Multimodal Social Book Search

Melanie Imhof[1,2], Ismail Badache[3], and Mohand Boughanem[3]

[1] Université de Neuchâtel, Neuchâtel, Switzerland
[2] Zurich University of Applied Sciences, Winterthur, Switzerland
imhf@zhaw.ch
[3] IRIT - Paul Sabatier University, Toulouse, France
{badache, boughanem}@irit.fr

Abstract. Today's information retrieval applications have become increasingly complex. The Social Book Search (SBS) lab at CLEF 2015 allows evaluating retrieval methods on a complex search task with several textual and non-textual meta-data fields. The challenge is to incorporate the different information types (modalities) into a single ranked list. We build a strong textual baseline and combine it with a document prior based on social signals. Further, we include non-textual modalities in relation to the user preferences using random forest learning to rank. Our experiments show that both the social document prior and the learning to rank approach improve the search results.

Keywords: Relevance feedback, random forest, non-textual modalities, social signals, document prior.

1 Introduction

The suggestion track of the INEX Social Book Search (SBS) lab at CLEF 2015 challenges researchers to find methods to retrieve books as requested by real users of LibraryThing. The complex collection consists of more than 50 meta-data fields of real books from Amazon. Thus, the retrieval methods can not rely on the content of the books but only on meta-data such as product descriptions, user-generated reviews and ratings. The lab's evaluation metric nDCG@10 reflects the user behavior that in such an application only the first few "recommendations" are considered. Hence, to maximize the number of relevant books in the first few results both the textual description of the user's query and the user's profile including his personal catalog matter. For such a complex task with that many information types, methods are required to handle and fuse them into a single ranked list. Analogously to multimedia retrieval, we call these different information types "modalities". Hence, our goal in this complex task was to fuse a strong textual baseline approach with several non-textual and social modalities that respect the user preferences. Therefore, we established and refined a textual baseline using traditional information retrieval weighting schemes, blind relevance feedback, user-profile based filtering and example book based relevance feedback. We enhanced this with document priors based on social signals such as the ratings and tags. Finally, we applied a random forest learning that further improves the results by including the non-textual modalities price and number of pages with respect to the user preferences.

2 Collection and Data

The SBS collection consists of 2.8 million book records from Amazon, extended with social meta-data from LibraryThing. Each book record is an XML file with fields like *isbn*, *title*, *review*, *summary*, *rating* and *tag*. The full list of fields is shown in Table 1.

Table 1. A list of all element names in the book descriptions.

tag name			
book	similarproducts	title	imagecategory
dimensions	tags	edition	name
reviews	isbn	dewey	role
editorialreviews	ean	creator	blurber
images	binding	review	dedication
creators	label	rating	epigraph
blurbers	listprice	authorid	firstwordsitem
dedications	manufacturer	totalvotes	lastwordsitem
epigraphs	numberofpages	helpfulvotes	quotation
firstwords	publisher	date	seriesitem
lastwords	height	summary	award
quotations	width	editorialreview	browseNode
series	length	content	character
awards	weight	source	place

There are 208 topics in the SBS 2015 lab. Each topic is a query that was posted on LibraryThing for a list of books and consists of five fields: *title, mediated_query, narrative, example* and *group*. Hereby, the *narrative* is the textual description of the query from which a hand-crafted *mediated_query* is derived. Further, the *example* field contains a list of books that the user has mentioned as positive or negative examples. Additionally, the personal LibraryThing *catalog* of each topic creator is available, which includes a list of the books the user has archived on LibraryThing along with his personal ratings.

The relevance assessments are based on the actual suggestions to the original query on the LibraryThing forum. The relevance values are weighted using a decision tree that includes reliability information such as whether the user who suggested a book has read it. The SBS 2015 topics are a subset of the topics used in 2014. However, the relevance assessments have been extended with additional book suggestions that have not been included in 2014.

3 Retrieval Models

3.1 Textual Models

As a basis for our methods we employ a textual baseline using a traditional information retrieval system. Therefore, we merge all textual fields of the document into a single textual index field. Further, we construct queries from the three topic fields *title, mediated_query* and *narrative* that are analogously merged into a single textual representation.

We extend the textual baseline with a query expansion (blind relevance feedback) based on Rocchio's method [4]. Therefore, the n most characteristic terms of the m top-ranked documents are added to the query. Hereby, the most characteristic terms of a document are chosen by the term weight determined by the weighting scheme.

As described in Section 2 the topics contain example books mentioned by the topic creators. We use the contents of the example books that are associated with a positive or neutral sentiment to expand the queries similar to the blind relevance feedback.

Additionally, we filter the books already read by the topic creator from the final ranked list, since this is a hard criterion in the relevance assessments [2]. Hereby, we determine the read books from the catalog of the topic creator as well as from the example books that are marked as read.

3.2 Social Signals-Based Model

Our approach consists of exploiting social data as a priori knowledge to take into account in the retrieval model. We combine textual relevance of a given document to a query and its social importance modeled as a prior probability.

3.2.1 Preliminaries

The social information that we exploit within the framework of our model can be represented by 3-tuple $< U, D, A >$ where *U, D* and A are finite sets of instances *Users*, *Documents* and *Actions*.

Documents. We consider a collection C=$\{D_1,\ D_2,...D_n\}$ of n documents, where each document D represents a book. We assume that a book can be represented by both a set of textual keywords D_w=$\{w_1,\ w_2,...w_y\}$ and a set of social actions A performed on the book, D_a=$\{a_1,\ a_2,...a_z\}$.

Actions. We consider a set A=$\{a_1,\ a_2,...a_m\}$ of m types of actions (signals) that users can perform on the documents. These actions represent the relation between users U=$\{u_1,\ u_2,...u_h\}$ and documents C.

3.2.2 Social Document Prior

We exploit textual models to estimate the relevance of a document to a query. Our approach combines the social document prior $P(D)$ and the relevance status value $RSV_{textual}(Q, D)$ between a query Q and document D as

$$RSV(D,Q) \overset{\text{rank}}{=} P(D) \cdot RSV_{textual}(Q,D) \tag{1}$$

$$\overset{\text{rank}}{=} P(D) \cdot \prod_{w_i \in Q} RSV_{textual}(w_i,D), \tag{2}$$

where w_i represents the terms in the query Q and $RSV_{textual}(w_i,D)$ can be estimated with different models such as BM25 and language model. The document prior $P(D)$ is a query-independent probability of seeing the document. It is useful for representing and incorporating other sources of evidence to the retrieval process. Our main contribution is a method to estimate $P(D)$ by exploiting social signals.

According to our previous approach [1], the priors are estimated by simply counting the number of actions performed on the documents. We assume that the signals are independent. Thus the general formula for calculating $P(D)$ is

$$P(D) = \prod_{a_i \in A} P(a_i), \tag{3}$$

where $P(a_i)$ is estimated using maximum-likelihood. It is calculated as

$$P(a_i) = \frac{\log(1+|D_{a_i}|)}{\log(1+|D_a|)}, \tag{4}$$

where $|D_{a_i}|$ is the number of actions of type a_i on document D and $|D_a|$ is the total number of actions on document D. Further, we use Dirichlet to smooth $P(a_i)$ by collection C to avoid zero probabilities. This leads to

$$P(D) = \prod_{a_i \in A} \left(\frac{\log(1+|D_{a_i}|) + \mu \cdot P(a_i|C)}{\log(1+|D_a|) + \mu} \right), \tag{5}$$

where $P(a_i|C)$, analogously to $P(a_i)$, is estimated using maximum-likelihood.

$$P(a_i|C) = \frac{\log(1+\sum_{D \in C}|D_{a_i}|)}{\log(1+\sum_{D \in C}|D_a|)} \tag{6}$$

In addition to considering social features separately as described above, we propose to incorporate the ratings as a measurement of the popularity and the reputation of a book. For this purpose, we use the Bayesian average (BA) of the ratings as a document prior, which takes into account how many users have rated a book. As more users rate the same book, the average becomes more reliable and less sensitive to outliers. Books that have many ratings are boosted with respect to books that have little ratings and books with high ratings are boosted more than books with low ratings. Hereby, the BA of a book is computed as

$$BA(D) = \frac{avg(D_r) \cdot |D_r| + \sum_{D' \in C} avg(D'_r) \cdot |D'_r|}{|D_r| + \sum_{D' \in C} |D'_r|}, \tag{7}$$

where *avg* is the average function and D_r is the set of ratings of document D. We note that considering logarithmic priors helps to compress the score range and thereby reduces the impact of the priors on the global score.

$$P_{BA}(D) = \frac{\log(1 + BA(D))}{\log(1 + \sum_{D' \in C} BA(D'))} \tag{8}$$

For books with no ratings this would result in a prior probability of zero. In order to avoid a multiplication by zero and thus ignoring the textual score, we use the Add-One smoothing method:

$$P_{BA}(D) = \frac{1 + \log(1 + BA(D))}{1 + \log(1 + \sum_{D' \in C} BA(D'))}. \tag{9}$$

3.3 Learning to Rank (Random Forests)

Besides the textual modalities, the SBS collection contains several non-textual modalities. We use random forests [3] to learn how to combine not only the different textual runs but also the non-textual modalities into a single ranked list. In particular, we use the price and number of pages of a book with respect to the user's preference as well as the book's ratings. Hereby, the user's preference is estimated by the average of the attributes in the topic creator's catalog; e.g. a user that only has short books in his catalog prefers short books. We assume that a user prefers to retrieve books that have similar attributes as the books he has read in the past. To achieve this, we add the difference between the average of the book prices in the topics creator's catalog and the price of the book to the random forest algorithm as an additional feature. Similarly, we add such a feature for the number of pages. For the ratings we assume that a general preference towards higher rated books exists for all users. Thus, we add the absolute average rating of a book as an additional feature to the random forests. To allow the algorithm to incorporate the significance of the average rating, we also add the number of ratings as a separate feature. The ratings are the ratings of the reviews of the book as well as the ratings in the catalogs of all topic creators. In order to combine these ratings, we divide the ratings in the catalogs by two, so that all ratings are in the same range.

4 Experimental Evaluation

We evaluated our approaches based on a series of experiments on the SBS 2015 task. Our goals in these experiments are to evaluate whether social signals (*tags* and *rating*) and other non-textual modalities can improve the search results.

4.1 Experimental Setup

For the textual baseline we used Lucene[4] for indexing and searching. We used the *EnglishAnalyzer*, which removes a small set of stopwords and stems terms

[4] https://lucene.apache.org/core/

using the Porter stemming algorithm. The weighting scheme used for most of the official runs is BM25 with $b = 0.75$ and $k_1 = 1.2$. We have also ran some experiments using language model with Dirichlet smoothing with $\mu = 2500$, however, we found that the BM25 achieved a better mean average precision (MAP) and nDCG@10 for the textual baseline. In order to validate the effectiveness of our approaches we used the topics and relevance assessments from SBS 2014.

For the blind relevance feedback, we experimented with the number of top-ranked documents used for the relevance feedback as well as with the number of terms extracted. However, we found that none of the combinations improve the textual baseline.

Since the topics from SBS 2015 are a subset of the topics from 2014, we were able to automatically add the example books from the 2015 topics to the corresponding topics in 2014. We found that expanding the queries with 35 terms extracted from the example books maximizes the nDCG@10 on the topics from 2014. Since we only have the example books for about 30% of the 2014 topics, the overall performance gain was not very big, however we have seen that the performance for the topics with example books has increased significantly.

Lucene does not provide a filter implementation that allows rejecting a list of documents, which is required to filter the read books. Thus we implemented our own filter with a similar concept as the Lucene's *FieldCacheTermsFilter*, which rejects all the documents that are not in the given list of documents.

As described in Section 3.2, we integrated social signals into the traditional textual model by re-ranking the results. The social signals are modeled as an a priori probability $P(D)$. We ran different experiments using all available social signals on the SBS collection (ratings, totalvotes, helpfulvotes, tags, etc.), but we found that the signals *tags* and *ratings*, estimated based on the formulas 5 and 9, achieved a better MAP and nDCG@10 compared to the other signals. We conducted our experiments in two ways: for Run3 and Run4 we multiplied $P(D)$ by the textual language model score; for Run5 and Run6, we combined the social signals score ($P(tags)$ multiplied by $P_{BA}(D)$) linearly with Run1, respectively with random forests trained with 100 trees. We set the smoothing parameter μ of formula 5 to 200, although more experiments will be necessary to get the best parameter. Experiments showed that the best combination parameter γ for the social score is 0.25 for Run5 and 0.2 for Run6.

We used RankLib[5] to train the random forests. For all the experiments, we left the default parameters unchanged except for the number of trees and the train metric which was set to nDCG@10. Unsurprisingly, increasing the number of trees results in a longer computation time, but also higher nDCG@10 values when training and testing on the SBS 2014 topics. However, with a higher number of trees the risk of over-fitting the data increases. The input for the random forests was built from the top 500 documents of six different textual runs together with the three non-textual modalities as described in Section 3.3. The textual runs were the textual baseline, the textual baseline with the read book filter, the textual baseline plus example based relevance feedback with and

[5] http://sourceforge.net/p/lemur/wiki/RankLib/

without filtering the read books and two runs using blind relevance feedback (total of 80 terms from 10 documents and total of 40 terms from 5 documents). Even though the blind relevance feedback runs on their own did not improve the textual baseline, we decided to add two runs using different parameters to the random forest in order to increase the variance of the input ranked lists. As training data we used the SBS 2014 topics and relevance assessments with the example books added from the 2015 topics. This is not an ideal situation, since the training data and the test data have an overlap. However, since we do not have example books for all the 2014 topics, we were not able to exclude the topics which are also in 2015 without losing the benefit of our example based relevance feedback.

For our participation to INEX SBS 2015 track, we built six runs by applying different configurations:

- **Run1**: Textual baseline using BM25 with example based relevance feedback using 35 terms and read book filtering.
- **Run2**: Random forests trained with 10 trees based on six textual runs and three non-textual modalities (price, number of pages and ratings).
- **Run3**: Run1 using language model combined with Bayesian average re-ranking based on *ratings*.
- **Run4**: Run1 using language model combined with re-ranking based on the *tags*.
- **Run5**: Run1 combined with re-ranking based on the *tags* and Bayesian average of *ratings*.
- **Run6**: Random forests trained with 100 trees based on six textual runs and three non-textual modalities (price, number of pages and ratings) combined with re-ranking based on the *tags* and Bayesian average of *ratings*.

In the next section we discuss the evaluation results of our official submission.

4.2 Results and Discussion

Table 2 summarizes our official results of SBS 2015 evaluated using nDCG@10 (Normalized Discounted Cumulative Gain), MRR (Mean Reciprocal Rank), MAP (Mean Average Precision) and R@1000 (Recall), whereas nDCG@10 is the official evaluation measure.

Table 2. Official results at SBS 2015. The runs are ranked according to nDCG@10.

Rank	Run	nDCG@10	MRR	MAP	R@1000	Train
1	Run6	0.186	0.394	0.105	0.374	yes
3	Run2	0.130	0.290	0.074	0.374	yes
8	Run5	0.095	0.235	0.062	0.374	no
10	Run3	0.094	0.237	0.062	0.374	no
11	Run4	0.094	0.232	0.061	0.375	no
21	Run1	0.082	0.189	0.054	0.375	no

We can see that the runs (Run2 and Run6) using random forest training far exceed the effectiveness of the runs using no training. During our experiments we saw that including the three non-textual modalities in the learning helps to increase the nDCG@10, which means that these modalities contain relevant information regarding the book suggestions.

Our textual baseline, although not submitted, achieves an nDCG@10 of 0.0768. Thus, the filtering together with the example based relevance feedback (Run1) significantly improves the nDCG@10 by 6.7% with a significance level of 58.4% calculated using the significance paired randomization test [5].

According to our experiments, Run3 and Run4 improve Run1 with language model (nDCG@10 of 0.0834) significantly (significance level $\alpha = 18.4\%$, respectively $\alpha = 15.3\%$). Using both the ratings and the tags (Run5) improves the effectiveness more than just using one of them. We note that the Run3 provides slightly better results in terms of MRR and MAP compared to Run4. One of the reasons of this is that the signal (rating) for Run3 that quantifies the reputation may be seen as expressing the engagement of a user who provides his explicit endorsement. For example, the document having more positive signals (ratings, likes, etc.) are more trustworthy than the ones that do not possess these social signals. If multiple users have found that the document is useful, then it is more likely that other users will find this document useful too. The social signals that quantify the popularity (number of reviews, tags, etc.) do not represent approval votes, as for example the reviews can be positive or negative, but they represent trend factors and a measure of information propagation. Therefore, a popular information always arouses the interest of the user.

The R@1000 is approximately the same for all runs, since they mostly are based on a re-ranking of Run1, for which we only retrieved the top 1000 documents. Since the learning based runs only used slight variations of Run1, they do not retrieve additional relevant documents beyond the top 1000 documents of Run1. For a recall-centric application, using a higher variety of runs as well as more documents per run would be beneficial.

5 Conclusions

In this paper, we described our participation to the suggestion track of the INEX SBS 2015 lab. We showed how to build a textual baseline and how to improve this using blind relevance feedback as well as example book based relevance feedback. Further, we proposed a method to include the social signals as a priori social knowledge that further enhanced the effectiveness of our system. The learning based approach using random forests, allowed us to incorporate the user preferences with respect to the book price and the number of pages as well as to combine the best aspects of the different variations of our textual methods.

So far, we did not use the anonymized user profiles from LibraryThing which would allow us to add additional ratings to the social model. Also we would like to test our learning approach with completely separated training and test datasets. Hence, we need to extract the example books for all the topics of SBS

2014. As a long term goal however, we think it is important to find methods that do not rely on learning. Although it might help to develop these by investigating the output of the random forests in order to better understand the modalities including their importance and their dependencies.

References

1. Badache, I., Boughanem, M.: Social priors to estimate relevance of a resource. In: IIiX Conference. pp. 106–114. IIiX'14, ACM, NY, USA (2014), http://doi.acm.org/10.1145/2637002.2637016
2. Bogers, T., Koolen, M., Jaap, K., Kazai, G., Preminger, M.: Overview of the inex 2014 social book search track. In: Conference and Labs of the Evaluation Forum. pp. 462–479 (2014)
3. Breiman, L.: Random forests. Machine learning 45(1), 5–32 (2001)
4. Rocchio, J.J.: Relevance feedback in information retrieval. In: The SMART Retrieval System: Experiments in Automatic Document Processing. pp. 313–323. Prentice-Hall, Englewood Cliffs NJ (1971)
5. Smucker, M.D., Allan, J., Carterette, B.: A comparison of statistical significance tests for information retrieval evaluation. In: CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. pp. 623–632. ACM, New York, NY, USA (2007)