Book Recommendation Using Information Retrieval Methods and Graph Analysis

Chahinez Benkoussas[1,2], Anaïs Ollagnier[1,2] and Patrice Bellot[1,2]

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

Abstract. In this paper, we present our contribution in INEX 2015 Social Book Search Track. This track aims to exploit social information (users reviews, ratings, etc...) from LibraryThing and Amazon collections. We used traditional information retrieval models, namely, InL2 and the *Sequential Dependence Model* (SDM) and tested their combination. We integrated tools from natural language processing (NLP) and approaches based on graph analysis to improve the recommendation performances.

Keywords: InL2, language model, recommender system, graph analysis, natural language processing, dependence analysis.

1 Introduction

The Social Book Search (SBS) Track [5] was introduced by INEX in 2010 with the purpose of evaluate approaches for supporting users in searching collections of books based on book metadata and associated user-generated content. As in 2014, the track 2015 includes two tasks: the suggestion task and the interactive task. As part of our work, we oriented on the suggestion task, which suggests a list of the most relevant books according to the request provided by the user.

SBS task builds on corpus of topics that consist of a set of 208 complex queries expressed in natural language made by users of LibraryThing[3] forums. And a collection of 2.8 million of real books extracted from Amazon[4] pages extended by social metadata.

In our contribution at SBS task, we tested several approaches. The first consists of combining the output of retrieval systems. Secondly, we performed topic representation by query expansion and recommend books using traditional retrieval methodology. Finally, We tested a new approach for document retrieval

[3] https://www.librarything.com/
[4] http://www.amazon.com/

based on graph analysis and exploit the PageRank algorithm for ranking documents with respect to a user's query. In the absence of manually-created hyperlinks, we use social information to create the Directed Graph of Documents (DGD) and argue that it can be treated in the same manner as hyperlink graphs.

We submitted 6 runs in which we used the reviews, the ratings attributed to books by Amazon users and PageRank for reranking.

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

This section presents brief description of retrieval models used for book recommendation and our new approach based on graph modeling.

2.1 InL2

We used InL2 model implemented in Terrier[5]. 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" [8]. InL2 signies Inverse Document Frequency model with Laplace after-effect and normalization 2.

2.2 Sequential Dependence Model

We used *Metzler* and *Croft's* Markov Random Field (MRF) model [7] to integrate multi word phrases in the query. Specially, we use the SDM (Sequential Dependence Model), which is a special case of MRF. SDM builds upon this idea by considering combinations of query terms with proximity constraints which are: 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:

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

[5] http://terrier.org/

Where feature weights are set based on the author's recommendation ($\lambda_T = 0.85$, $\lambda_O = 0.1$, $\lambda_U = 0.05$) in [3]. f_T , f_O and f_U are the log maximum likelihood estimates of query terms in document D, computed over the target collection using a Dirichlet smoothing. We applied this model to the queries using Indri[6] Query Language[7].

2.3 Combination of Retrieval Systems outputs

Combining the output of many search system, in contrast to using just a single retrieval technique, can improve the retrieval effectiveness as shown by Belkin in [1]. He combined the results of probabilistic and vector space models. In our work, we combined the results of InL2 model and SDM model. The retrieval models use different weighting schemes therefore we should normalize the scores. We used the maximum and minimum scores according to Lee's formula [6] as followed:

$$normalizedScore = \frac{oldScore - minScore}{maxScore - minScore}$$

We have shown in [2] that InL2 and SDM models have different levels of retrieval effectiveness, thus it is necessary to weight individual model scores depending on their overall performance. We used an interpolation parameter (α) varied to get the best interpolation that provides better retrieval effectiveness.

2.4 Query Expansion With Dependence Analysis

Due to the nature of the proposed queries, namely, complex and long queries expressed in natural language, we used a dependency parser to extract bigrams of words syntactically dependent. We used for this purpose Stanford tool *Stanford Dependencies*[8] [4]. We performed this analysis on fields "title", "mediated_query" and "narrative" of each topic and then extended the query with resulting dependencies. We established two types of query expansions: first, with all dependencies and second, with some selected dependencies that we have retained during frequency study. Among selected dependencies we have identified the names (8,4% dependencies present in queries) and dependencies composed of prepositions defined, in linguistic, as words that establish a logical connection between two words. Among the prepositions that we considered relevant we can identify those based on *of* (4.63% dependencies), *to* (1.27% dependencies) and *about* (1.12% dependencies). Figure 1 shows an example of a query after analysis.

[6] `http://www.lemurproject.org/indri/`
[7] `http://www.lemurproject.org/lemur/IndriQueryLanguage.php`
[8] http://nlp.stanford.edu/software/stanford-dependencies.shtml

Fig. 1. Result for parsing the narrative: 'I love alternative histories - two great ones I've enjoyed are Robert Harris's Fatherland and Kim Stanley Robinson's Years of Rice and Salt . Any other recommendations? John'

<narrative_analyze> nsubj(love-2, I-1), root(ROOT-0, love-2), amod(histories-4, alternative-3), dobj(love-2, histories-4), num(ones-8, two-6), amod(ones-8, great-7), dobj(enjoyed-11, ones-8), nsubj(Fatherland-16, ones-8), nsubj(enjoyed-11, I-9), aux(enjoyed-11, 've-10), rcmod(ones-8, enjoyed-11), cop(Fatherland-16, are-12), nn(Harris-14, Robert-13), poss(Fatherland-16, Harris-14), parataxis(love-2, Fatherland-16), nn(Robinson-20, Kim-18), nn(Robinson-20, Stanley-19), poss(Years-22, Robinson-20), parataxis(love-2, Years-22), conj_and(Fatherland-16, Years-22), prep_of(Years-22, Rice-24), parataxis(love-2, Salt-26), conj_and(Fatherland-16, Salt-26), det(recommendations-30, Any-28), amod(recommendations-30, other-29), dep(Salt-26, recommendations-30), appos(Salt-26, John-32) </narrative_analyze>

2.5 Graph Based Recommendation

We tested a new approach of book recommendation based on graphs. This approach combines the results of retrieval methodology and graph analysis. To construct the graph, we exploited a special type of similarity based on several factors. This similarity is provided by Amazon and corresponds to "Similar Products" given generally for each book. Amazon suggests books in "Similar Products" field according to there similarity to book. The degree of similarity depends of users' social information like: clicks or purchases and content-based information like book attributes (book description, book title, etc.). The exact formula used by Amazon to combine social and content based information to compute similarity is proprietary.

To perform data modeling into Directed Graph of Documents (DGD), we extracted the "Similar Products" links between documents. Once used it to enrich results from the retrieval models, in the same spirit as pseudo-relevance-feedback.

Nodes are connected with directed links, given nodes $\{A, B\} \in S$, if A points to B, B is suggested as Similar Product to A. The DGD network contains 1.645.355 nodes (89.86% of nodes are in the collection and the rest does not belong to it) and 6.582.258 relationships.

Each node in the DGD represents document (Amazon description of book), and has set of properties:

- ID: book's ISBN;
- *content* : book description that include many other properties (title, product description, author(s), users' tags, content of reviews, etc.)
- $MeanRating$: average of ratings attributed to the book
- PR : value of book PageRank

In this section we use some fixed notations. The collection of documents is denoted by C. In C, each document d has a unique ID. The set of topics is denoted by T, the set $D_{init} \subset C$ refers to the documents returned by the initial retrieval model. $StartingNode$ indicates document from D_{init} which is used as input to graph processing algorithms in the DGD. The set of documents present

in the graph is denoted by S. D_{t_i} indicates the documents retrieved for topic $t_i \in T$.

Algorithm 1 takes as inputs: D_{init} returned list of documents for each topic by the retrieval techniques described in Section 3, DGD network and parameter β which is the number of the top selected $StartingNode$ from D_{init} denoted by $D_{StartingNodes}$. We fixed β to 100 (10% of the returned list for each topic). The algorithm returns list of recommendations for each topic denoted by "D_{final}". It processes topic by topic, and extracts the list of all neighbors for each $StartingNode$. It performs mutual Shortest Paths computation between all selected $StartingNode$ in DGD. The two lists (neighbors and nodes in computed Shortest Paths) are concatenated after that all duplicated nodes are deleted. The set of documents in returned list is denoted by "D_{graph}". After that, documents in D_{graph} are added to the initial list of documents (all duplications are deleted), a new final list of retrieved documents is obtained, "D_{final}" and reranked using different reranking schemes described in the next section.

Algorithm 1 Retrieving based on DGD feedback

1: $D_{init} \leftarrow$ Retrieving Documents for each $t_i \in T$
2: **for** each $D_{t_i} \in D_{init}$ **do**
3: $\quad D_{StartingNodes} \leftarrow$ first β documents $\in D_{t_i}$
4: $\quad$ **for** each $StartingNode$ **in** $D_{StartingNodes}$ **do**
5: $\quad\quad D_{graph} \leftarrow D_{graph} + neighbors(StartingNode, DGD)$
6: $\quad\quad D_{SPnodes} \leftarrow$ **all** $D \in ShortestPath(StartingNode,\ D_{StartingNodes},\ DGD)$
7: $\quad\quad D_{graph} \leftarrow D_{graph} + D_{SPnodes}$
8: $\quad\quad$ Delete all duplications from D_{graph}
9: $\quad D_{final} \leftarrow D_{final} + (D_{t_i} + D_{graph})$
10: $\quad$ Delete all duplications from D_{final}
11: $\quad$ Rerank D_{final}

3 Runs

We submitted 6 runs for the SBS Task. We used 2 frameworks that implement retrieval models : Indri[9] and Terrier (TERabyte RetrIEveR). Porter stemmer and Bayesian smoothing with Dirichlet priors (Dirichlet prior $\mu = 1500$) are used in our experiments. We performed a preprocessing step to convert Inex SBS corpus into Trec Collection Format[10], 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:

[9] http://www.lemurproject.org/
[10] http://lab.hypotheses.org/1129

Fig. 2. Example of document in Trec 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.

We described previously our approach based on DGD. We used NetworkX[11] tool of Python to handle the graph processing. In all runs, the index is built on all fields in the book xml files

INL2_fulldep This run is based on InL2 model, for each topic we use mediated_query, group, narrative tags and we extended the topics by all syntactic dependencies.

INL2_SelectDep This run is based on InL2 model, we extended the topics by selected dependencies presented in section 2.4.

INL2_fdep_SDM We used first, the extended topics with syntactic dependencies and performed retrieving using INL2. Secondly, we concatenate the title and mediated_query fields and then perform retrieving with SDM model. Scores of each retrieval system are normalized and combined using interpolation parameter $\alpha = 0.8$.

INL2_SDM_Graph In this run we used the result of INL2 and SDM outputs combination. We selected the 100 top returned documents as starting points in the DGD. Then we performed neighbors computing for each starting point and collected all nodes in Shortest paths between the starting points. We integrated the resulting nodes in the first returned list of INL2 and SDM combination. To rerank the final list, we weighted the combined score of retrieval systems (INL2 and SDM) with PageRank value for each book.

INL2_fdep_Graph In this run we used the extended topics with syntactic dependencies and INL2 retrieval model. We selected the 100 top returned documents as starting points in the DGD. Then we performed neighbors computing for each starting point and collected all nodes in Shortest paths between the starting points. We integrated the resulting nodes in the first returned list by INL2. To rerank the final list, we weighted the score of retrieval system (INL2) with PageRank value for each book.

[11] https://networkx.github.io/

INL2_Gph_SimJac In this run we used the extended topics with syntactic dependencies and INL2 retrieval model. We selected the 100 top returned documents as starting points in the DGD. Then we performed neighbors computing for each starting point and collected all nodes in Shortest paths between the starting points. We integrated the resulting nodes in the first returned list by INL2. To rerank the final list, we weighted the score of retrieval system (INL2) with value of Jaccard similarity computed between "title + description" of book and "title + narrative" of topic.

4 Results

Table 1 shows 2015 official results for our 6 runs. Our models presented this year show substantially similar results. The combination of retrieval models INL2 and SDM with the integration of graph processing performs the best in term of all measurements. The combination of IR models increases retrieval results compared to the use of each model individually. We can observe in the table that using Jaccard similarity to weight INL2 score for reranking books decrease the performances. This is mainly due to the difference between users' descriptions of their needs and the Amazon descriptions.

Table 1. Official results at INEX 2015. The runs are ranked according to nDCG@10.

Run	nDCG@10	Recip_Rank	MAP	Recall@1000
Best_run_2015	0.186	0.394	0.105	0.374
INL2_SDM_Graph	**0.081**	**0.183**	**0.058**	**0.401**
INL2_fdep_SDM	0.076	0.171	0.057	0.401
INL2_fdep_Graph	0.075	0.162	0.054	0.388
INL2_fulldep	0.070	0155	0.052	0.388
INL2_SelectDep	0.069	0.161	0.053	0.382
INL2_Gph_SimJac	0.069	0.158	0.052	0.393

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) and their combination, and used social links present in "SimilarProducts" field to construct a Directed Graph of Document (DGD) and use it to enrich recommendation list returned by IR systems. We performed also dependences analysis for query expansion. We showed that combining INL2 and SDM models increases retrieval effectiveness, and integrating result of graph analysis in retrieval process enhances the performances.

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. Nicholas J. Belkin, Paul B. Kantor, Edward A. Fox, and Joseph A. Shaw. Combining the evidence of multiple query representations for information retrieval. *Inf. Process. Manage.*, 31(3):431–448, 1995.
2. Chahinez Benkoussas, Hussam Hamdan, Shereen Albitar, Anaïs Ollagnier, and Patrice Bellot. Collaborative filtering for book recommandation. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 501–507, 2014.
3. 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.
4. Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure trees. In *LREC*, 2006.
5. 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.
6. Joon Ho Lee. Combining multiple evidence from different properties of weighting schemes. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, pages 180–188, New York, NY, USA, 1995. ACM.
7. 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.
8. Stephen E. Robertson, C. J. van Rijsbergen, and Martin F. Porter. Probabilistic models of indexing and searching. In *SIGIR*, pages 35–56, 1980.