Collaborative Filtering for Book Recommendation

Chahinez Benkoussas[1,2], Hussam Hamdan[1,2], Shereen Albitar[1,2], Anaïs Ollagnier[1,2] and Patrice Bellot[1,2]

[1] Aix-Marseille Université, CNRS, LSIS UMR 7296, 13397, Marseille, France
{chahinez.benkoussas, hussam.hamdan, shereen.albitar, anais.ollagnier, patrice.bellot}@lsis.org
[2] Aix-Marseille Université, CNRS, CLEO OpenEdition UMS 3287, 13451, Marseille, France
{chahinez.benkoussas, hussam.hamdan, shereen.albitar, anais.ollagnier, patrice.bellot}@openedition.org

Abstract. In this paper, we present our contribution in INEX 2014 Social Book Search Track. This track aims to exploit social information (users reviews, ratings, etc...) from LibraryThing and Amazon collections. In our experiments we used different methods, one of our submissions which uses INL2 got the second rank w.r.t nDCG@10 measure, the official measure for this task. In addition, we tested the combination of the Sequential Dependence Model (SDM) and the use of social information that takes into account ratings,tags and customer reviews, we also tested several query expansion techniques: concept expansion, tag expansion and pseudo relevance feedback.

Keywords: XML retrieval, controlled metadata, book recommendation, reranking, query expansion, pseudo relevance feedback.

1 Introduction

Previous editions of the INEX Book Track focused on the retrieval of real out-of-copyright books [2]. These books were written almost a century ago and the collection consisted of the OCR content of over 50 000 books. The topics and the books of the collection differ in vocabulary and writing style. Information Retrieval systems had difficulties to find relevant information, and assessors had difficulties in judging the relevance of documents.

The document collection is composed of the Amazon [3]pages of real books. IR must search through editorial data, user reviews and ratings of each book, instead of searching through the whole content of the book. The topics were extracted from LibraryThing [4] forums and they represent real requests from real users.

We tested several approaches for retrieval. We submitted 6 runs in which we used the reviews and the ratings attributed to books by Amazon users. We

[3] http://www.amazon.com/

[4] http://www.librarything.com/

computed a “social score” for each book, considering the amount of reviews and the ratings. We also performed topic conceptualization for query expansion and pseudo relevance feedback using tags and important terms for retrieved books.

The rest of this paper is organized as follows. The following section describes our retrieval frameworks. In section 3, we describe the submitted runs. Finally, we present the obtained results in section 4.

2 Retrieval Model

2.1 InL2

We used InL2 model implemented in Terrier. InL2 is DFR-based model (Divergence From Randomness). The DFR models are based on this idea: "The more the divergence of the within-document term-frequency from its frequency within the collection, the more the information carried by the word t in the document d" [7]. InL2 signifies Inverse Document Frequency model with Laplace after-effect and normalization 2.

2.2 Sequential Dependence Model

We used a language modeling approach to retrieval [4]. We use *Metzler* and *Croft's* Markov Random Field (MRF) model [5] to integrate multi word phrases in the query. Specifically, we use the Sequential Dependence Model (SDM), which is a special case of MRF. In this model, three features are considered: single term features (standard unigram language model features, f_T), exact phrase features (words appearing in sequence, f_O) and unordered window features (require words to be close together, but not necessarily in an exact sequence order, f_U).

Finally, documents are ranked according to the following scoring function:

$$\begin{aligned} SDM(Q,D) = \lambda_T \sum_{q \in Q} f_T(q,D) \\ +\lambda_O \sum_{i=1}^{|Q|-1} f_O(q_i, q_i+1, D) \\ +\lambda_U \sum_{i=1}^{|Q|-1} f_U(q_i, q_i+1, D) \end{aligned}$$

Where the feature weights are set according to the author's recommendation ($\lambda_T = 0.85$, $\lambda_O = 0.1$, $\lambda_U = 0.05$). f_T , f_O and f_U are the log maximum likelihood estimates of query terms in document D as shown in Table 1, computed over the target collection using a Dirichlet smoothing.

Table 1. Language modeling-based unigram and term weighting functions. Here, $tf_{e,D}$ is the number of times term e matches in document D, $cf_{e,D}$ is the number of times term e matches in the entire collection, $|D|$ is the lenght of document D, and $|C|$ is the size of the collection. Finaly, μ is a weighting function hyperparameter that is set to 2500.

Weighting	Description
$f_T(q_i, D) = log[\frac{tf_{q_i,D}+\mu\frac{cf_{q_i}}{\lvert C\rvert}}{\lvert D\rvert+\mu}]$	Weight of unigram q_i in document D.
$f_O(q_i, q_{i+1}, D) = log[\frac{tf_{\#1(q_i,q_{i+1}),D}+\mu\frac{cf_{\#1(q_i,q_{i+1})}}{\lvert C\rvert}}{\lvert D\rvert+\mu}]$	Weight of exact phrase "q_i q_{i+1}" in document D.
$f_O(q_i, q_{i+1}, D) = log[\frac{tf_{\#uw8(q_i,q_{i+1}),D}+\mu\frac{cf_{\#uw8(q_i,q_{i+1})}}{\lvert C\rvert}}{\lvert D\rvert+\mu}]$	Weight of unordered window "q_i q_{i+1}" (span = 8) in document D.

2.3 Pseudo Relevance Feedback

We deployed the query expansion (Pseudo Relevance Feedback) mechanism implemented in Terrier[5], this mechanism is a generalization of Rocchio's method [8]. It adds the terms from the top-ranked retrieved documents to the query and re-weights the query terms by taking into account the pseudo relevance set. We used the expansion model Bo1 that is based on the Bose-Einstein statistics and on the DFR framework, its efficacy for the standard TREC collections and tasks, is proven in [6] and [3]. We extended the query of each topic by the first 10 most informative terms in the first 3 top ranked documents.

We also used book tags for query expansion. We selected from the pseudo relevance set (the 10 first retrieved books) the tags which are attributed by more then 3 users (having "count" > 3). Then, we performed query expansion with the selected tags for each topic. The following XML code illustrates an example of an extended query with tags.

```
<topic id="1116">
   <title>Which LISP?</title>
   <mediated_query>introduction book to Lisp</mediated_query>
   <group>Purely Programmers</group>
   <narrative> It'll be time for me to shake things up and learn a new language soon. I had started on Erlang a while back and
          getting back to it might be fun. But I'm starting to lean toward Lisp--probably Common Lisp rather than Scheme. Anyone
          care to recommend a good first Lisp book? Would I be crazy to hope that there's one out there with an emphasis on using
           Lisp in a web development and/or system administration context? Not that I'm unhappy with PHP and Perl, but the best
          way for me to find the time to learn a new language is to use it for my work...
   </narrative>
   <feedback_tags>['artificial intelligence', 'Computing', 'Computers', 'non-fiction', 'ai', 'Reference', 'computer science', '
          programming', 'programming languages', 'Computer programming', 'lisp', 'artificial intelligence', 'ai', 'Reference', '
          computer science', 'Computing', 'own', 'wishlist', 'cs', 'commonlisp', 'Emacs', 'Emacs']
   </feedback_tags>
</topic>
```

[5] http://terrier.org/

2.4 Query Expansion With Concepts

In order to deploy semantics in book retrieval, conceptualization phase extracts mappings from semantic resources for terms in the topic. In these experiments, we used DBpedia[6] a semantic resource and DBpedia Spotlight[7] for word to concept mapping. Spotlight is a tool for semantic text annotation that searches for candidate terms and then searches for adequate mappings between these terms and concepts in DBpedia. A mapped concept might be a direct match such as *(Berlin → Berlin)* or approximative such as *(embassies → Diplomatic mission)*.

In the context of our participation in Inex Social Book Search, we applied conceptualization on the "narrative" field of each topic and then extended the query with resulting concepts from DBpedia. Thus, classical IR models can take into consideration topic semantics, that are expressed in natural language by the user in the narrative, which might enhance the relevance of the results. In following, we show an example of "narrative" field conceptualization of the previous topic. We combined in "extended_query" tag both the "mediated_query" content and the obtained concepts.

```
<topics>
  <topic id="1116">
    <title>Which LISP?</title>
    <mediated_query>introduction book to Lisp</mediated_query>
    <group>Purely Programmers</group>
    <narrative>  It'll be time for me to shake things up and learn a new language soon. I had started on Erlang a while back and
            getting back to it might be fun. But I'm starting to lean toward Lisp--probably Common Lisp rather than Scheme. Anyone
            care to recommend a good first Lisp book? Would I be crazy to hope that there's one out there with an emphasis on using
             Lisp in a web development and/or system administration context? Not that I'm unhappy with PHP and Perl, but the best
            way for me to find the time to learn a new language is to use it for my work... </narrative>
    <concepts_narrative>Book Master_Shake Learning Depression_\%28mood\%29 Administration_\%28government\%29 Erlang_\%28
            programming_language\%29 Context_menu Developmental_psychology Common_good Crazy_\%28Gnarls_Barkley_song\%29 Emphasis_
            \%28typography\%29 Perl Good_and_evil Common_Lisp PHP CARE_\%28relief_agency\%29 Lean_manufacturing Hope Language
            System Scheme_\%28programming_language\%29
    </concepts_narrative>
    <extended_query>introduction book to Lisp Book Master Shake Learning Depression (mood) Administration (government) Erlang (
            programming language) Context menu Developmental psychology Common good Crazy (Gnarls Barkley song) Emphasis (
            typography) Perl Good and evil Common Lisp PHP CARE (relief agency) Lean manufacturing Hope Language System Scheme (
            programming language)
    </extended_query>
</topic>
```

2.5 Modeling book likeliness

We modeled book likeliness based on the following idea: the more the number of reviews it has, the more interesting the book is (it may not be a good or popular book but a book that has a high impact) [1].

$$Likliness(D) = \frac{\sum_{r \in R_D} r}{|Reviews_D|}$$

where R_D is the set of all ratings given by the users for the book D, and $|Reviews_D|$ is the number of reviews. We further re-ranked books according to a linear interpolation of the previously computed SDM score with the likeliness score, using a coefficient (α) to control the influence of each model. The scoring function of a book D given a query Q is thus defined as follows:

[6] http://en.wikipedia.org/wiki/DBpedia

[7] spotlight.dbpedia.org

$$SDM\_Likeliness(Q, D) = \alpha.(SDM(Q, D)) + (1 - \alpha).(Likliness(D))$$

Where α is a constant that is set according to previous results (obtained on 2012 and 2013 datasets), with the value of 0.89.

3 Runs

We submitted 6 runs for the Social Book Search Task. We used 2 IR systmes for indexing and searching: Indri[8] and Terrier. We performed a preprocessing step to convert Inex SBS corpus into Trec Collection Format[9], we consider that the content of all tags in each XML file is important for indexing; therefore we take the whole XML file as one document identified by its ISBN. Thus, we just need two tags instead of all tags in XML, the ISBN and the whole content (named text) following this format:

```
<book>
  <isbn>123</isbn>
  <text>the content of first book</text>
</book>
<book>
  <isbn>124</isbn>
  <text>the content of second book</text>
</book>
```

Inex SBS corpus is composed of 2.8 million documents, distributed in 1100 folders, we generate for each folder only one Trec formatted file which contains all xml files in this folder. In fact this processing is necessary for improving the execution time of Terrier indexing process.

InL2:
This run is based on InL2 model, the index is built on all fields in the book xml files, for each topic we use "mediated_query", group, narrative tags as a query.

InL2Feedback:
This run is based on InL2 model, the index is built on all the fields in the book xml files, we extended the topics by the 10 most informative terms in the 3 top ranked files.

InL2tagFeedback:
This run is based on InL2 model, the index is built on all fields in the book xml files, we extended the topics by the tags extracted on the top 10 books retrieved and ranked by InL2 where the tag count is more than 3.

SDM_Rating:

[8] http://www.lemurproject.org/

[9] http://lab.hypotheses.org/1129

This run combines the implementation of the Sequential Dependence Model and the use of social information which is the "Ratings" given by users. We re-rank books according to a linear interpolation of the SDM model with the average of "Ratings" values, using a coefficient (b) to control the influence of each model. Only the "mediated_query" field of the topic was used for this run.

SDM_Concept:

This run is the implementation of the Sequential Dependence Model (SDM) which is a special case of the Marcov Random Field (MRF) model. Three features are considered: Single Term Feature (standard unigram language model feature), Exact Phrase Features (words appearing in sequence) and Unordered Window Features (words appearing close together, but not necessarily ordered). The "mediated_query" has been extended by concepts extracted from narrative tag using DBpedia spotlight, this extended query has been used for this run.

SDM_Tag_Feedback:

This run is the implementation of the Sequential Dependence Model (SDM) which is a special case of the Marcov Random Field (MRF) model. Three features are considered: Single Term Feature (standard unigram language model feature), Exact Phrase Features (words appearing in sequence) and Unordered Window Features (words appearing close together, but not necessarily ordered). The "mediated_query" has been extended by the tags extracted from the first top 10 books retrieved and ranked by SDM where the tag count exedes 3.

4 Results

Table 2 shows 2014 official results for our 6 runs. InL2 model gives fair results in comparison with other models participating at Inex SBS 2014 workshop, where this model has classified the second w.r.t the measure nDCG@10 the official evaluation measure for the workshop. The different Query Expansion approaches enhanced SDM performences. SDM model gives unsatisfactory results as compared to InL2.

Table 2. Official results at INEX 2014. The runs are ranked according to nDCG@10.

Run	nDCG@10	Recip Rank	MAP	Recall@1000
Best_Run_2014	0.142	0.275	0.107	0.426
InL2	**0.128**	**0.236**	**0.101**	**0.441**
InL2Feedback	0.114	0.230	0.094	0.434
InL2tagFeedback	0.102	0.212	0.075	0.388
SDM_Rating	0.062	0.120	0.047	0.314
SDM_Concept	0.056	0.118	0.039	0.253
SDM_Tag_Feedback	0.055	0.112	0.040	0.267

5 Conclusion

In this paper we presented our contribution for the INEX 2014 Social Book Search Track. In the 6 submitted runs, we tested 2 retrieval models (SDM for MRF and InL2 for DFR) with different Pseudo Relevance Feedback mechanisms, which deploy terms and tags. We performed also topic conceptualization for query expansion. The 4 runs in which we added other terms (tags, important terms in the pseudo relevance set and concepts), the results decreased as compared to InL2 and SDM_Ratings runs.

6 Acknowledgements

This work was supported by the French program "*Investissements d'Avenir - Développement de l'Economie Numérique*" under the project Inter-Textes #O14751-408983.

References

1. Ludovic Bonnefoy, Romain Deveaud, and Patrice Bellot. Do social information help book search? In Pamela Forner, Jussi Karlgren, and Christa Womser-Hacker, editors, *CLEF (Online Working Notes/Labs/Workshop)*, 2012.
2. Gabriella Kazai, Marijn Koolen, Jaap Kamps, Antoine Doucet, and Monica Landoni. Overview of the inex 2010 book track: Scaling up the evaluation using crowdsourcing. In Shlomo Geva, Jaap Kamps, Ralf Schenkel, and Andrew Trotman, editors, *INEX*, volume 6932 of *Lecture Notes in Computer Science*, pages 98–117. Springer, 2010.
3. Craig Macdonald, Ben He, Vassilis Plachouras, and Iadh Ounis. University of glasgow at trec 2005: Experiments in terabyte and enterprise tracks with terrier. In Ellen M. Voorhees and Lori P. Buckland, editors, *TREC*, volume Special Publication 500-266. National Institute of Standards and Technology (NIST), 2005.
4. Donald Metzler and W. Bruce Croft. Combining the language model and inference network approaches to retrieval. *Inf. Process. Manage.*, 40(5):735–750, 2004.
5. Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In Ricardo A. Baeza-Yates, Nivio Ziviani, Gary Marchionini, Alistair Moffat, and John Tait, editors, *SIGIR*, pages 472–479. ACM, 2005.
6. I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.
7. Stephen E. Robertson, C. J. van Rijsbergen, and Martin F. Porter. Probabilistic models of indexing and searching. In *SIGIR*, pages 35–56, 1980.
8. J.J. Rocchio. Relevance feedback in information retrieval. *The SMART retrieval system: experiments in automatic document processing*, pages 313–323, 1971.