LIA at INEX 2010 Book Track

Romain Deveaud, Florian Boudin and Patrice Bellot

Laboratoire Informatique d'Avignon - University of Avignon (CERI-LIA)
339, chemin des Meinajariès, F-84000 Avignon Cedex 9
firstname.lastname@univ-avignon.fr

Abstract. In this paper we describe our participation and present our contributions in the INEX 2010 Book Track. Digitized books are now a common source of information on the Web, however OCR sometimes introduces errors that can penalize Information Retrieval. We propose a method for correcting hyphenations in the books and we analyse its impact on the Best Books for Reference task. The observed improvement is around 1%.
This year we also experimented different query expansion techniques. The first one consists of selecting informative words from a Wikipedia page related to the topic. The second one uses a dependency parser to enrich the query with the detected phrases using a Markov Random Field model. We show that there is a significant improvement over the state-of-the-art when using a large weighted list of Wikipedia words, meanwhile hyphenation correction has an impact on their distribution over the book corpus.

1 Introduction

The number of books available in electronic format increases continuously. The mass-digitization of books is creating large digital libraries containing information about a broad range of topics. Well known examples of these digital libraries are the project Gutenberg[1] and Google Books[2], allowing people to read books for free on different devices (iPhone, iPad, Kindle...). The development of specialized Information Retrieval methods for this kind of documents is a real issue for the community.

The electronic representation of books are obtained using an Optical Character Recognition (OCR) process that automatically generates the machine-encoded text corresponding to the images of the pages. However it generally introduces some errors [10], increasing the difficulty for retrieval models to deal with these documents. Hyphenated words are one source of errors. They are introduced to control line wrapping in the physical books, but they will be interpreted as two different words at the indexing step. Considering the following lines[3]:

[1] www.gutenberg.org
[2] books.google.com
[3] Extract from *1984*, Georges Orwell, http://books.google.com

On each landing, opposite the lift
shaft, the poster with the **enor-**
mous face gazed from the wall.

In this example, the terms "*enor-*" and "*mous*" are indexed instead of "*enormous*". As far as we know, no previous work has reported experiments on the impact of word hyphenation correction on book retrieval effectiveness. We propose in Section 2 a simple and efficient approach for correcting hyphenated words, which produce an almost errorless version of the corpus. We evaluate its performance on the Best Books for Reference task with the topics and the *qrels* of the 2009 and 2010 Book tracks.

Several studies had been conducted on Information Retrieval (IR) inside books in general, and some of them show that indexing specific parts (e.g. headers, titles or table of contents) is nearly as effective as indexing the entire content of books [6, 11]. However, considering these information are not always available, we did not take into account these different parts in our experiments.

This year we tried different query expansion and query enrichment methods for the Best Books for Reference task. We started by using Wikipedia as an external source of knowledge for selecting informative words. A Wikipedia page is associated to each topic, and the most informative words of the pages are selected to expand the queries. This approach was already studied in [4] and we enhanced it with some features, different weighting schemes and several term extraction measures. We finally observed that on previous year *qrels* (2009), the *entropy* measure for selecting the words within the Wikipedia page gives the better results. As a second approach for modifying the query, we used the Stanford Parser [1] to extract phrases from the topics and applied a Markov Random Field model [8] to enrich the query.

The rest of the paper is organized as follows. In Section 2, we present our contribution with the correction of the hyphenated words in the corpus and its evaluation. Then, we detail in Section 3 our retrieval framework, our query expansion approaches and the runs we submitted for the Best Books for Reference task. Finally, we present our results in Section 4.

2 Word Hyphenation Correction

Although we observed a small amount of OCR errors in the corpus, there is a large number of hyphenations. To tackle this problem, we decided to reconstruct hyphenated words using a lexicon made of 118,221 unique words extracted from the English Gigaword corpus[4].

The correction algorithm iterates through each couple of successive lines and generates word candidates from the last substring of the first line and the first substring of the second line. The candidate word is then corrected if it occurs in the lexicon.

[4] LDC Catalog No. LDC2007T07, Available at `www.ldc.upenn.edu`

We evaluated the correction impact with the official *qrels* and topics from the 2009 and 2010 Book Tracks [2, 3]. This year, 21 topics were validated over the 82 initially proposed, but there were fewer relevance judgements than in 2009, where only 16 topics were validated. Table 1 sums up some information about the collection. We can note that the standard deviation is high, it reveals that the sizes of the books vary a lot within the collection.

Table 1. Some important numbers about the book collection.

Number of books	50,239
Size of the collection (words)	5,080,414,177
Size of the largest book	1,659,491
Size of the smallest book	590
Average size	101,125
Standard deviation of the size	102,127

The collection contains 613,107,923 lines, in which 37,551,834 (6,125%) were corrected by our method. To measure how much book retrieval is impacted by these corrections, we tested with three configurations of the same retrieval model. This model uses a Language Modeling (LM) approach to IR with different Dirichlet prior smoothing (μ) values, along with the stopword list provided by Lemur and the Porter stemmer. Queries are generated from the `<title>` fields of the 2009 topics and the `<query>` fields of the 2010 topics. The number of retrieved books is set to 100. Results are reported in Table 2 for the 2009 topics and in Table 3 for the 2010 ones.

Table 2. Book retrieval results on both initial and corrected Book Track corpus, with the 2009 topics and qrels, in terms of Mean Average Precision (MAP) and precision at 10 (P@10).

Model	**Uncorrected data**		**Corrected data**	
	MAP	P@10	MAP	P@10
LM, $\mu = 2500$	0.302	0.486	0.304	0.507
LM, $\mu = 1000$	0.299	0.493	0.302	0.507
LM, $\mu = 0$	0.244	0.443	0.243	0.450

Despite the sizeable number of corrected words, the improvement is relatively low, however we note that it is in the same order for these two years ($\approx 1\%$). As said previously, books are larger that traditionnal web documents and there are very few words that appear only once in a book. Hence, the errors introduced by some misspelled words are greatly reduced.

Table 3. Book retrieval results on both initial and corrected Book Track corpus, with the 2010 topics and qrels, in terms of Mean Average Precision (MAP) and precision at 10 (P@10).

Model	Uncorrected data		Corrected data	
	MAP	P@10	MAP	P@10
LM, $\mu = 2500$	0.444	0.581	0.447	0.586
LM, $\mu = 1000$	0.435	0.576	0.439	0.581
LM, $\mu = 0$	0.390	0.528	0.393	0.524

Apart from the word hyphenation correction, we can see that the scores vary a lot between the 2009 and the 2010 topics. The very high scores achieved by this basic model on this year topics results from the small amount of assessments that could be collected. We used the corrected version of the corpus for all our further experiments and all the runs we submitted.

3 Best Books for Reference

3.1 Retrieval model

All the runs and experiments we will further describe follow the same retrieval model. We use Indri, which is part of the Lemur project[5] and provide an implementation of a LM approach for retrieval [7]. The embedded stoplist provided by Lemur is used for stopword removal along with the standard Porter stemmer.

Given a sequence of query terms $Q = (q_1, ..., q_n)$ treated as a bag of words, the scoring function of a document D is defined as follow :

$$s_Q(D) = \prod_{i=1}^{n} p_D(q_i)^{\frac{1}{n}}$$

$p_D(\cdot)$ is estimated by Maximum Likelihood Estimation with Dirichlet prior smoothing :

$$p_D(q_i) = \frac{tf_{q_i,D} + \mu \times p_{\mathcal{C}}(q_i)}{|D| + \mu}$$

where $\mathcal{C}$ is the entire collection, $|D|$ the size of the documents and $tf_{q_i,D}$ the frequency of the query term q_i in the document D. The μ parameter is empirically set to 2500, which is also the default value proposed by Indri.

3.2 Baselines

The first baseline (namely **baseline_1**) uses the content of the `<query>` fields of the topics, while the second one (**baseline_2**) uses the content of the `<fact>`

[5] `www.lemurproject.org`

fields. We submitted two other baselines (**baseline_1_wikifact** and **baseline_2_wikifact**) which are exactly the same as before, except that we add the content of the `<wikifact>` field, when it is available. The text of this field corresponds to the first paragraph of a Wikipedia page identified as related to the topic. Queries are treated as bag of words and retrieval is performed using the model described in Section 3.1. Results are presented in Section 4.

3.3 Contextual Query Expansion using Wikipedia

Several studies previously investigated the use of Wikipedia as an external corpus for Query Expansion [4, 5, 9, 12]. In their approach, Koolen *et al.* [4] extract useful terms from Wikipedia pages to expand queries and use them for Book Retrieval. A page is selected by querying Wikipedia with the original query and getting the page that matches the query, or the best result. The well-known $tf.idf$ measure is then computed for each word of the selected Wikipedia page, and the expanded query is formed by adding the top-ranked N words to the original query. The *idf* values are computed within the whole test collection. They employ a simple term weighting method: the original query terms are weighted N times more than the N added terms. We started by expanding this work.

We use the `<wikiurl>` field, when available, to get a Wikipedia page closely related to each topic. Otherwise we query the Wikipedia search engine with the `<query>` field and we select the best ranked article. Then we extract terms from this article using different measures described below (*tf*, *tf.idf*, *entropy*...) and we use them to expand the query. We also keep the scores of these term selection measures in order to weigh the words inside the query. Indeed, some terms are more important than other in the Wikipedia page, and a representation of this relative importance is the score of the measure. A weight is therefore associated to each selected term.

In the following runs, we used the `<query>` field as the original query. We also noticed that the `<fact>` field was most of the time a *cut-and-paste* sentence from a book, therefore we used it as a first query expansion. Given a sequence of query terms $Q = (q_1, ..., q_k)$, a sequence of fact terms $F = (f_1, ..., f_m)$ and a list of weighted terms $T_Q = \{(t_1, w_1), ..., (t_n, w_n)\}$ extracted from related Wikipedia pages, we rank books according to the following scoring function $\Delta_Q(D)$:

$$\Delta_Q(D) = \left(\prod_{i=1}^{k} p_D(q_i)^{\frac{1}{k}}\right)^{\frac{X}{X+Y+Z}} \times \left(\prod_{i=1}^{m} p_D(f_i)^{\frac{1}{m}}\right)^{\frac{Y}{X+Y+Z}} \times \left(\prod_{i=1}^{n} p_D(t_i)^{\frac{w_i}{\sum_{j=1}^{n} w_j}}\right)^{\frac{Z}{X+Y+Z}}$$

Here, we could not learn an appropriate weighting scheme for the X, Y and Z weights, so they were set empirically. We gave the same weight to the `<query>`

and the `<fact>` fields ($X = Y = 4$), whereas the expansion terms were weighted half ($Z = 2$). We use these weights for all the runs featuring Wikipedia query expansion.

In the following runs, we split the Wikipedia pages into chunks with Tree-Tagger[6]. Therefore, a term can be composed of one or many words.

Fact_query_tfwiki Run In this run, the terms from the associated Wikipedia page are ranked by tf, and the top 10 ones are selected for the expansion. Their scores are also normalized inside the expansion in order to weight appropriately the important words.

Fact_query_tfidfwiki Run This run is practically the same as above, except that we rank the terms by $tf.idf$, where the idf is computed whithin the whole collection. The scores are also normalized and used in the expansion.

Fact_query_entropy Run This run is similar to the `fact_query_tfidfwiki` run but the term selection measure is only computed within the associated Wikipedia page. We use an *entropy* measure to rank the words accordingly to their informativeness, and the 10 words with best score are selected for the expansion. Considering a sequence of words $S = (w_1, ..., w_n)$, the *entropy* measure we use is defined as follow:

$$E(S) = -\sum_{i=1}^{n} p(w_i) \log_2(p(w_i))$$

Where the $p(w_i)$ are computed within the whole Wikipedia article.

Fact_query_10bestswiki Run This run is a sort of query expansion baseline. Indeed we didn't normalized the scores inside the expansion and the selected terms are words and not chunks from Tree-Tagger. As for the prior runs, the 10 most frequent words are selected for the expansion.

3.4 Using the Stanford Parser

In this model, we consider multiword phrases. It is clear that finding the exact phrase "*New York*" is a much stronger indicator of relevance than just finding "*New*" and "*York*" scattered within a document. We use Metzler and Croft's Markov Random Field model [8] to integrate that. In this model three features are considered: single term features (standard unigram language model features), exact phrase features (words appearing in sequence) and unordered window features (require words to be close together, but not necessarily in an exact sequence order). Features weights are set according to the authors's recommendation. Multiword phrases are detected using the Stanford parser [1]. In this

[6] `www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`

work, we use the typed dependency representation of the `<fact>` fields to extract complex noun phrases (e.g. "*london daily mail*", "*sioux north american shields*" or "*symphony no 3*"). The submitted run is named **fact_stanford_deps**.

3.5 Applying the Book Track's contextual Query Expansion to the Ad Hoc Track

In this Section we briefly present the generalization of the approach we tried in the Book Track. This approach consists of expanding the user query with contextual words taken from a Wikipedia page related to the topic. The contextual Wikipedia page is obtained by querying the Wikipedia search engine with the original user query (which is the `<title>` field in the INEX Ad Hoc topics). A page is automatically selected if the query matches its title. Otherwise, we assume that the first result returned by the Wikipedia search engine is relevant enough.

We developped a specific tool named Mirmiri[7] to achieve this automatic selection. This tool is a Ruby library which provides some utilities related to Information Retrieval and Text Processing in general. It is Open-Source and available for free[8].

We produced a run which expands the original query with 200 words taken from the related Wikipedia page, and we evaluated it with the official *qrels*. The retrieval is done at the Document level (i.e. Relevance in Context task).

Table 4. Document retrieval results on the Relevance in Context task in terms of Mean Average Precision (MAP) and Precision at 5 and 10 documents (P@5 and P@10).

Run	MAP	P@5	P@10
`p22-Emse301R`	0.4292	0.6962	0.6423
`qe_wiki_entropy`	0.3858	0.6500	0.6077
`Reference Run`	0.3805	0.6423	0.5750
`baseline`	0.3496	0.6115	0.5596

We can see in Table 4 that it does not beat the best run of this year (`p22-Emse301R`) for the Relevance in Context Tak. However it outperforms the Reference Run given by the INEX organizers which was high this year, due to the tuning on the 2009 topics and judgements. The `baseline` run we used in this experiment is obtained by using only the `<title>` fields as a bag of words (i.e. the same run as `qe_wiki_entropy` but without expansion terms). The results show that expanding the `baseline` queries with Wikipedia terms leads to an improvement of the perfomance wich is situated in the order of 10%. These different results confirm that this approach performs well for the retrieval of whole documents.

[7] `mirimiri.org`
[8] `github.com/romaindeveaud/mirimiri`

4 Results

The official results of the Book Track are presented in Table 5.

Table 5. Evaluation results of all runs submitted for the Best Books for Reference task. Our run identifiers are prefixed with `p98`.

Runs	MAP	P@10	NDCG@10
p14-BOOKS2010_T2_PAGE_SUM_300.trec	0.5050	0.6667	0.6579
p98-baseline_2_wikifact.trec	**0.5044**	**0.6381**	**0.6500**
p98-baseline_2.trec	0.4806	0.6143	0.6302
p98-fact_query_tfwiki.trec	0.4706	0.5571	0.5919
p98-fact_stanford_deps.trec	0.4573	0.5857	0.5976
p98-baseline_1_wikifact.trec	0.4565	0.5905	0.5960
p98-baseline_1.trec	0.4374	0.5810	0.5764
p98-fact_query_10wikibests.trec	0.4328	0.5714	0.5638
p98-fact_query_entropy.trec	0.4250	0.5476	0.5442
p14-BOOKS2010_T2FB_BASE_BST.trec	0.3981	0.5048	0.5456
p98-fact_query_tfidfwiki.trec	0.3442	0.4667	0.4677
p6-inex10.book.trec	0.3286	0.4429	0.4151
p6-inex10.book.fb.10.50.trec	0.3087	0.4286	0.3869
p14-BOOKS2010_CLM_PAGE_SUM_300.trec	0.1640	0.2810	0.2156
p14-BOOKS2010_CLM_PAGE_SUM.trec	0.1507	0.2714	0.2017

We can see that our baselines that use the `<fact>` fields achieve the best results. This behaviour can be explained by the fact that the `<fact>` are actual full sentences taken from the books for most of the topics. Again, relevance judgements are a bit insufficient considering the number of topics, and it favours the books containing these sentences. Assessments sparsity also explains the high scores achieved by the best runs. The **fact_stanford_deps**, ranked fifth, also performed well considering that no external resources are involved except a dependency parser.

We see that the *term-frequency* measure for selecting expansion words within a Wikipedia page performs better than the *tf.idf* or the *entropy*. It denies our initial intuition which was that the *entropy* would perform better. Indeed the good results presented in Section 3.5 led us to think that query expansion was an efficient approach, and that the *entropy* was an appropriate measure for selecting important and informative words. There are two main reasons that can explain the relative gap between the Ad Hoc and the Book results for the same method. First, the vocabulary of a common encyclopedia and of 19^{th} century books is very different, and can cause word mismatch. Second, the fact that full sentences directly taken from the books appear in the topics highly favours the query words. The small number of assessments collected also plays a major role here.

To highlight this problem we experimented the same method with the 2009 topics and *qrels* and compared it with the best run from last year and the

same baseline as **baseline_1**. The results presented in Table 6 show that the *entropy* measure achieves the best results overall when performing the contextual query expansion we presented in Section 3.3. This experiment indicates that this approach is efficient and can achieve high scores even with very different documents.

Table 6. Evaluation results for the 2009 INEX Book Retrieval task in terms of Mean Average Precision (MAP) and Precision at 10 documents (P@10).

Run	MAP	P@10
`qe_wiki_entropy`	0.363	0.593
`BR_inex09.book.fb.10.50` (best 2009 run)	0.347	0.486
`baseline_1`	0.304	0.507

5 Conclusions

In this paper we presented our contributions to the INEX Book Track. We proposed to enhance Book Search performance by correcting word hyphenations and produced a corrected version of the collection. Although we cannot see a significant improvement on a Book Retrieval task the retrieval accuracy of the models we experimented were all enhanced. We expect that this correction can lead to better better in focused search tasks such as page or extent retrieval.

We also presented the runs we submitted within the Best Books for Reference task. Our baselines achieved the best results mainly because of the topics that were containing unmodified sentences from books, and also because of the small number of relevance judgements collected. However we evaluated the query expansion approach on the 2009 topics and *qrels* and we showed that an appropriated weighting scheme combined to a score normalization between the terms of the expansion leads to better results.

References

1. M.C. De Marneffe, B. MacCartney, and C.D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC'06 conference*, 2006.
2. Gabriella Kazai, Antoine Doucet, Marijn Koolen, and Monica Landoni. Overview of the inex 2009 book track. In *Proceedings of the Focused retrieval and evaluation, and 8th international conference on Initiative for the evaluation of XML retrieval*, INEX'09, pages 145–159, Berlin, Heidelberg, 2010. Springer-Verlag.
3. Gabriella Kazai, Marijn Koolen, Antoine Doucet, and Monica Landoni. Overview of the inex 2010 book track: At the mercy of crowdsourcing. In *Proceedings of the Focused retrieval and evaluation, and 9th international conference on Initiative for the evaluation of XML retrieval*, INEX'10, Berlin, Heidelberg, 2011. Springer-Verlag.

4. Marijn Koolen, Gabriella Kazai, and Nick Craswell. Wikipedia pages as entry points for book search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 44–53, New York, NY, USA, 2009. ACM.
5. Yinghao Li, Wing Pong Robert Luk, Kei Shiu Edward Ho, and Fu Lai Korris Chung. Improving weak ad-hoc queries using wikipedia as external corpus. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 797–798, New York, NY, USA, 2007. ACM.
6. Walid Magdy and Kareem Darwish. Book search: indexing the valuable parts. In *Proceeding of the 2008 ACM workshop on Research advances in large digital book repositories*, BooksOnline '08, pages 53–56, New York, NY, USA, 2008. ACM.
7. D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Inf. Process. Manage.*, 40:735–750, September 2004.
8. Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 472–479, New York, NY, USA, 2005. ACM.
9. David N. Milne, Ian H. Witten, and David M. Nichols. A knowledge-based search engine powered by wikipedia. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 445–454, New York, NY, USA, 2007. ACM.
10. Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic ir to ocr text. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pages 202–211, New York, NY, USA, 1994. Springer-Verlag New York, Inc.
11. Hengzhi Wu, Gabriella Kazai, and Michael Taylor. Book search experiments: investigating ir methods for the indexing and retrieval of books. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 234–245, Berlin, Heidelberg, 2008. Springer-Verlag.
12. Yang Xu, Gareth J.F. Jones, and Bin Wang. Query dependent pseudo-relevance feedback based on wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 59–66, New York, NY, USA, 2009. ACM.