# LaHC at INEX 2014: Social Book Search Track

Meriem Hafsi[1], Mathias Géry[1], Michel Beigbeder[2]

[1] Université de Lyon, F-42023, Saint-Étienne, France,
CNRS, UMR 5516, Laboratoire Hubert Curien, F-42000, Saint-Étienne, France
meriem.hafsi@etu.univ-st-etienne.fr
mathias.gery@univ-st-etienne.fr
[2] École Nationale Supérieure des Mines de Saint-Étienne
158, cours Fauriel - F 42023 Saint-Étienne, France
michel.beigbeder@emse.fr

**Abstract.** In the article, we describe our participation in the INEX 2014 Social Book Search track. We present the different approaches exploiting user social information such as reviews, tags and ratings. These social informations are assigned by users to the books. We optimize our models using the INEX Social Book Search 2013 collection and we test them on the INEX 2014 Social Book Search track.

**Keywords:** Social Information Retrieval, Recommendation, Structured Information Retrieval.

## 1 Introduction

In this article, we present the different approaches used in our participation to the INEX 2014 Social Book Search (SBS). The idea is to exploit some user generated content such as reviews and ratings to recommend some books. We have also used an approach based on the similarity between users. For the experiments, we have used the both collections INEX SBS 2013 and 2014[3]. Our goal is to improve the information retrieval process by optimizing our models with the INEX SBS 2013 collection. In the following section, we present the INEX SBS 2014 collection and data. Then, we present the models optimized on the INEX SBS 2013. Finally, we detail our official runs ans the results obtained.

## 2 Collection and Data

The collection contains 2.8 million book descriptions from Amazon, composed of 64 XML fields. Among these fields, we distinguish:

- Metadata: <book>, <isbn>, <title>, <authorid>, etc.
- Social information: <review>, <summary>, <tags>, <rating>, etc.

[3] INEX SBS: https://inex.mmci.uni-saarland.de/tracks/books/

LT User-Profiles are provided from LibraryThing (LT) in a text file containing 93,976 anonymous users. These profiles do not contain the personal information: they contain only the personal catalog of the users. Each user catalog is presented as a set of rows where each row represents the review of the user on one book with a rating and eventually some tags.

There are 680 topics in INEX 2014 SBS Track, where each topic contains five fields: <title>, <query>, <narrative>, the <group> where the topic was posted and a personal catalog <catalog> of the anonymous user who wrote the topic.

In our experiments, we use both collections, INEX SBS 2013 [1] and INEX SBS 2014. Both collections use the same set of documents (book descriptions). We use INEX SBS 2013 collection because the relevance judgments were available so it allowed the optimization of our system before the actual submission of our 2014 runs. The difference between the two collections lies in the topics and the users profiles. In 2014, we have only the personal catalog of each user, but in 2013 we have complete users profiles.

## 3 Retrieval models and their optimization with INEX SBS 2013

### 3.1 Preprocessing

The preprocessing step eliminates the fields we do not need. Each book description can contain one <reviews> field, that contains one or several reviews in <review> fields. Each <review> field is composed of <summary>, <content> and <tags> fields. In the <tags> field, we find some tags <tag>. After this preprocessing step, the collection contains the fields:

- <**docno**>: <isbn> field of the book.
- <**title**>: <title> field of the book.
- <**summary**>: Concatenation of the <summary> fields.
- <**content**>: Concatenation of the <content> fields.
- <**tags**>: Concatenation of the <tag> fields. The new field <tags> contains as many copies of the <tag> field content as indicated by the count attribute. For example, <tag count="3">moon</tag> will be written: <tags>moon moon moon</tags>.

### 3.2 Indexing and querying

We use the Terrier 3.6 Search Engine[4] which can index large XML collections. We use the default stop-words list of Terrier and the Porter Stemmer. Then, we create five book description index as follows:

- **Index-Title:** Only the <title> field of each book description, so no social information is indexed.
- **Index-Summary:** <summary> field only.
- **Index-Content:** <content> field only.

[4] Terrier: http://terrier.org/

– **Index-Tags:** <tags> field only.
– **Index-All-Fields:** The concatenation of all the fields: <title>, <summary>, <content> and <tags>.

We build four set of queries:

– **Topic-Title:** Only the <title> field of each topic.
– **Topic-Query:** Only the <query> field.
– **Topic-Title-Query:** <title> and <query> fields.
– **Topic-All-Fields:** <title>, <query> and <narrative> fields.

### 3.3 Content-Based Retrieval

First, we combine each of the four queries set with each of the five documents index, using the BM25 model [2]. We select one of the combinations and then we optimize the weight of each field in the BM25F model [3]. To evaluate the runs of the 20 combinations with BM25 and the run with BM25F, we test on 52 topics manually selected among the 386 topics of INEX SBS 2013. The selected topics are those in which the information need is based on the actual book content and not only on its usage. We evaluate our results with nDCG@1000 (shortened to nDCG in this paper) using trec_eval[5]. Evaluation results of these experiments are shown in Table 1.

**Table 1.** nDCG measures of BM25 and BM25F with different combinations of document and document fields.

| Model | Index | Topic-Title | Topic-Query | Topic-Title-Query | Topic-All-field |
|---|---|---|---|---|---|
| **BM25** | **Index-Title** | 0.0285 | 0.0377 | 0.0416 | 0.0725 |
| **BM25** | **Index-Summary** | 0.0527 | 0.0657 | 0.0692 | 0.0890 |
| **BM25** | **Index-Content** | 0.1005 | 0.1061 | 0.1263 | 0.1726 |
| **BM25** | **Index-Tags** | 0.1116 | 0.1115 | 0.1342 | 0.1628 |
| **BM25** | **Index-All-Field** | **0.1161** | **0.1296** | **0.1504** | **0.1991** |
| **BM25F** | **Index-All-Field** | - | - | **0.2132** | - |

The results show that indexing user generated content (<summary>, <content> and <tags>) improved the search results. We notice that a field based BM25 weight function (BM25F) obtains better results (0.2132) than the classical weight function BM25 (0.1504), while they index the same information. We choose to focus our experiments on the topics composed only by the <title> and <query> fields. Thus, the BM25F has been used only with one set of queries, even if the best results are obtained with queries including the <narrative> field. In the sequel, we will consider the index **BM25F Index-All-Fields** which is the most promising one.

[5] trec_eval version 9.0: http://trec.nist.gov/trec_eval/

### 3.4 Social Re-Ranking

Our goal is to experiment re-ranking methods in order to improve content-based search results obtained in the previous section. We propose four models using the books ratings ($Score_{AmazonRatings}$, $Score_{LTRatingsPop}$ and $Score_{LTRatingsRep}$) and the similarity between users ($Score_{UsersSimilarity}$).

**Amazon Books Rating based approach($Score_{AmazonRatings}$):** Some books were commented and rated by Amazon users. There are 14,042,020 ratings in the collection ranging from 0 to 5, 5 indicating the maximum rating. These ratings are distributed as shown in Table 2.

**Table 2.** Number of occurrences of book ratings.

| Rating | Occurrences | % |
|---|---|---|
| **0** | 6 | 0 % |
| **1** | 971,288 | 6.92 % |
| **2** | 804,193 | 5.73 % |
| **3** | 1,311,752 | 9.34 % |
| **4** | 2,933,483 | 20.89 % |
| **5** | 8,021,298 | 57.12 % |

We compute a score $AmazonRating(d)$ for each book $d$ using $m$ user ratings of $d$ as presented in equation 1. We define the score $Score_{AmazonRatings}(d, q)$ of a book $d$ for a query $q$ by a linear combination of $BM25F$ and $AmazonRating(d)$ scores (cf. equation 2).

$$AmazonRating(d) = \frac{\sum_{i=0}^{m} Rating_d(i)}{m} + \ln(m) \quad (1)$$

$$Score_{AmazonRatings}(d, q) = \alpha_1 BM25F(d, q) + (1 - \alpha_1) AmazonRating(d) \quad (2)$$

where $\alpha_1$ is a free parameter of our model.

**LibraryThing Books Rating based approaches ($Score_{LTRatingsPop/Rep}$):** The LibraryThing ratings range from 0 to 10. We introduce two concepts:

- Popularity: $Pop(d)$ is based on the number of times the book has been added to catalog. The more the book is reviewed, the higher is its popularity.
- Reputation: $Rep(d)$ is based on the number of times the book received a rating grater than 6. Then, the more the book is highly rated, the higher is its reputation.

$Pop(d)$ and $Rep(d)$ are obtained as follows:

$$Pop(d) = \begin{cases} 0 & \text{if } (m = 0) \\ \ln(m) & \text{if } (m \geq 1) \end{cases} \quad (3)$$

$$Rep(d) = \begin{cases} 0 & \text{if } (l = 0) \\ \ln(l) & \text{if } (l \geq 1) \end{cases} \quad (4)$$

where $m$ is the number of ratings and $l$ is the number of ratings higher than 6.

Then, we define the scores $Score_{LTRatingsPop}(d,q)$ and $Score_{LTRatingsRep}(d,q)$:

$$Score_{LTRatingsPop}(d,q) = \alpha_2 BM25F(d,q) + (1-\alpha_2)Pop(d) \quad (5)$$

$$Score_{LTRatingsRep}(d,q) = \alpha_3 BM25F(d,q) + (1-\alpha_3)Rep(d) \quad (6)$$

**Users Similarity based approach ($Score_{UsersSimilarity}$):** This approach is based on the similarity between users. The idea is that users who read liked the same books in the past are likely to like the same things in the future. So, we will recommend to the user who submit a topic, the books reviewed by similar users. For each book $d$, we calculate the score $Sim(d,q)$ according to the similarity in the following manner:

$$Sim(d,q) = \begin{cases} \max_{u_i \in Reviewers(d)} UsersSim(u_q, u_i) & \text{if } (Reviewers(d) \neq \emptyset) \\ 0 & \text{else} \end{cases} \quad (7)$$

with:

- $u_q$: The user who submit topic $q$.
- $Reviewers(d)$: Users who have reviewed $d$.
- $UsersSim(u_i, u_j)$: Similarity between the catalogs of the users $u_i$ and $u_j$ represented by two binary vectors (a component is set to one if the rating of the book is higher than 6).

The final score of each book is computed as follows:

$$Score_{UsersSimilarity}(d,q) = \alpha_4 BM25F(d,q) + (1-\alpha_4)Sim(d,q) \quad (8)$$

### 3.5 Results on INEX SBS 2013

The results of three of our social information retrieval models, presented in Table 3, show that two of the three models improve slightly the results compared to the $BM25F$ model. The free parameters $\alpha_1$,$\alpha_2$ and $\alpha_3$ have been optimized to respectively 0.5, 0.75, and 0.75. Note that our fourth model has not been optimized because INEX SBS 2013 collection does not have a large number of users profiles.

**Table 3.** Optimization results of three social information retrieval models.

| Model | Index | Topic | nDCG |
|---|---|---|---|
| BM25F | Index-All-Fields | Topic-Title-Query | 0.2132 |
| $Score_{AmazonRatings}$ | Index-All-Fields | Topic-Title-Query | 0.2129 |
| $Score_{LTRatingsPop}$ | Index-All-Fields | Topic-Title-Query | **0.2175** |
| $Score_{LTRatingsRep}$ | Index-All-Fields | Topic-Title-Query | 0.2145 |

## 4 Experiments and results on INEX SBS 2014 Track

For our participation to INEX SBS 2014 track, we built six runs by applying the models that we optimize on INEX SBS 2013 collection and the model $Score_{UsersSimilarity}$. These runs are summarized in Table 4 and their results are shown in Table 5. With

**Table 4.** Summary of submitted runs to INEX 2014 SBS Track.

| Run | Model | Index | Topic |
|---|---|---|---|
| HAFSI-324 | BM25F | Index-All-Fields | Topic-Title-Query |
| HAFSI-325 | $Score_{AmazonRatings}$ | Index-All-Fields | Topic-Title-Query |
| HAFSI-326 | BM25F | Index-All-Fields | Topic-All-Fields |
| HAFSI-328 | $Score_{LTRatingsPop}$ | Index-All-Fields | Topic-Title-Query |
| HAFSI-329 | $Score_{LTRatingsRep}$ | Index-All-Fields | Topic-Title-Query |
| HAFSI-345 | $Score_{UsersSimilarity}$ | Index-All-Fields | Topic-All-Fields |

**Table 5.** INEX 2014 SBS results.

| Run | Model | Rank(40) | nDCG@10 | MRR | MAP | R@1000 |
|---|---|---|---|---|---|---|
| HAFSI-326 | BM25F | 2 | **0.1424** | **0.2753** | **0.1070** | **0.4262** |
| HAFSI-328 | $Score_{LTRatingsPop}$ | 13 | 0.1167 | 0.2255 | 0.0879 | 0.3923 |
| HAFSI-329 | $Score_{LTRatingsRep}$ | 14 | 0.1161 | 0.2174 | 0.0866 | 0.3923 |
| HAFSI-325 | $Score_{AmazonRatings}$ | 15 | 0.1153 | 0.2139 | 0.0873 | 0.3923 |
| HAFSI-324 | BM25F | 17 | 0.1124 | 0.2136 | 0.0857 | 0.3923 |
| HAFSI-345 | $Score_{UsersSimilarity}$ | 34 | 0.0524 | 0.1125 | 0.0369 | 0.3832 |
| Baseline | - | - | 0.0178 | 0.0410 | 0.0136 | 0.1164 |

the INEX 2014 SBS track official measures (nDCG@10), our six runs are ranked as shown in Table 5. Their rank/nDCG curves are presented in figure 1. Our best run is the one that exploits the <narrative> field of the topic and uses BM25F model. Table 5 displays also the results of a post-INEX baseline obtained with the BM25 model queried with the <title> and <query> fields of the topic. The baseline is the sole run which does not use the <review> and <tag> field and their results are much worse. This was also observed with the SBS 2013 collection. There is a slight improvement in nDCG@10 and MRR when taking into account the user generated content. As in the 2013 results, taking into account the <narrative> topic field improves the results because the user information needs are sometimes better exposed.

## 5 Conclusion

In this paper, we described our participation to the INEX 2014 SBS track. We tested different approaches using social information: indexing of the book reviews and tags,

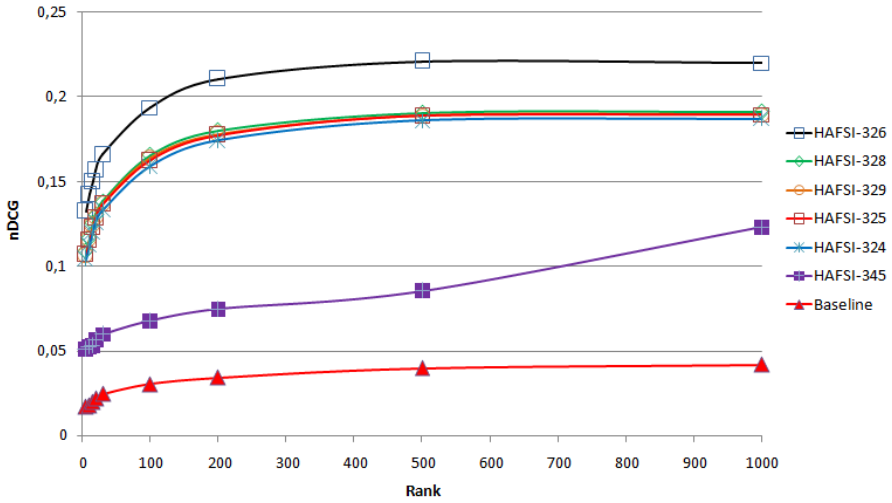


**Fig. 1.** INEX SBS 2014 results.

querying with all topic fields, using scores based on book ratings and similarity between users. These approaches give interesting results, except the approach based on the similarity between users. This is probably due to the fact that we recommend to user the whole list of books appreciated by his similar users. It would have been interesting to filter this list with another kind of social information. Also, this approach should be improved by optimizing its parameters.

## References

1. Koolen, M., Kazai, G., Preminger, M., Doucet, A.: Overview of the INEX 2013 social book search track. In: CLEF 2013 Evaluation Labs and Workshop, Online Working Notes (2013)
2. Robertson, S., Walker, S., Hancock-Beaulieu, M., Gatford, M., Payne, A.: Okapi at TREC'4. In: The Fourth Text REtrieval Conference (TREC'4). pp. 73–96. TREC-4 (1996)
3. Robertson, S., Zaragoza, H., Taylor, M.: Simple BM25 extension to multiple weighted fields. In: Conference on Information and Knowledge Management. pp. 42–49. CIKM'04, ACM, New York, NY, USA (2004)