use of HITS does not improve our runs, that is, it is best to use the language-model approach for both filtering and ranking.

The ranks of the results of our runs among those using the non-filtered assessments are very similar to those described above (using the filtered assessments). A remarkable difference is the following. In the non-filtered results, Run 15.11.38 (forth in the filtered assessments) jumped to the first place, pushing Runs 18.03.02 and 18.11.06 to the second and third places, respectively.

## 5.2 Focused Task

In the Focused task, overlapping elements were eliminated as follows. For each document  $D$  in the filtered corpus, we listed all the elements of  $D$  in descending

**Table 4.** Results in *BestInContext* task, Metric BEPD (77 submissions)

runID	method	co/s	A=0.01	At A=0.1	At A=1.0	At A=10.0	At A=100.0					
			Score	rnk	Score	rnk	Score	rnk	Score	rnk		
19_08_40	$F_{LM}/R_{LM}$	cos	0.1604	8	0.2329	7	0.3502	8	0.5437	8	0.7451	10
20_12_09	$F_{LM}/R_{LM}$	cos	0.1441	15	0.2166	16	0.3365	18	0.5348	15	0.7430	11
19_03_22	$F_{LM}/R_{LM}$	co	0.1610	6	0.2334	6	0.3493	9	0.5404	9	0.7374	13
19_06_40	$F_{LM}/R_{LM}$	co	0.1469	11	0.2190	15	0.3374	17	0.5327	16	0.7357	17
17_11_09	$F_{HITS}/R_{HITS}$	co	0.1127	33	0.1644	40	0.2474	48	0.3946	48	0.5548	50
17_09_18	$F_{HITS}/R_{HITS}$	cos	0.1097	34	0.1619	41	0.2458	49	0.3886	49	0.5448	53

**Table 5.** Results in *BestInContext* task, Metric EPRUM-BEP-Exh-BEPDistance (77 submissions)

runID	method	co/s	A=0.01	At A=0.1	At A=1.0	At A=10.0	At A=100.0
			Score rnk	Score rnk	Score rnk	Score rnk	Score rnk
20_12_09	$F_{LM}/R_{LM}$	cos	0.0251 21	0.0428 31	0.0752 30	0.1421 30	0.2320 25
19_08_40	$F_{LM}/R_{LM}$	cos	0.0292 7	0.0486 13	0.0809 21	0.1460 22	0.2325 23
19_03_22	$F_{LM}/R_{LM}$	co	0.0286 9	0.0477 16	0.0801 22	0.1481 18	0.2372 18
19_06_40	$F_{LM}/R_{LM}$	co	0.0258 19	0.0433 30	0.0758 29	0.1460 22	0.2367 19
17_11_09	$F_{HITS}/R_{HITS}$	co	0.0239 22	0.0423 32	0.0741 37	0.1469 21	0.2515 16
17_09_18	$F_{HITS}/R_{HITS}$	cos	0.0258 16	0.0441 26	0.0768 25	0.1489 15	0.2528 15

rank. We then traversed the list (in the order of descending rank) and removed every element that overlapped with any previous element.

Tables 2 and 3 show the results in the Focused task, with overlap on and off, respectively. These results correspond to the filtered assessments. (We later consider the non-filtered ones). All the runs use the same parameters (the length cutoff is 20). Note that the specified ranks (denoted **rnk** in the tables) are the *effective* ranks of our runs, i.e., with the invalid submissions excluded.

Consider Table 2, with overlap on. In all the runs, we used  $R_{LM}$  for element ranking. The first run (using  $F_{HITS}$  as a filter) yields the best scores among our runs for nxCG[5] and nxCG[10]. However, for nxCG[25] and nxCG[50], the fourth and fifth runs (that use the filter  $F_{LM}$ ) are better and, in fact, the best among all submissions. A different behavior is exposed in Table 3 (overlap off). There, the first run is superior to the others under all the nxCG metrics, except for nxCG[50] where it is second (and the second run is the best).

In the non-filtered assessments, our runs got almost identical ranks, except for a few minor differences. The most significant difference is that Run 18\_12\_32, ranked 6 in the overlap-on task under the metric nxCG[10], is only ranked 9 in the corresponding list of non-filtered results.

### 5.3 Best-in-Context and All-in-Context Tasks

In the Best-in-Context task, the element with the highest score was chosen for each document. The results in this task, under the metrics BEPD and EPRUM-BEP-Exh-BEPDistance, are shown in Tables 4 and 5, respectively. The length cutoff was 30 in runs 19\_06\_40 and 20\_12\_09. In the other runs, it was 20.

In the All-in-Context task, overlap was eliminated similarly to the Focused task. Table 6 shows the results in the All-In-Context task. The length cutoff was 10 in runs 19\_01\_56 and 18\_11\_02; in the other runs, it was 20.

Compared to other submissions, our methods obtained better results in the first two tasks than in the third and fourth. Note, however, that in the Best-in-Context task, Run 19\_08\_40 (Table 4) was among the top 10 under the BEPD metric for all values of A. Furthermore, in the All-in-Context task, the first four runs of Table 6 are always among the top 13 and they are significantly better than the fifth and sixth.

**Table 6.** Results in *AllInContext* task (56 submissions)

runID	method	co/s	MAgP		gP[5]		gP[10]		gP[25]		gP[50]	
			Score	rnk	Score	rnk	Score	rnk	Score	rnk	Score	rnk
18_11_02	$F_{LM}/R_{LM}$	co	0.1601	7	0.3176	13	0.2585	13	0.1956	9	0.1446	8
18_09_30	$F_{LM}/R_{LM}$	co	0.1599	8	0.3198	12	0.2603	11	0.1957	8	0.1440	9
19_12_30	$F_{LM}/R_{LM}$	cos	0.1584	9	0.3303	7	0.2631	9	0.1927	11	0.1411	12
19_01_56	$F_{LM}/R_{LM}$	cos	0.1584	10	0.3262	10	0.2600	12	0.1927	10	0.1418	10
17_03_42	$F_{HITS}/R_{HITS}$	cos	0.0353	52	0.0925	52	0.0788	52	0.0625	52	0.0441	53
17_02_23	$F_{HITS}/R_{HITS}$	co	0.0348	53	0.0886	53	0.0781	53	0.0610	53	0.0445	52

## 6 Conclusion

In our participation in the INEX 2006 Ad-hoc retrieval track, we mainly studied two retrieval techniques. The first is a preliminary filtering of the corpus in order to obtain the documents from which the actual elements are considered. The second is the use of HITS for either filtering of documents or ranking of elements. The results of our submissions show that preliminary filtering improves the quality of retrieval, since our runs were among the best in the Thorough and Focused tasks. Furthermore, the use of HITS is useful for appropriately identifying the few top elements. In comparison, language models generally yielded better results in identifying large collections of relevant elements. In future work, we plan to further study the integration of the techniques presented in this paper in order to achieve the best of both worlds. In particular, we will study the use of PageRank [7], instead of HITS, for link analysis.

## References

1. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: SIGIR, pp. 275–281. ACM Press, New York (1998)
2. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: SODA, pp. 668–677. ACM Press, New York (1998)
3. Kamps, J., de Rijke, M., Sigurbjörnsson, B.: Length normalization in XML retrieval. In: SIGIR, pp. 80–87. ACM Press, New York (2004)
4. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: SIGIR, New York, NY, USA, pp. 334–342. ACM Press, New York (2001)
5. Trotman, A., Sigurbjörnsson, B.: Narrowed extended XPath I (NEXI). In: Fuhr, N., Lalmas, M., Malik, S., Szilávik, Z. (eds.) INEX 2004. LNCS, vol. 3493, pp. 16–40. Springer, Heidelberg (2005)
6. The JUNG Framework Development Team: JUNG java universal network/graph framework (2006) <http://jung.sourceforge.net>
7. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. *Computer Networks* 30(1-7), 107–117 (1998)