The University of Amsterdam at INEX 2008: Ad Hoc, Book, Entity Ranking, Interactive, Link the Wiki, and XML Mining Tracks

Khairun Nisa Fachry[1], Jaap Kamps[1,2], Rianne Kaptein[1], Marijn Koolen[1], and Junte Zhang[1]

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

Abstract. This paper documents our experiments in six of the INEX 2008 tracks: Ad Hoc, Book Search, Entity Ranking, Interactive, Link the Wiki, and XML Mining. We discuss our aims, experiments, results and findings for each of these tracks.

1 Introduction

In this paper, we describe our participation in six of the INEX 2008 tracks.

For the *Ad Hoc Track*, we explore the good performance of standard document retrieval systems in term of their superior document ranking when compared to element retrieval approaches. Our aim is to investigate the relative effectiveness of both approaches. We experiment with combining the two approaches to get the best of both worlds.

For the *Book Track*, our aims were two-fold: 1) to investigate the effectiveness of using book level evidence for page level retrieval, and 2) 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.

For the *Entity Ranking Track*, our aim was to explore the relations and dependencies between Wikipedia pages, categories and links.

For the *Interactive Track*, we particated in the concerted experiment designed by the organizers. The specific aims of the experiment were to investigate the impact of the task context for two types of simulated tasks were defined that are believed to represent typical information needs of Wikipedia-users: fact-finding tasks and research tasks.

For the *Link the Wiki Track*, our aim was to investigate a two-tier approach to link detection: first, a relevant pool of foster (candidate) articles is collected; second, substring matching with the list of collected titles to establish an actual link. We specifically look at the effectiveness of the two-tier approach on early precision and on recall.

For the *XML Mining Track*, our aim was to explore whether we can use link information to improve classification accuracy.

The document collection for all tracks except the Book Search track is based on the English Wikipedia [31]. The collection has been converted from the wiki-syntax to an XML format [4]. 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.

The rest of the paper describes our experiments for each of the tracks in a relatively self-contained sections. First, in Section 2, we report the results for the Ad Hoc Track. Then Section 3 presents our retrieval approach in the Book Track. Followed by our results for the Entity Ranking Track in Section 4. Then, in Section 5, we discuss our Interactive Track experiments. Followed by Section 6 detailing our approach and results for the Link the Wiki Track. And last but not least, in Section 7, we discuss our XML Mining Track experiments. Finally, in Section 8, we discuss our findings and draw some conclusions.

2 Ad Hoc Track

For the INEX 2008 Ad Hoc Track we aim to 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.

2.1 Retrieval Model and Indexing

Our retrieval system is based on the Lucene engine with a number of home-grown extensions [11, 20].

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 link evidence and other query independent evidence [9, 19].

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 [27]. For comprehensive experiments on the earlier INEX data, see [24].

Our indexing approach is based on our earlier work [6, 15, 16, 25, 26, 27].

- *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 [26].
- *Article index*: We also build an index containing all full-text articles (i.e., all wikipages) 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. [17], article retrieval leads to a better document ranking, whereas element retrieval fares better at retrieving relevant text within documents. 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:

- `inex08_art_B1_loc_in_100_and_el_B1_T`: using the ArtRank method to combine article and element runs;
- `inex08_art_B1_loc_in_100_comb_sum_el_B1_T`: using the CombSUM method to combine article and element runs; and
- `inex08_art_B1_loc_in_100_x_el_B1_T`: using Multiplication to combine article and element runs.

Table 1: Results for the Ad Hoc Track Focused Task (runs in emphatic are no 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

We also made CAS versions of these Thorough runs, using the same filtering method as last year [6]. 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.

All 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 all 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. These high scoring non-target elements are then removed in the CAS filtering step, and we would lose many more promising elements than if we apply the CAS filter first.

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]. 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 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

Table 2: Results for the Ad Hoc Track Relevant in Context Task (runs in emphatic are no 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

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 board, which is to be expected, given the results reported in [17]. 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 on all cutoff levels. This time, the CombSUM method benefits most from the CAS filter. Whereas it was well behind on performance 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.

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.

Table 3: Results for the Ad Hoc Track Best in Context Task (runs in emphatic are no 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

We experimented with three methods of selecting the best entry point:

- *Highest Scoring Element*: simply 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, on all cutoff levels 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 substantial 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 start 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 our aims were two-fold: 1) to investigate the effectiveness of using book level evidence for page level retrieval, and 2) 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 [28] 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.

3.1 Book Retrieval Task

Koolen et al. [18] 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 1) on queries that have no exact matching Wikipedia page and 2) in the context of focussed retrieval, namely, the Page in Context task.

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 [18]. 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:

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

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.

To combine content based retrieval scores with external evidence, Craswell et al. [3] guessed transformation functions from looking at distributions of log odds estimates for different features. URL length, link indegree and click distance (the minimum of clicks to the page from a root page) were modelled by sigmoid functions, leading to substantial improvements when combined with a BM25 baseline. We choose a standard sigmoid function for the transformation:

$$sigmoid(b, q) = \frac{1}{1 + e^{-cl(b,q)}} \tag{5}$$

where $cl(b, q)$ is the closeness score for book b and query q. The sigmoid function ensures that at the low end of the distribution, where there is no relation to relevance, the closeness scores are transformed to values very close to 0.5 (a closeness score of zero would be transformed to 0.5). Close to 1, the closeness scores rapidly increase to 0.73. Thus, only the books at the high end of the distribution receive a boost. We combine this with Indri's retrieval score by simple addition:

$$S(b, q) = Indri(b, q) + sigmoid(b, q) \tag{6}$$

The INEX 2007 Book Track test collection contains only judgements on the book level, with shallow pools – an average of 15.7 books judged for each of the 250 topics. Although there are only 70 topics in the 2008 topic set, the test collection should contain

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

more judgements per topic and even judgements on the page level. This will allow for a much more detailed analysis of the effectiveness of using Wikipedia as an intermediary between search queries and books.

3.2 Page in Context

As in the Ad Hoc Track, we experiment with methods of re-ranking the *page* level runs using *book* level evidence. 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. *Plus*: add exponents of page score and book score (if the book is not retrieved, use only page score).
2. *Multiplication*: multiply exponents of page and book scores (if book is not retrieved, discard page).
3. *BookRank*: use book score for all retrieved pages.

The 2008 Book Track assessment still have to take place at the time of writing, so we cannot give any results on our submitted runs.

4 Entity Ranking

In the entity ranking track, our aim is to explore the relations and dependencies between Wikipedia pages, categories and links. For the entity ranking task we have looked at some approaches that proved to be successful in last year's entity ranking and ad hoc track. In these tracks it has been shown that link information can be useful. Kamps and Koolen [14] use link evidence as document priors, where a weighted combination of the number of incoming links from the entire collection and the number of incoming links from the retrieved results for one topic is used. Tsikrika et al. [29] use random walks to model multi-step relevance propagation from entities to their linked entities. For the entity ranking track specifically also the category assignments of entities can be used. Vercoustre et al. [30] use the Wikipedia categories by defining similarity functions between the categories of retrieved entities and the target categories. The similarity scores are estimated using lexical similarity of category names. We implemented, extended and combined the aforementioned approaches..

4.1 Model

We would like to create a model that exploits both link and category information and try to find a natural way of combining these different sources of information.

Category information Although for each topic one or a few target categories are provided, relevant entities are not necessarily associated with these provided target categories. Relevant entities can also be associated with descendants of the target category or other similar categories. Therefore, simply filtering on the target categories is not sufficient. Also, since Wikipedia pages are usually assigned to multiple categories, not all categories of an answer entity will be similar to the target category. We calculate for each target category the distances to the categories assigned to the answer entity. To calculate the distance between two categories, we tried three options. The first option (binary distance) is a very simple method: the distance is 0 if two categories are the same, and 1 otherwise. The second option (contents distance) calculates distances according to the contents of each category, the third and option (title distance) calculates a distance according to the category titles. For the title and contents distance, we need to calculate the probability of a term occurring in a category. To avoid a division by zero, we smooth the probabilities of a term occurring in a category with the background collection:

$$P(t_1, ..., t_n|C) = \sum_{i=1}^{n} \lambda P(t_i|C) + (1-\lambda)P(t_i|D) \tag{7}$$

where C, the category, consists either of the category title to calculate title distance, or of the concatenated text of all pages belonging to that category to calculate contents distance. D is the entire wikipedia document collection, which is used to estimate background probabilities.

We estimate $P(t|C)$ using a parsimonious model [10] that use an iterative EM algorithm as follows:

$$\begin{aligned} \text{E-step:} \qquad e_t &= tf_{t,C} \cdot \frac{\alpha P(t|C)}{\alpha P(t|C) + (1-\alpha)P(t|D)} \\ \text{M-step:} \quad P(t|C) &= \frac{e_t}{\sum_t e_t}, \text{i.e. normalize the model} \end{aligned} \tag{8}$$

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:

$$S_{cat}(C_d|C_t) = -D_{KL}(C_d|C_t) = -\sum_{t \in D} \left(P(t|C_t) * \log \left(\frac{P(t|C_t)}{P(t|C_d)} \right) \right) \tag{9}$$

where d is a document, i.e. an answer entity, C_t is a target category and C_d a category assigned to a document . The score for an answer entity in relation to a target category ($S(d|C_t)$) is the highest score, corresponding to the smallest distance, from the scores $S(C_d|C_t)$, the scores for the distances from the categories of the document to the target category.

In contrast to [30], where a ratio of common categories between the categories associated with an answer entity and the provided target categories is calculated, we take for each target category only the minimal distance of the distances from the answer entity categories to a target category. So if one of the categories of the document is exactly the 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 an answer entity in relation to a query topic ($S(d|QT)$) is the sum of the scores of all target categories:

$$S_{cat}(d|QT) = \sum_{C_t \in QT} \underset{C_d \in d}{\operatorname{argmax}} S(C_d|C_t) \tag{10}$$

List Completion The second task in the entity ranking track is list completion. Instead of the target category, for each topic a few relevant examples entities are given. We treat all categories assigned to the example entities as target categories. Our approach for using the link and the category information is the same as before. But to get the final score of an article in relation to a topic, we use two variants. The first one is:

$$S_{Sum}(d|QT) = \sum_{ex \in QT} \sum_{C_{ex} \in ex} \underset{C_d \in d}{\operatorname{argmax}} S_{cat}(C_d|C_{ex}) \tag{11}$$

In the second variant $S_{Max}(d|QT)$, instead of summing the score of each example category, we only take the maximum score i.e. shortest distance for all example categories of the entity examples to one of the categories of the document. Furthermore, we use the example entities for explicit relevance feedback through query expansion.

Link Information We implement two options to use the link information: relevance propagation and document link degree prior. For the document link degree prior we use the same approach as in [14]. The prior for a document d is:

$$P_{Link}(d) = 1 + \frac{Indegree_{Local}(d)}{1 + Indegree_{Global}(d)} \tag{12}$$

The local indegree is equal to the number of incoming links from within the top ranked documents retrieved for one topic. The global indegree is equal to the number of incoming links from the entire collection.

The second use of link information is through relevance propagation from initially retrieved entities, as was done last year in the entity ranking track by Tsikrika et al. [29].

$$P_0(d) = P(q|d) \tag{13}$$

$$P_i(d) = P(q|d)P_{i-1}(d) + \sum_{d' \to d} (1 - P(q|d'))P(d|d')P_{i-1}(d') \tag{14}$$

Probabilities $P(d|d')$ are uniformly distributed among all outgoing links from the document. Documents are ranked using a weighted sum of probabilities at different steps:

$$P(d) = \mu_0 P_0(d) + (1 - \mu_0) \sum_{i=1}^{K} \mu_i P_i(d) \tag{15}$$

For K we take a value of 3, which was found to be the optimal value last year. We try different values of μ_0 and distribute $\mu_1 ... \mu_K$ uniformly.

Table 4: Document link degree prior results

# docs	μ	MAP	P10
Baseline		0.1840	0.1920
50	0.6	0.1898 ˉ	**0.2040** ˉ
50	0.5	0.1876 ˉ	0.2000 ˉ
100	0.7	0.1747 ˉ	0.2000 ˉ
100	0.3	0.1909 ˉ	0.1920 ˉ
500	0.5	**0.1982**°	0.2000 ˉ
500	0.3	0.1915 ˉ	**0.2040**°
1,000	0.5	0.1965 ˉ	0.1960 ˉ
1,000	0.4	0.1965°	0.2000 ˉ

Table 5: Category distances results

Dist.	Weight	MAP	P10
Binary	0.1	0.2145 ˉ	0.1880 ˉ
Cont.	0.1	0.2481◉	0.2320°
Title	0.1	0.2509°	0.2360°
Cont.	0.05	**0.2618**◉	**0.2480**◉
Title	0.05		

Note: *Significance of increase over baseline according to t-test, one-tailed, at significance levels 0.05 (°), 0.01 (◉), and 0.001 (•).*

Combining Information Finally, we have to combine our different sources of information. We start with our baseline model which is a standard language model. We have two possibilities to combine information. We can make a linear combination of the probabilities and category score. All scores and probabilities are calculated in the log space, and then a weighted addition is made. Secondly, we can use a two step model. Relevance propagation takes as input initial probabilities. Instead of the baseline probability, we can use the scores of the run that combines the baseline score with the category information. Similarly, for the link degree prior we can use the top results of the baseline combined with the category information instead of the baseline ranking.

4.2 Experiments

For our training data we use the 25 genuine entity ranking test topics that were developed for the entity ranking track last year. Since no results are available on the test data yet, we only report on our results on the training data.

For our baseline run and to get initial probabilities we use the language modeling approach with Jelinek-Mercer smoothing, Porter stemming and pseudo relevance feedback as implemented in Indri [28] to estimate $P(d|q)$. We tried different values for the smoothing λ, and $\lambda = 0.1$ gives the best results, with a MAP of 0.1840 and a P10 of 0.1920. For the document link degree prior we have to set two parameters: μ, that determines the weight of the document prior. For μ we try all values from 0 to 1 with steps of 0.1. The weight of the baseline is here $(1 - \mu)$. Only values of μ that give the best MAP and P10 are shown in Table 4. We have to decide how many documents are used to calculate the document prior, and look at 50, 100, 500 and 1,000 documents.

The results of using category information are summarized in Table 5. The weight of the baseline score is 1.0 minus the weight of the category information. For all three distances, a weight of 0.1 gives the best results. In addition to these combinations, we also made a run that combines the original score, the contents distance and the title distance. When a single distance is used, the title distance gives the best results. The combination of contents and title distance gives the best results overall.

In our final experiments we try to combine all information we have, the baseline score, the category and the link information. Firstly, we combine all scores by making

Table 6: Results linear combination

Link Info	Weight	MAP	P10
Prior	1.0	$0.2564^{\circledcirc}$	$0.2680^{\circledcirc}$
Prior	0.5	$0.2620^{\circledcirc}$	$0.2560^{\circledcirc}$
Prop.	0.1	**$0.2777^{\circledcirc}$**	**$0.2720^{\circledcirc}$**

Table 7: Results two step model

Link info	Weight	MAP	P10
Prior	0.5	$0.2526^{\circledcirc}$	$0.2600^{\circledcirc}$
Prop.	0.2	$0.2588^{\circledcirc}$	**$0.2960^{\bullet}$**
Prop.	0.1	**$0.2767^{\circledcirc}$**	$0.2720^{\circledcirc}$

Table 8: Feedback results

RF	PRF	MAP	P10
No	No	0.1409	0.1240
Yes	No	**0.1611**	0.1600
Yes	Yes	0.1341	**0.1960**

Table 9: List Completion results

Dist.	Weight	$S(A\|QT)$	C_t	MAP	P10
	Baseline LC			0.1611	0.1600
Cont.	0.1	Sum	No	$0.2385^{\circledcirc}$	0.2520°
Cont.	0.9	Sum	Yes	$0.2467^{\bullet}$	0.2560°
Cont.	0.2	Max	No	0.1845^{-}	0.2360^{-}
Title	0.1	Sum	No	$0.2524^{\circledcirc}$	0.2640°
Title	0.9	Sum	Yes	**$0.2641^{\bullet}$**	**0.2760°**
Title	0.5	Max	No	0.1618^{-}	0.2080^{-}
Cont. Title	0.05 0.05	Sum	No	$0.2528^{\bullet}$	0.2640°

a linear combination of the scores and probabilities (shown in Table 6). The best run using category information (weight contents = 0.05, and weight title = 0.05) is used in combination with the link information. Secondly, we combine the different sources of information by using the two step model (see Table 7). Link information is mostly useful to improve early precision, depending on the desired results we can tune the parameters to get optimal P10, or optimal MAP. Relevance propagation performs better than the document link degree prior in both combinations.

For the list completion task, we can also use the examples for relevance feedback. To evaluate the list completion results, we remove the example entities from our ranking. Applying explicit and pseudo relevance feedback leads to the results given in Table 8. Additional pseudo relevance feedback after the explicit feedback, only improves early precision, and harms MAP. We take the run using only relevance feedback as our baseline for the list completion task.

When we look at the previous entity ranking task, the largest part of the improvement comes from using category information. So here we only experiment with using the category information, and not the link information. We have again the different category representations, content and category titles. Another variable here is how we combine the scores, either add up all the category scores ($S_{Sum}(A|QT)$) or taking only the maximum score ($S_{Max}(A|QT)$). Not part of the official task, we also make some runs that use not only the categories of the example entities, but also the target category(ies) provided with the query topic. In Table 9 we summarize some of the best results. The combination of contents and title distance, does not lead to an improvement over using only the title distance. The maximum score does not perform as well as the summed scores. We use all categories assigned to the entity examples as target categories, but some of these categories will not be relevant to the query topic introducing noise in the target categories. When the scores are summed, this noise is leveled out, but when only the maximum score is used it can be harmful.

4.3 Findings

We have presented our entity ranking approach where we use category and link information. Category information is the factor that proves to be most useful and we can do more than simply filtering on the target categories. Category information can both be extracted from the category titles and from the contents of the category. Link information can also be used to improve results, especially early precision, but these improvements are small.

5 Interactive Experiments

In this section, we discuss our participation in the INEX 2008 Interactive Track (iTrack). For the Interactive Track, we participated in the concerted experiment designed by the organizers. The overall aim of the iTrack is to understand the how users interact with structured XML documents; the specific aims of the INEX 2008 Interactive Track were to investigate the impact of the task context. Two types of simulated tasks were defined that are believed to represent typical information needs of Wikipedia-users: fact-finding tasks and research tasks. The track is set up to investigate whether the different task types lead to different information interaction: for example, a test-person may prefer different levels of granularity to view documents, different numbers of results, etc. In addition, other factors that may impact the test person's information seeking behavior and satisfaction, such as her knowledge of and interest in the task at hand, are also recorded

As in previous years, the Daffodil system is used for element retrieval. The system returns elements of varying granularity based on the hierarchical document structure. All elements coming from the same document are grouped together in the hitlist (as in the Ad Hoc Track's Relevant in Context task). Any result in the hitlist is clickable and shows the corresponding part of the document in the result display. In addition, all elements matching the query are shown with background highlighting in the result display. For further information about the track set-up we refer to [22].

5.1 Approach

We recruited eight test persons in which seven of them completed the experiment. One test person failed to complete the experiment due to system failure. Hence, our results are based on the data from seven test persons. The organizers provided six simulated search tasks corresponding to two different task types. The first type is *fact-finding*: search tasks that request specific information for a topic. An example fact-finding task is:

> *As a frequent traveler and visitor of many airports around the world you are keen on finding out which is the largest airport. You also want to know the criteria used for defining large airports.*

The second type is *research*: search tasks that required broad information of a topic and the information can only be found by collecting information from several documents. An example research task is:

Table 10: Post-task questionnaire, with answers on a 5-point scale (1-5).

Q3.1: How understandable was the task?
Q3.2: How easy was the task?
Q3.3: To what extent did you find the task similar to other searching tasks that you typically perform?
Q3.6: Are you satisfied with your search results?
Q3.7: How relevant was the information you found?
Q3.8: Did you have enough time to do an effective search?
Q3.9: How certain are you that you completed the task?

Table 11: Post-task responses on searching experience: mean scores and standard deviations (in brackets)

Type	Q3.1	Q3.2	Q3.3	Q3.6	Q3.7	Q3.8	Q3.9
All tasks	4.50 (0.65)	3.29 (1.38)	3.71 (1.27)	3.14 (1.61)	3.29 (1.59)	3.14 (1.35)	2.86 (1.41)
Fact Finding	4.71 (0.49)	3.00 (1.91)	3.43 (1.72)	2.43 (1.81)	2.43 (1.81)	2.86 (1.77)	2.57 (1.81)
Research	4.29 (0.76)	3.57 (0.53)	4.00 (0.58)	3.86 (1.07)	4.14 (0.69)	3.43 (0.79)	3.14 (0.90)

> *You are writing a term paper about political processes in the United States and Europe, and want to focus on the differences in the presidential elections of France and the United States. Find material that describes the procedure of selecting the candidates for presidential elections in the two countries.*

Each test person chose and worked with one simulated task from each category.

5.2 Results

We provide an initial analysis of the data gathered through questionnaires and log files. We will limit our attention here to the post-task questionnaire (Q3, selected questions are shown in Table 10) and the corresponding log data.

The responses of the test persons are summarized in Table 11. If we look at the responses over all tasks the average response varies from 2.86 to 4.50 signaling that the test persons were reasonably positive. We also look at the responses for each task type. Here we see that test persons understood both tasks very well (Q3.1). Fact finding receives higher responses on average, which makes sense given the nature of the simulated tasks and thereby confirms that the chosen simulated tasks represent the particular task types. For all other questions in Table 11, the research task type is receiving higher responses on average. That is, the research task was regarded easier (Q3.2) and more similar to other searching task (Q3.3) compare to the fact finding task. Admittedly, this may be a result of our choice of test persons who all have an academic education. Moreover, test persons were more satisfied with the search results provided by the system (Q3.6) for the research task. A possible explanation is that the research tasks are more open-ended than the fact finding task where test persons need to find a specific and precise answer. Hence, additional material provided by the system may be more useful in the research task context. This explanation is supported by the response when asked about the relevancy of the found information (Q3.7). Test persons believed that they found more relevant results for the research task.

Next, we look at the time test persons spent on each task. On the question whether there was enough time for an effective search (Q3.8), responses for the fact finding tasks were lower than for the research tasks. This is rather surprising since the fact-finding task is less open-ended. As a case in point, the log data shows that, on average, test persons spent 9 minutes and 17 seconds on a fact finding task and 11 minutes 17 seconds on a research task. Almost all test persons did not use the maximally allotted time of 15 minutes per task. A possible explanation is that the system did not support them well enough in finding relevant results for fact-finding (Q3.6), or at least that they expected the system to do better (which may be an unrealistic expectation based on searching for similar questions on the Web at large using Internet search engines). This is consistent with the assessment of task completion (Q3.9) where, on average, test persons were less certain that they completed the fact finding task compared to the research task. Also note that the standard deviation for fact finding task in almost all questions are larger than the research task. A possible explanation is again that several test persons were not satisfied with the results they found when completing the fact finding task. We look forward to analysing this hypothesis against the data collected in the other experiments, hoping to reveal whether it can indeed be attributed to the impact of the task type.

5.3 Findings

We reported the result of the Interactive Track experiment based on seven test persons using the post-task questionnaire and corresponding log files. We found that our test persons spent more time on completing the research task in comparison to fact finding task. We also found that test persons regarded the research task easier, were more satisfied with the search result and found more relevant information for the research task. This is plausibly related to the task type, where test persons regard more information as relevant or useful when searching for a more open-ended research task. Fact finding tasks require a more specific and precise answer, which may diminish the additional value of exploring a wide range of search results. We plan to analyze the data in greater detail, e.g., by examining possible differences between tasks types also for the data collected by other groups.

6 Link Detection Experiments

In this section, we discuss our participation in the Link The Wiki (LTW) track. LTW is aimed at detecting or discovering missing links between a set of topics, and the remainder of the collection, specifically detecting links between an origin node and a destination node, hence effectively establishing cross-links between Wikipedia articles. This year the track consisted of two tasks. What both tasks had in common was that it consisted of 2 sub-tasks; the detection of links from an ‘orphan’ (outgoing) and to an ‘orphan’ (incoming). The issue of link density and link repetition as mentioned in [33] has not been addressed, henceforth we restricted our experimentation to detecting unique cross-links.

The first task was a continuation of the track of last year with the detection of links between whole articles where no anchor or Best Entry Point (BEP) was required. A difference with the task of the previous year was the number of ‘orphan’ topics, namely a random sample of 6,600 topics which equals about 1% of the total collection. Existing links in origin nodes were removed from the topics, making these articles ‘orphans.’ A threshold of 250 was set for both the number of incoming and outgoing links.

The second task used 50 orphan topics which were submitted by the track participants and went further than link detection on the article level as the BEP was required. Another requirement for these topics was that a plausible number of outgoing and incoming links is 50. Multiple BEPs were allowed; each anchor was allowed to have 5 BEPs.

6.1 Experiments with Detection of Article-to-Article Links

Information Retrieval methods have been employed to automatically construct hypertext on the Web [1, 2]. Previous research with generic and learning approaches of link detection in the Wikipedia are for example [6, 7, 8, 12, 13, 21]. General information retrieval techniques were used in [6, 7, 8, 13]. An approach that used link structures to propagate more likely ‘linkable’ anchor texts was presented in [12], and a machine learning approach with standard classifiers was presented in [21].

We have chosen to employ a collection-independent approach with IR techniques as outlined in [6] and continue the experimentation with that approach and put it more under the test. This means we do not rely on any learning, heavy heuristics or existing link structures in the Wikipedia. Our approach is mostly based on the assumption that to detect whether two nodes are implicitly connected, it is necessary to search the Wikipedia pages for some text segments that both nodes share. Usually it is only one specific and extract string [1]. One text segment is defined as a single line, and a string that both nodes share is a potentially relevant substring. Only relevant substrings of at least 3 characters length are considered in our approach, because anchor texts of 3 characters or less do not occur frequently, and to prevent detecting too many false positives.

We adopt a *breadth m–depth n* technique for automatic text structuring for identifying candidate anchors and text node, i.e. a fixed number of documents accepted in response to a query and fixed number of iterative searches. So the similarity on the document level and text segment level is used as evidence. We used the whole document as a query with the standard Vector Space Model (VSM) implementation of Lucene [20], i.e., for a collection D, document d, query q and query term t:

$$\begin{aligned}&sim(q,d) = \\ &\textstyle\sum_{t\in q} \frac{tf_{t,q}\cdot idf_t}{norm_q} \cdot \frac{tf_{t,d}\cdot idf_t}{norm_d} \cdot coord_{q,d} \cdot weight_t,\end{aligned} \tag{16}$$

where

$$\begin{array}{rcc} tf_{t,X} &=& \sqrt{\mathrm{freq}(t,X)} \\ idf_t &=& 1+\log\frac{|D|}{\mathrm{freq}(t,D)} \\ norm_q &=& \sqrt{\sum_{t\in q} tf_{t,q}\cdot idf_t{}^2} \\ norm_d &=& \sqrt{|d|} \\ coord_{q,d} &=& \frac{|q\cap d|}{|q|}. \end{array}$$

Moreover, we also used the standard Language Modeling (LM) framework of ILPS-Lucene [11], which we already discussed in Section 2.

Before the actual link detection process starts, we do some pre-processing by extracting for each topic the title enclosed with the `<name>` tag with a regular expression and store that in a hash-table for substring matching. We do not apply case-folding, but we do remove any existing disambiguation information put between brackets behind the title.

We do not assume that links are reciprocal, so we have different approaches for detecting outgoing and incoming links, though we set a threshold of 250 for both type of links and do not allow duplicated links. Links also appear locally within an article to improve navigation on that page, but this was outside the scope of the LTW track.

- There is an *outgoing link* for an 'orphan' topic when the title of a 'foster' article occurs in the orphan topic.
- There is an *incoming link* for an orphan when the title of the orphan occurs in a foster topic.

We submitted 3 official runs based on this generic approach:

`Amsterdam_Turfdraagsterpad(UvA)_a2a_1` The whole orphan article is used as a query, where the VSM is used. The pool of plausible 'foster' (candidate) articles is the top 300 of the ranked list returned by this query. This is our baseline run.

`Amsterdam_Turfdraagsterpad(UvA)_a2a_2` The term frequencies of the orphan article are conflated, and the resulting bag of words is used as query with LM with default settings. The pool of plausible 'foster' (candidate) articles is the top 300 of the ranked list returned by this query.

`Amsterdam_Turfdraagsterpad(UvA)_a2a_3` The whole orphan article is used as a query, where the VSM is used. The pool of plausible 'foster' (candidate) articles is the top 500 of the ranked list returned by this query.

6.2 Experiments with Detection of Article-to-BEP Links

We submitted 4 runs for the article-to-BEP task. For all of these runs, we assume that the BEP is always the start of the article, thus the offset is always 0. Another difference with the first task was that actual anchor text had to be specified using the File-Offset-Length (FOL) notation. Multiple BEPs per anchor were only computed for the run `Amsterdam_Turfdraagsterpad(UvA)_a2bep_5`. The 4 official submitted Article-to-BEP runs were:

`Amsterdam_Turfdraagsterpad(UvA)_a2bep_1` The whole orphan article is used as query with the VSM, and the top 300 results are used to find potential cross-links.

`Amsterdam_Turfdraagsterpad(UvA)_a2bep_2` The term frequencies of the orphan article are conflated, and the resulting bag of words is used as query with LM with default settings. The pool of plausible 'foster' (candidate) articles is the top 300 of the ranked list returned by this query.

`Amsterdam_Turfdraagsterpad(UvA)_a2bep_3` The whole orphan article is used as query with the VSM. The top 50 ranking articles is harvested. Each of these article is used again as a query to retrieve its top 6 results. This results in a potential pool of 300 foster articles.

Table 12: Official results of the Article-to-Article runs for the Link The Wiki Track

Run	Links	MAP	P@5	Rank
Amsterdam_Turfdraagsterpad(UvA)_a2a_1	In	0.339272695	0.833743098	16/25
Amsterdam_Turfdraagsterpad(UvA)_a2a_2	In	0.287957616	0.832444188	22/25
Amsterdam_Turfdraagsterpad(UvA)_a2a_3	In	**0.357581856**	**0.837737848**	15/25
Amsterdam_Turfdraagsterpad(UvA)_a2a_1	Out	0.107164298	0.284862095	15/25
Amsterdam_Turfdraagsterpad(UvA)_a2a_2	Out	**0.108885207**	**0.293142884**	14/25
Amsterdam_Turfdraagsterpad(UvA)_a2a_3	Out	0.101743892	0.268992346	17/25

Table 13: Official results of the Article-to-BEP runs for the Link The Wiki Track

Run	Links	MAP	P@5	Rank
Amsterdam_Turfdraagsterpad(UvA)_a2bep_1	In	**0.234947342**	0.932647059	9/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_2	In	0.16157935	**0.949807692**	22/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_3	In	0.156616929	0.933123249	25/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_5	In	**0.234947342**	0.932647059	8/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_1	Out	0.097272945	0.524529751	20/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_2	Out	0.087151118	0.48392250	22/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_3	Out	0.091064252	0.604179122	21/30
Amsterdam_Turfdraagsterpad(UvA)_a2bep_5	Out	**0.143689485**	**0.748596188**	14/30

Amsterdam_Turfdraagsterpad(UvA)_a2bep_5 This run is similar to the first Article-to-BEP run, but we expanded this run by allowing more than 1 BEP for each anchor. We use the depth-first strategy, and the broader-narrower conceptualization of terms by re-grouping the extracted list of titles based on a common substring. For example, the anchor text *Gothic* could refer to the topic "Gothic," but also to topics with the titles *"Gothic alphabet"*, *"Gothic architecture"*, *"Gothic art"*, *"Gothic Chess"*, and so on.

6.3 Experimental Results

The current experimental results are evaluated against the set of existing links (in the un-orphaned version of the topics) both for the sample of 6,600 topics in the first task as well as the 50 topics of the article-to-BEP task. The results of the user assessments of the 50 topics have not been released yet of this writing. The results of our official submissions are depicted in Table 12 for the first task with the detection of links on the article-level and in Table 13 for the article-to-BEP. The performance was measured with the Mean Average Precision (MAP) and the Precision at rank like the precision at 5 percent (P@5). Moreover, our runs are put in context by ranking them in the list that includes the runs of the other participants based on the MAP score.

These results, especially the sub-optimal results for the outgoing links and the general results on the article-level, warrant some reflection on several limitations of our approach. We did exact string matching with the titles of the potential foster topics and did no apply case-folding or any kind of normalization. This means we could have incorrectly discarded a significant number of relevant foster articles (false negatives); effectively under-generating the outgoing links and under-linking the topics.

We achieved one of the highest early precisions for the incoming links for both tasks. The precision drops later on as the fallout increases, which hurts the eventual MAP score—this suggests that the ranked list of results should be cut-off earlier, and that more relevant articles should be retrieved in the top. The limitations of using whole document as query, combined with the VSM, has been tested. We also tested the LM method with default parameter values for compiling such a list by using a conflated set of terms from an orphan topic, however, this has proven not to be very effective for the accuracy of the detection of the incoming links, and also hardly affects the accuracy of the detection of the outgoing links.

6.4 Findings

In summary, we continued with our experimentation with the Vector Space Model, conducted some tests with Language Modeling, and simple string processing techniques for detecting missing links in the Wikipedia. The link detection occurred in 2 steps: first, a relevant pool of foster (candidate) articles is collected; second, substring matching with the list of collected titles to establish an actual link. We used entire orphaned articles as query. Clearly, we showed the constraints of this approach, especially on the article level. Our Article-to-BEP runs are also dependent on this first step. However, we found that we can significantly improve the accuracy of the detection of our outgoing links by generating multiple BEPs for an anchor.

7 XML Mining Track

Previous years of the XML mining track have explored the utility of using XML document structure for classification accuracy. It proved to be difficult to obtain better performance [5]. This year the data consist of a collection of wikipedia XML documents that have to be categorized into fairly high-level wikipedia categories and the link structure between these documents. Link structure has been found to be a useful additional source of information for other tasks such as ad hoc retrieval [14] and entity ranking (see Section 4). Our aim at the XML Mining Track is to examine whether link structure can also be exploited for this classification task.

7.1 Classification Model

For our baseline classification model we use a classical Naive Bayes model [23]. The probability of a category given a document is:

$$P(cat|d) = \frac{P(d|cat) * P(cat)}{P(d)} \tag{17}$$

Since $P(d)$ does not change over the range of categories we can omit it. For each document the categories are ranked by their probabilities, and the category with the highest probability is assigned to the document:

$$\begin{aligned} ass\_cat(d) &= \arg\max_{cat \in cats} P(d|cat) * P(cat) \\ &= \arg\max_{cat \in cats} P(t_1|cat) * P(t_2|cat) * .. * P(t_n|cat) * P(cat) \end{aligned} \tag{18}$$

where $t_i...t_n$ are all terms in a document. The probability of a term occurring in a category is equal to its term frequency in the category divided by the total number of terms in the category. Feature (term) selection is done according to document frequency. We keep 20% of the total number of features [32]. The link information is used as follows:

$$\begin{aligned} P_0(cat|d) &= P(cat|d) \\ P_1(cat|d) &= \sum_{d \rightarrow d'} P(d|d')P(cat|d') \\ P_2(cat|d) &= \sum_{d' \rightarrow d''} P(d|d'')P(cat|d'') \end{aligned} \tag{19}$$

where d' consist of all documents that are linked to or from d, and d'' are all documents that are linked to or from all documents d'. The probabilities are uniformly distributed among the incoming and/or outgoing links. The final probability of a category given a document is now:

$$P'(cat|d) = \mu P_0(cat|d) + (1-\mu)(\alpha P_1(cat|d) + (1-\alpha)P_2(cat|d)) \tag{20}$$

The parameter μ determines the weight of the original classification versus the weight of the probabilities of the linked documents. Parameter α determines the weight of the first order links versus the weight of the second order links.

7.2 Experiments

Documents have to be categorized into one of fifteen categories. For our training experiments, we use 66% of the training data for training, and we test on the remaining 33%. We measure accuracy, which is defined as the percentage of documents that is correctly classified, which is equal to micro average recall. Our baseline Naive Bayes model achieves an accuracy of 67.59%. Macro average recall of the baseline run is considerably lower at 49.95%. All documents in the two smallest categories are misclassified. Balancing the training data can improve our macro average recall.

When we use the link information we try three variants: do not use category information of linked data, use category information of the training data, and always use category information of linked data. Other parameters are whether to use incoming or outgoing links, μ and α. For parameter μ we tried all values from 0 to 1 with steps of 0.1, only the best run is shown. The results are given in Table 14. The accuracy of the runs using link information is at best only marginally better than the accuracy of the baseline. This means that the difficult pages, which are misclassified in the baseline model, do not profit from the link information. The links to or from pages that do not clearly belong to a category and are misclassified in the baseline run, do not seem to contribute to classification performance. These linked pages might also be more likely to belong to a different category.

On the test data we made two runs, a baseline run that achieves an accuracy of 69.79%, and a run that uses in- and outlinks, $\alpha = 0.5$ and $\mu = 0.4$, with an accuracy of 69.81%. Again the improvement in accuracy when link information is used is only marginal.

Table 14: Training Classification results

		Inlinks		Outlinks		In- and Outlinks	
Link info	α	μ	Accuracy	μ	Accuracy	μ	Accuracy
Baseline			0.6759		0.6759		0.6759
None	0.5	0.5	0.6766	1.0	0.6759	0.4	0.6764
None	0.75	1.0	0.6759	1.0	0.6759	1.0	0.6759
None	1.0	1.0	0.6759	1.0	0.6759	1.0	0.6759
Training	0.5	0.5	**0.6793**	0.4	0.6777	0.4	0.6819
Training	0.75	0.5	**0.6793**	0.5	0.6777	0.5	0.6806
Training	1.0	0.6	0.6780	0.5	0.6780	0.6	0.6777
All	0.5	0.5	0.6780	0.3	0.6816	0.4	**0.6858**
All	0.75	0.6	0.6780	0.3	0.6848	0.5	0.6819
All	1.0	0.6	0.6784	0.4	**0.6858**	0.6	0.6787

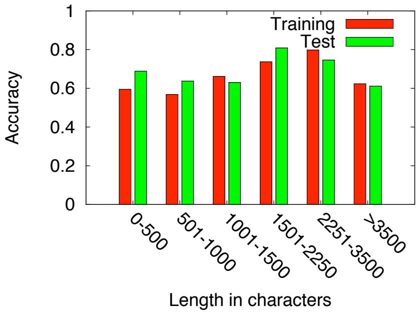

Fig. 1: Accuracy vs. document length

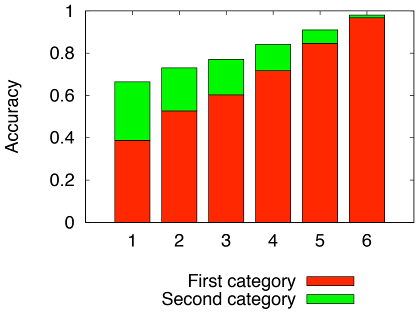

Fig. 2: Accuracy of first two categories

7.3 Analysis

We try to analyze on which kind of pages the most errors are made in our baseline run. Considering the length of pages, shorter pages do not tend to be more difficult than longer pages as can be seen in Figure 1. The difficult pages to classify can be recognized by comparing the output probability of the two highest scoring categories. This is shown in Figure 2 where we divided the training data over 6 bins of approximately the same size sorted by the fraction (P_{cat1}/P_{cat2}).

In our baseline run pages without links also seem to get misclassified more often than pages with in- and/or outlinks (see Figure 3). When link information is available, and we try to use it, there are two sources of error. The first source of error, is that not all linked pages belong to the same category as the page to classify (see Table 15). However, when we classify pages that have links using only the link information, there are some cases where the accuracy on these pages is well above the accuracy of the complete set. To obtain our test data we have used both incoming and outgoing links, which means that almost half of the pages do not belong to the same category as the page to classify. Secondly, we only know the real categories of the pages in the training data, which is only 10% of all data. For all pages in the test data, we estimate the

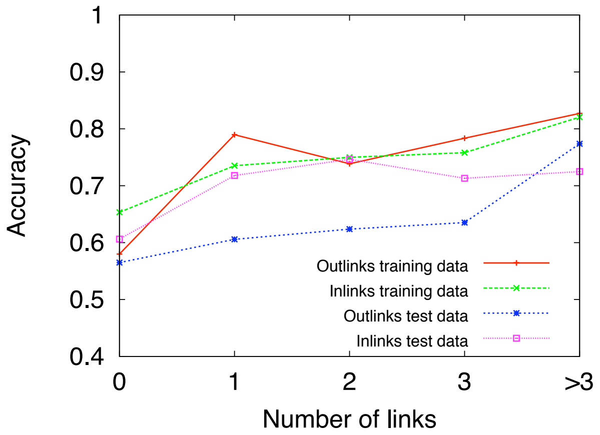

Fig. 3: Accuracy vs. number of links

Table 15: Statistics of training and test data

	# pages				links	% links with same cat.		
Data	total	with inlinks	with outlinks	with links	/page	inlinks	outlinks	links
Training	11,437	2,627 (23%)	5,288 (46%)	5,925 (52%)	0.7	76.8%	41.1%	45.8%
Test	113,366	88,174 (77%)	103,781 (91%)	107,742 (94%)	5.6	77.2%	53.4%	59.0%

probability of each category belonging to that page. With a classification accuracy of almost 70%, this means we introduce a large additional source of error.

7.4 Findings

It is difficult to use link information to improve classification accuracy. A standard Naive Bayes model achieves an accuracy of almost 70%. While link information may provide supporting evidence for the pages that are easy to classify, for the difficult pages link information is either not available or contains too much noise.

8 Discussion and Conclusions

In this paper, we documented our efforts at INEX 2008 where we participated in six tracks: Ad hoc, Book, Entity Ranking, Interactive, Link the Wiki, and XML-Mining.

For the *Ad Hoc Track*, we investigated the effectiveness of combining article and element retrieval methods. We 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 start 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. We also combined a standard content based book retrieval run with external evidence from Wikipedia. Using the idea that Wikipedia covers many of both the topics found in books and searched for by users, and the dense link graph between Wikipedia pages, we use the link distance between pages matching the query and pages matching the content of the books in the collection as measure of topical closeness. The topic assessment phase has yet to start, so we cannot report any results in this paper, but we aim to investigate the effectiveness of combination methods for document and focused retrieval techniques in a book retrieval setting, where the documents are far larger than in the Wikipedia collection. With entire books, providing a good access point to a specific topic might show focused retrieval techniques to be much more effective than traditional document retrieval.

For the *Entity Ranking Track*, we have presented our entity ranking approach where we use category and link information. Category information is the factor that proves to be most useful and we can do more than simply filtering on the target categories. Category information can both be extracted from the category titles and from the contents of the category. Link information can also be used to improve results, especially early precision, but these improvements are small.

For the *Interactive Track*, we found that our test persons spent more time on completing the research task in comparison to fact finding task. We also found that test persons regarded the research task easier, were more satisfied with the search result and found more relevant information for the research task. This is plausibly related to the task type, where test persons regard more information as relevant or useful when searching for a more open-ended research task. Fact finding tasks require a more specific and precise answer, which may diminish the additional value of exploring a wide range of search results.

For the *Link the Wiki Track*, we continued with our experimentation with the Vector Space Model, conducted some tests with Language Modeling, and simple string processing techniques for detecting missing links in the Wikipedia. The link detection occurred in 2 steps: first, a relevant pool of foster (candidate) articles is collected; second, substring matching with the list of collected titles to establish an actual link. We used entire orphaned articles as query. Clearly, we showed the constraints of this approach, especially on the article level. Our Article-to-BEP runs are also dependent on this first step. However, we found that we can significantly improve the accuracy of the detection of our outgoing links by generating multiple BEPs for an anchor.

For the *XML-Mining Track*, our aim was to explore whether we can use link information to improve classification accuracy. Previous years of the XML mining track have explored the utility of using XML document structure for classification accuracy. Link structure has been found to be a useful additional source of information for other tasks such as ad hoc retrieval. Our experiments suggest that it is difficult to use link information to improve classification accuracy. A standard Naive Bayes model achieves an accuracy of almost 70%. While link information may provide supporting evidence for the pages that are easy to classify, for the difficult pages link information is either not available or too noisy to be promoting the classification accuracy.

Acknowledgments Jaap Kamps was supported by the Netherlands Organization for Scientific Research (NWO, grants # 612.066.513, 639.072.601, and 640.001.501). Rianne Kaptein was supported by NWO under grant # 612.066.513. Marijn Koolen was supported by NWO under grant # 640.001.501. Khairun Nisa Fachry and Junte Zhang were supported by NWO under grant # 639.072.601.

Bibliography

[1] M. Agosti, F. Crestani, and M. Melucci. On the use of information retrieval techniques for the automatic construction of hypertext. *Information Processing and Management*, 33: 133–144, 1997.

[2] J. Allan. Building hypertext using information retrieval. *Information Processing and Management*, 33:145–159, 1997.

[3] N. Craswell, S. Robertson, H. Zaragoza, and M. Taylor. Relevance weighting for query independent evidence. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 416–423. ACM Press, New York NY, USA, 2005.

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

[5] L. Denoyer and P. Gallinari. Report on the XML mining track at INEX 2007: categorization and clustering of XML documents. *SIGIR Forum*, 42(1):22–28, 2008.

[6] 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.

[7] S. Fissaha Adafre and M. de Rijke. Discovering missing links in wikipedia. In *LinkKDD '05: Proceedings of the 3rd international workshop on Link discovery*, pages 90–97. ACM Press, New York NY, USA, 2005.

[8] S. Geva. GPX: Ad-hoc queries and automated link discovery in the Wikipedia. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 404–416. Springer Verlag, Heidelberg, 2008.

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

[10] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In *Proceedings SIGIR 2004*, pages 178–185. ACM Press, New York NY, 2004.

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

[12] K. Y. Itakura and C. L. A. Clarke. University of waterloo at INEX 2007: Adhoc and link-the-wiki tracks. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 417–425. Springer Verlag, Heidelberg, 2008.

[13] D. Jenkinson and A. Trotman. Wikipedia ad hoc passage retrieval and wikipedia document linking. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 426–439. Springer Verlag, Heidelberg, 2008.

[14] J. Kamps and M. Koolen. The importance of link evidence in Wikipedia. In *Advances in Information Retrieval: 30th European Conference on IR Research (ECIR 2008)*, pages 270–282, 2008.

[15] J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjörnsson. The importance of morphological normalization for XML retrieval. In *Proceedings of the First Workshop of the INitiative for the Evaluation of XML retrieval (INEX)*, pages 41–48. ERCIM Publications, 2003.

[16] 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.

[17] 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.

[18] 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.

[19] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, New York NY, USA, 2002.

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

[21] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge mining*, pages 509–518. ACM Press, New York NY, USA, 2008.

[22] N. Pharo, R. Nordlie, and K. N. Fachry. The interactive track at INEX 2008. In *This Volume*, 2008.

[23] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

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

[25] 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.

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

[27] 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.

[28] 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.

[29] T. Tsikrika, P. Serdyukov, H. Rode, T. Westerveld, R. Aly, D. Hiemstra, and A. P. de Vries. Structured document retrieval, multimedia retrieval, and entity ranking using PF/Tijah. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 306–320. Springer Verlag, Heidelberg, 2008.

[30] A.-M. Vercoustre, J. Pehcevski, and J. A. Thom. Using wikipedia categories and links in entity ranking. In *Proceedings INEX 2007*, volume 4862 of *LNCS*, pages 321–335. Springer Verlag, Heidelberg, 2008.

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

[32] K. Williams. Ai::categorizer - automatic text categorization. Perl Module, 2003.

[33] J. Zhang and J. Kamps. Link detection in XML documents: What about repeated links? In A. Trotman, J. Kamps, and S. Geva, editors, *Proceedings of the SIGIR 2008 Workshop on Focused Retrieval*, pages 59–66. University of Otago, Dunedin New Zealand, 2008.