Building Your Own Reading List Anytime via Embedding Relevance, Quality, Timeliness and Diversity

Bo-Wen Zhang, Xu-Cheng Yin*, Fang Zhou* and Jian-Lin Jin
bowenzhang@xs.ustb.edu.cn,xuchengyin@ustb.edu.cn
zhoufang@ies.ustb.edu.cn,jinjianlin@live.com
Department of Computer Science, School of Computer and Communication Engineering
University of Science and Technology Beijing, Beijing 100083, China

ABSTRACT

During every summer holidays, several editions of reading lists are recommended and emerged on mass media, e.g., New York Times, and BBC. However, these reading lists are built for whole people with general topics for some purposes. What if we expect the books of a specific topic at a specific moment? How to generate the requested reading list for our own automatically? In this paper, we propose a searching framework for building a topical reading list anytime, where the *Relevance* (between topics and books), *Quality* (of books), *Timeliness* (of popularities) and *Diversity* (of results) are embedded into vector representations respectively based on user-generated contents and statistics on social media. We collected 8,197 real-world topics from 198 diverse groups on *Librarything.com*. The proposed methods are evaluated on the topic collection and the public benchmarks *Social Book Search 2012-2016 (SBS)*. Experimental results demonstrate the robustness and effectiveness of our framework.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Language models*; *Novelty in information retrieval*;

KEYWORDS

Reading List; Relevance Embedding; Quality Embedding; Timeliness Embedding; Diversity Embedding

1 INTRODUCTION

In today's technology-driven world, most people should make time in their hectic schedules to read great literatures. Meanwhile, some famous media, (e.g. American Library Association,New York Times, and BBC) or some famous people, (e.g. Bill Gates) usually recommend and generate some reading lists for general readers, which aims to highlight outstanding books from the genres that merit special attention by the general public from massive available books.

However, these reading lists are not adaptable to real-world book requests in many cases. Usually the reading interests vary from one person to another, even change temporarily from time to time. *LibrartyThing.com* (LT) [9] provides users a forum to discuss their topical book requests, where users typically describe what they are looking for, give some examples of what they like or not. Others post their recommendations due to their reading experiences. Through the interactions, one can have own reading list based on the recommendations. However, the recommendation process cannot be real-time. The topics would remain unresolved unless someone concerns. Moreover, the quality cannot be guaranteed because some specific topics which require professional knowledge. Consequentially, it is expected to construct an automatic way to build a topical reading list.

Very few research has been conducted on this task. Jardine [4] studied the task of generating reading lists for novices in a scientific field. He proposed to combine Latent Topic Models with Personalized PageRank and Age Adjustment to generate reading lists of scientific papers. Ekstrand et al. [3] explored several methods for augmenting existing collaborative and content-based filtering to build research reading lists. These two approaches are both focusing on scientific papers, which contains large amount of text. For book collections, only some professional metadata, including the title, or some other publication information, is provided.

Social media opened up new horizons for book recommendations. On social media, book descriptions also contain a wealth of user-generated social contents (e.g., ratings, tags and reviews) that come from the web-described properties, contents and attributes [11]. A public track named *Social Book Search* aims at searching books from social media, e.g. Amazon.com and the participants manage to utilize social information appropriately to promote the performances. The Bellot et al. [1] focused on the ratings and reviews of books and conducted a weighting function with the number of reviews and ratings for the initial ranking score. Furthermore, Koolen et al. [8] found that indexing with useful types of social information can help to improve search accuracy. These approaches mainly focus on the relevance, which might ignores other factors, e.g. popularity, overlapping.

In this paper, we propose a searching framework for generating a topical reading list through embedding various data resources from social media to represent four designed useful

* Xu-Cheng Yin and Fang Zhou are corresponding authors.

SIGIR '17, August 07–11, 2017, Shinjuku, Tokyo, Japan.

ACM ISBN 978-1-4503-5022-8/17/08…$15.00
http://dx.doi.org/10.1145/3077136.3080734

metrics: *Relevance*, *Quality*, *Timeliness*, and *Diversity*. Specifically, the *Relevance* stands for the relevance between topics and recommended books, which can be computed through the similarities between the representations of the topics and the book contents. Moreover, the *Quality* reflects the opinions of users on the recommended books, which can be accessed through the historic statistics of the selected books. On social media, a rating is the user's assessment over books in terms of quality while the reviews have various forms of comments and discussions in which a book is evaluated. As a result, the ratings and the historic statistics of browsing, reviewing, reading over the whole website can be used to assess the quality. Different from the *Quality* describing the popularity of the books, the *Timeliness* is designed to visualize the popularity trend at the time posting the requests, which the statistics of the most popular books or collections can be used. Finally, the *Diversity* of the recommended books is a metric to offer a better balanced reading list to prevent overlapping among the books. In this paper, we adopt an approach to embed the similarities between the results, to minimize the redundancy of the results. The embeddings of four designed metrics are finally fed into classifiers to determine whether to recommend a book or not.

We collected 8,197 real-world topics from 198 diverse groups which contains history, literature, fiction and other various interest groups on *Librarything.com*. The proposed framework is evaluated through extensive experiments on the topic collection as well as a 5-year (2012-16) Social Book Search public benchmarks. Experimental results show the great performances of our framework on all datasets compared with some state-of-the-art systems.

2 TASK DESCRIPTION

The task is to recommend a reading list matching the user's book request descriptions (known as a specific topic) posted on the LT discussion forum. The user requests may vary from asking for books on a particular genre, or books written in a certain period. Two example topics from LT forum are shown in Table 1, where the field *Title* is the general description headings, while *Narrative* contains the first message of recommendation requests and *Group* simply shows the name of the discussion group. Additionally, the date when the topic was posted is also provided for the consideration of timeliness.

The candidate books are the collection of 2.78 million book records which consist of professional metadata extracted from Amazon Books enriched with user-generated contents from LT. Professional metadata contains title, isbn, publisher, sometimes table of contents while user-generated contents contains similar-products (together-bought books), tags (labels tagged by users), reviews, ratings, browse-Nodes (a hierarchy of nodes to organize for sale). In addition, some statistics generated by users indirectly are also contained like some popularity statistics counted by month, specifically how many members added, reviewed, recommend the book at each month.

Table 1: Two example topics from LT forum.

Topic 1: **Title**: The Best Peace Corps Novel **Time**: Aug 20, 2006 **Group**: Returned Peace Corps Readers **Narrative**: I'm looking for people's concept of what is the best novel for the Peace CorpsVolunteer - pre, during, or post service. This could be a novel that typifies life in the country of service...
Topic 2: **Title**: Hugh's take on 2017 **Time**: Jan 1, 2017 **Group**: The Green Dragon **Narrative**: A brand new year and a brand new thread, hopefully to be filled with books, CHEESE, pictures and any musings that take one's fancy. So let me start by wishing one and all a Happy New Year!

3 METHODOLOGY

The whole architecture to generate a reading list is presented in Fig. 1. Firstly keywords are extracted from topics to generate appropriate queries for search engine to achieve a baseline ranking result. Afterwards, we embed the four designed factors (the *Relevance*, *Quality*, *Timeliness* and *Diversity*) into vector representations for each topic-book pair, which is claimed as our proposed framework. Finally, *softmax* is applied to learn to classify the candidate topic-book pairs to build the reading list. In the following, the embedding process of the four factors are introduced in detail.

3.1 Relevance Embedding

The most important factor to build a topical reading list is to match books whose contents are relevant to the topics, namely *Relevance*. The basic idea to embed the relevance is to learn the semantic similarity of topic-book pairs with sentence models. In the following, we first describe the sentence model for mapping topics and books to their intermediate representations and then describe how to learn semantic matching between them.

3.1.1 Sentence Model. We follow the approach of [10] composed of a single wide convolutional layer followed by a non-linearity and simple max pooling, shown in Fig. 2. The input to the network are the sequential words appeared in topics and books that needs to be translated into real-valued feature vectors processed by subsequent layers of the network.

The sentences appeared in one specific field in a topic or a book (e.g. a *review* for a book) are treated as a sequence of words: $[w_1, w_2, \cdots, w_{|s|}]$ from the vocabulary list. Words are represented by distributional vectors $\mathbf{w} \in \mathbb{R}^d$, which can be looked up from pre-trained word vectors. These vectors are concatenated together into a sentence matrix $\mathbf{S} \in^{d \times |S|}$. To learn the semantics of the whole sentences, the convolutional layer is utilized to extract patterns in training instances. We use a filter $f \in \mathbb{R}^m$ (m stands for the size of f) on $\mathbf{S}$ through the convolution operation. As shown in Fig. 2, it slides along the column dimension of $\mathbf{S}$ to produce a convonlution feature map, in the form of a matrix $C \in \mathbb{R}^{d \times |s|} - m + 1$. With several filters with different sizes, a set of multiple feature maps are generated in the convolutional layer.

The output from the convolutional layer are passed through the pooling layer in order to aggregate the information and reduce the representation. Several descriptions in a book (e.g. review) or a topic (e.g. narrative) contains a wealth of useless

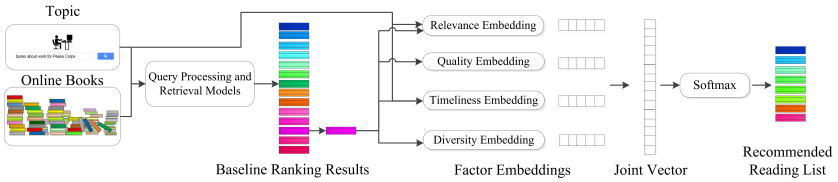

Figure 1: The whole architecture of building a topical reading list.

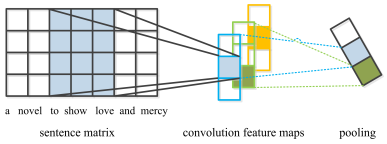

Figure 2: The sentence model to mapping text descriptions to semantic vector representations.

contents, so the dynamic k-max pooling proposed in [5] is applied as the pooling function, where the value of k depends on the input size.

Through the sentence model, rich feature representations of some text descriptions can be built to capture the overall semantics of the input sentences.

3.1.2 Matching Topics and Books. As describe in Sec. 2, topics are made up with title, group and narrative while books consist of title, tags and reviews. We make a hypothesis that the text descriptions appeared in a same field are indivisible. Therefore, the title text, the group text and all sentences in narrative text are respectively fed into the sentence model to learn the vector representations. Similarly, when embedding the book contents, the book title, the tags and the reviews are separate input of the sentence model. It is worth mentioned that for the tags, the number of users annotating the tag is multiplied as a weight of convolution feature maps before the pooling layer. Finally, the resulting vector representations of topics and books $\mathbf{x}_t$ and $\mathbf{x}_b$ can be used for computing the similarity score with Eq. (1),

$$sim(\mathbf{x}_t, \mathbf{x}_b) = \mathbf{x}_t^T M \mathbf{x}_d \tag{1}$$

where M is a parameter optimized while training, which is introduced detailedly in Sec. 3.5.

3.2 Quality Embedding

Quality is a factor which is critical for every recommendation tasks. Even though a book might be highly relevant with a topic, we cannot recommend the book if written poorly. Compared to *Relevance*, this factor only concerns the books. We categorize the statistics of books on social media into two groups: ratings and popularities throughout the time. We compute the Bayesian Average Ratings of each month through Eq. (2),

$$BA(B) = \frac{\hat{n} \cdot \hat{m} + \sum_{r \in R_B} r}{n + \hat{n}} \tag{2}$$

where R_B is the set of ratings of book B, $\hat{m}$ is the unweighed arithmetic mean value of ratings over all the books. And n represents the number of ratings, while $\hat{n}$ is the average of n.

We respectively collect the statistics of different user actions (like how many people added the book into catalog *add*, how many people reviewed *rev*, how many people recommend *rec*) monthly and directly utilize the normalization score as feature vectors. As a result, (BA, add, rev, rec) can be regarded as a monthly quality embedding and we can also use the convolutional neural networks mentioned in Sec. 3.1.1 to generate the vector representation for *Quality*.

3.3 Timeliness Embedding

The reading interests changes from time to time, thus *Timeliness* is another factor to represent the differences among popularity of books. Compared to *Quality*, this factor concerns more about the popularity trends, like what genres of books are more popular at that time. Therefore, the simplest way to embed the timeliness is to compare with the most popular books. We can achieve the vector representations of books from Sec. and the similarities between two books B_i and B_j can be easily computed through Eq. 3.

$$sim(B_i, B_j) = cos\langle vec_i, vec_j \rangle \tag{3}$$

where vec_i stands for the vector representation for book B_i and $\langle vec_i, vec_j \rangle$ is the angle between the two vectors. In this way, the *Timeliness* can be represented by the similarities between the book and top K popular books at the time when the topic is posted.

3.4 Diversity Embedding

Diversity is another significant issue to be addressed when building reading lists because no duplicated or overlapped results are expected. The general goal of diversification is to balance between the good performances of individual results based on the above three factors and the overlapping among the results. Therefore, we can compute the similarities with the other candidate books to map the *Diversity* into vectors.

3.5 Training

The model is trained to minimize the cross-entropy function:

$$cost = -\sum_{i=1}^{N} [y_i \log \alpha_i + (1 - y_i) \log(1 - \alpha_i)] + \lambda ||\theta||_2^2 \tag{4}$$

where α is the output of the softmax layer and θ contains all the parameters. The parameters are optimized with stochastic gradient descent (SGD) using back-propagation algorithm to compute the gradients.

4 EXPERIMENTS

We collect 8,197 topics from 198 diverse groups on LT forum, which are roughly divided into Simple (very specific groups like "Writer's Brag and Rag Bag", 2459 topics), Normal (groups concerning about some topic like "Weird Fiction", 4810 topics) and Hard (very challenging group like "50 Book Challenge", 928 topics). The ground truth sets are constructed by the actual suggestions from other users.

We also evaluate the performances on the Social Book Search (SBS) 2012-16 datasets. These benchmarks are also collected from LT. However, the benchmarks merely focus on the simple topics and the importance of the suggestions are defined by the relevance values based on the assessments of the topic creators and other members.

4.1 Comparisons with Components

In order to deeply understand the effectiveness of the four embedded factors, we respectively present the performances with all four factors represented as input features, namely *RQTD*, and seven baselines which contain some parts, namely, *R*, *RD*, *RT*, *RQ*, *RTD*, *RQD* and *RQT* in Table 2.

The results show that the method with fully input vectors performs better than all the other baselines across all the benchmarks. On most benchmarks, the *Relevance* are the most important factor. However, for hard topics, the *Quality* and *Timeliness* performs even better than the *Relevance*. Moreover, through comparisons between with or without embedding one factor, we can conclude that the four designed factors are all beneficial for the overall performances.

Table 2: Performances (NDCG@10, P@10, MRR) of components on all benchmarks.

Dataset	RQTD	R	RD	RT	RQ	RTD	RQD	RQT
SBS12	**0.233**	0.150	0.161	0.191	0.197	0.200	0.207	0.210
	0.169	0.113	0.121	0.122	0.134	0.140	0.141	0.155
	0.456	0.312	0.328	0.348	0.392	0.405	0.415	0.426
SBS13	**0.186**	0.143	0.144	0.157	0.164	0.168	0.169	0.170
	0.104	0.076	0.083	0.083	0.091	0.092	0.0970	0.0977
	0.284	0.215	0.221	0.232	0.246	0.2553	0.2581	0.2633
SBS14	**0.196**	0.146	0.164	0.165	0.165	0.166	0.171	0.173
	0.324	0.279	0.281	0.284	0.286	0.290	0.295	0.295
	0.138	0.102	0.104	0.105	0.105	0.120	0.122	0.125
SBS15	**0.204**	0.178	0.178	0.178	0.185	0.197	0.190	0.194
	0.426	0.376	0.377	0.386	0.386	0.389	0.400	0.401
	0.125	0.100	0.106	0.108	0.109	0.109	0.113	0.114
SBS16	**0.216**	0.168	0.170	0.171	0.181	0.182	0.186	0.195
	0.525	0.393	0.405	0.426	0.442	0.465	0.477	0.481
	0.126	0.088	0.091	0.092	0.098	0.101	0.112	0.118
Simple	**0.498**	0.435	0.454	0.463	0.465	0.470	0.481	0.483
	0.642	0.567	0.573	0.578	0.585	0.590	0.594	0.598
	0.396	0.363	0.367	0.368	0.369	0.374	0.379	0.385
Normal	**0.226**	0.178	0.178	0.178	0.185	0.197	0.190	0.194
	0.462	0.415	0.417	0.423	0.425	0.428	0.430	0.435
	0.157	0.128	0.132	0.134	0.136	0.137	0.140	0.144
Hard	**0.118**	0.058	0.060	0.072	0.083	0.092	0.096	0.098
	0.402	0.274	0.286	0.305	0.328	0.333	0.357	0.464
	0.096	0.055	0.063	0.064	0.062	0.078	0.081	0.084

4.2 Comparisons with SBS Participants

We also perform experiments our proposed framework on SBS 2012-2016 benchmarks, compared with best participants, shown in Table 3. The statistics indicates that, on the five-year datasets, our framework obtains better results and improves the best participating systems to a large extent. Hence, we have proved the effectiveness and the robustness of our proposed framework as well as all the four designed factors through the experimental results on several benchmarks.

Table 3: Comparisons with SBS participants.

System	NDCG@10	P@10	MRR
our framework	**0.2325**	**0.1686**	**0.4556**
1st top team of 2012 SBS	0.1492	0.1198	0.3069
2nd top team of 2012 SBS	0.1460	0.1380	0.370
3rd top team of 2012 SBS	0.1339	0.1260	0.3410
our framework	**0.1856**	**0.1035**	**0.2834**
1st top team of 2013 SBS	0.1361	0.0653	0.2286
2nd top team of 2013 SBS	0.1331	0.0771	0.2342
3rd top team of 2013 SBS [6]	0.1150	0.0479	0.1839
our framework	**0.196**	**0.324**	**0.138**
1st top team of 2014 SBS [12]	0.165	0.298	0.106
2nd top team of 2014 SBS	0.142	0.275	0.107
3rd top team of 2014 SBS	0.128	0.236	0.101
our framework	**0.204**	**0.426**	**0.125**
1st top team of 2015 SBS	0.186	0.394	0.105
2nd top team of 2015 SBS	0.137	0.285	0.093
3rd top team of 2015 SBS	0.106	0.232	0.068
our framework	**0.2157**	**0.5247**	**0.1253**
1st top team of 2016 SBS [2]	0.1567	0.3513	0.0838
2nd top team of 2016 SBS	0.1158	0.2563	0.0563
3rd top team of 2016 SBS [7]	0.0944	0.2272	0.0548

5 CONCLUSION

This paper innovatively design four factors (Relevance, Quality, Timeliness and Diversity) and simply embed them into vector representations for building topical reading lists. However, the method to embed some factors are simple and raw. We will continue to enhance the representation approaches.

ACKNOWLEDGMENTS

The research was partly supported by National Natural Science Foundation of China (61473036).

REFERENCES

[1] Ludovic Bonnefoy, Romain Deveaud, and Patrice Bellot. 2012. Do Social Information Help Book Search?. In *CLEF 2012*.
[2] Messaoud Chaa, Omar Nouali, and Patrice Bellot. 2016. Verbose Query Reduction by Learning to Rank for Social Book Search Track. In *CLEF 2016*. 1072–1078.
[3] Michael D. Ekstrand, Praveen Kannan, James A. Stemper, John T. Butler, Joseph A. Konstan, and John Riedl. 2010. Automatically building research reading lists. In *RecSys 2010*. 159–166.
[4] James Gregory Jardine. *Automatically generating reading lists*. Ph.D. Dissertation.
[5] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint* (2014).
[6] Marijn Koolen and Gabriella Kazai et al. 2012. Overview of the INEX 2012 Social Book Search Track. In *CLEF 2012*.
[7] Marijn Koolen and Toine Bogers et al. 2016. Overview of the CLEF 2016 Social Book Search Lab. In *CLEF 2016*. 351–370.
[8] Marijn Koolen, Hugo C. Huurdeman, and Jaap Kamps. 2013. Comparing Topic Representations for Social Book Search. In *CLEF 2013*.
[9] Roser Marfany. 2009. LibraryThing. *Item Revista De Biblioteconomia I Documentaci* (2009).
[10] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR 2015*. ACM, 373–382.
[11] X. C. Yin, B. W. Zhang, X. P. Cui, J. Qu, B. Geng, F. Zhou, L. Song, and H. W. Hao. 2016. ISART: A Generic Framework for Searching Books with Social Information. *Plos One* 11, 2 (2016).
[12] Bo-Wen Zhang and Xu-Cheng Yin et al. 2014. Social Book Search Reranking with Generalized Content-Based Filtering. In *CIKM 2014*. 361–370.