BBookX: An Automatic Book Creation Framework

Chen Liang[‡], Shuting Wang[†], Zhaohui Wu[†], Kyle Williams[‡], Bart Pursel[‡*],
Benjamin Brautigam[*], Sherwyn Saul[*], Hannah Williams[*], Kyle Bowen[*],C. Lee Giles[‡†]
[‡]Information Sciences and Technology
[†]Computer Science and Engineering
[*]Teaching and Learning with Technology
The Pennsylvania State University, University Park, PA 16802, USA
cul226@ist.psu.edu, sxw327@cse.psu.edu,
{zzw109,kwilliams,bkp10,bjb40,sps20,hrw115,kbowen}@psu.edu, giles@ist.psu.edu

ABSTRACT

As more educational resources become available online, it is possible to acquire more up-to-date knowledge and information. We propose BBookX, a novel computer facilitated system that automatically and collaboratively builds free open online books using publicly available educational resources such as Wikipedia. BBookX has two separate components: one creates an open version of existing books by linking different book chapters to Wikipedia articles, while another with an interactive user interface supports interactive real-time book creation where users are allowed to modify a generated book from explicit feedback.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.7.m [**Document and Text Processing**]: Miscellaneous

Keywords

Automatic book creation; personalization; open educational resources

1. INTRODUCTION

In an era of ubiquitous learning where information is rapidly increasing and changing, creating a contemporary personalized book has many advantages. To facilitate learning in such an environment, customized and personalized educational options and learning resources have been proposed, including MOOCs, Wikibooks[1], Wikiversity[2], etc. While they provide choices and flexibility in learning, creating and maintaining high-quality up-to-date learning resources on a large scale is still a challenging problem, especially for fast changing domains such as computer science. For example, the latest version of the popular book "Pattern Recognition and Machine Learning" misses much of the recent development of certain machine learning methods such as deep learning. However, much of this content is publicly available online in high-quality open formats.

To deal with this we propose the BBookX system which utilizes information retrieval techniques to intelligently harvest online resources and reorganize them in a format of a personalized learning resource similar to a textbook using open access textbooks and Wikipedia. By automatically linking book chapters to Wiki articles, BBookX is able to utilize the knowledge available on Wikipedia and create an open version of existing classic textbooks.

With an interactive user interface, BBookX supports collaborative real-time book creation whereby an open book is generated from user content and queries. An explicit relevance feedback mechanism allows user feedback to reformulate the query for additional searches.

Such a tool has many advantages for open educational resources: reduce the cost of creating and maintaining learning resources, contribute to open educational resources, and provide more up to date information for fast changing fields.

The rest of this paper is organized as follows: Section 2 introduces the system overview of BBookX; Section 3 discusses the construction of open book repository; Section 4 describes the interactive book creation; Section 5 presents related work and Section 6 concludes.

2. SYSTEM OVERVIEW

The system overview of BBookX shown in Figure 1 consists of two major components: the open book repository construction and the interactive book creation tool. The open book repository is built from two resources. First, existing online open access textbooks are collected. Second, for other textbooks, it is possible to create a Wiki-based open version by linking chapters to Wikipedia articles. Both open access books and Wiki-based open books are stored in the open book repository and indexed using Solr/Lucene[3]. The interactive book creation component allows users to specify the information of the book which they want to build using queries. The system will then retrieve a list of indexed educational resources ranked by the relevance to the query. An interactive user interface provides easy click selection and drag/drop functions allowing users to evaluate the returned resources. User feedback is utilized by an explicit relevance

[1]http://en.wikibooks.org/wiki/Main_Page
[2]http://en.wikiversity.org/wiki/Wikiversity:Main_Page
[3]http://lucene.apache.org/solr/

DocEng'15, September 8–11, 2015, Lausanne, Switzerland.

DOI: http://dx.doi.org/10.1145/2682571.2797094 .

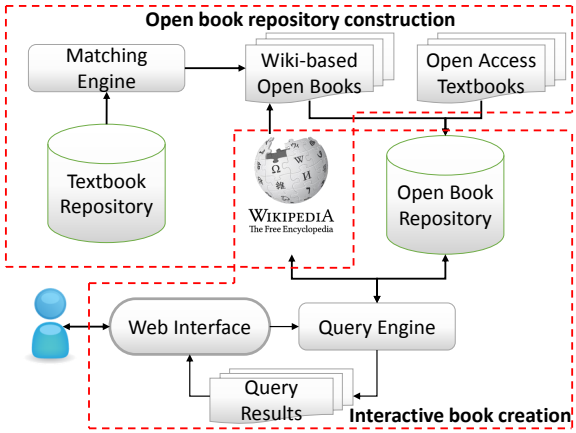


Figure 1: BBookX system overview.

feedback mechanism to reformulate the query to generate a new list of results. The generated book will be refined through such an interactive search process. Details of these two components are discussed later.

3. OPEN BOOK REPOSITORY

3.1 Collecting Open Access Textbooks

In order to construct our open book repository, we first create a collection of existing online open access textbooks by crawling different websites. Our available data sources include Wikibooks, Saylor Academy[4], MIT OCW[5], OpenStax College[6], etc. So far we have collected more than 3000 open access textbooks.

3.2 Generating Open Books Using Wikipedia

In order to provide high-quality and well-structured open books, we also utilize classic textbooks created by experts. Here we create an open version of existing books using Wikipedia. Our method is to link book chapters to Wiki articles. Specifically, for each chapter we provide users a list of Wiki articles which are most relevant to the topics covered in the book. Our method consists of three modules: concept identification, candidate selection and candidate ranking.

Concept Identification identifies the important concepts discussed in each book chapter. First, we build a domain-specific dictionary which contains the concepts related to the book topic. A depth-first search method is used to crawl Wikipedia with the seed set to be the main Wiki page related to the topic. Titles of Wiki articles visited by our crawler will be added to the concept dictionary [9]. Second, we match concepts from the dictionary in the book chapter and calculate the importance score for each concept using term frequency-inverse document frequency (tf-idf).

Candidate Selection selects the candidate Wiki articles related to the concepts in the book chapter. Two approaches are applied to collecting the candidate set. The similarity between the title of the book chapter and the title of Wiki articles is determined whereby if the Wiki title also appears in the chapter title, then the Wiki article is added to the candidate set. In addition, the similarity between the content of Wikipedia articles and that of the book chapter when each book chapter and Wikipedia article is represented as a tf-idf vector using all vector space concepts in the dictionary. The content similarity is calculated by the cosine similarity between tf-idf vectors and top N Wiki articles with high similarity are included in the candidate set.

[4]http://www.saylor.org/books/
[5]http://ocw.mit.edu/courses/online-textbooks/
[6]http://openstaxcollege.org/

Textbook	P@1	P@3	P@5	MAP@10
Computer networks	0.84	0.52	0.42	0.37
Macroeconomics	0.83	0.54	0.42	0.34
Precalculus	0.83	0.46	0.39	0.34

Table 1: Performance of the candidate ranking on three different textbooks.

3. The Application Layer	**3. The Application Layer**
3.1 Principles	3.1 Principles
3.1.1 The Peer-to-peer Model	3.1.1 Peer-to-peer
3.1.2 The Transport Services	3.1.2 Connectionless-mode Network Service
3.2 Application-level Protocols	3.2 Application-level Protocols
3.2.1 The Domain Name System	3.2.1 Domain Name System
3.2.2 Electronic Mail	3.2.2 Email
3.2.3 The HyperText Transfer Protocol	3.2.3 Hypertext Transfer Protocol
5. The Network Layer	**5. The Network Layer**
5.2 Internet Protocol	5.2 Internet Protocol
5.2.1 IP Version 4	5.2.1 IPv4
5.2.2 ICMP Version 4	5.2.2 Internet Control Message Protocol

Figure 2: Example of a Computer Networking book created using Wikipedia. The left part is the table of contents of the original book. The right part is the generated table of contents.

Candidate Ranking ranks the candidates for the most relevant Wiki articles. Using a learning to rank model, SVM^{rank} [4], different features are extracted for training including local features, such as content similarity and the Jaccard distance between titles, and global features such as redundancy features and consistency features[7]. While local features are able to capture the relatedness between the book chapter and Wiki candidates, global features are used to ensure global coherence between Wiki candidates. SVM^{rank} is tested on three textbooks from different domains. For each book chapter, three graduate students label each Wiki candidate as "relevant" or "irrelevant" with evaluation of 5-fold cross-validation on all chapters of the three books. As listed in Table 1, the performance of the proposed ranking method is consistent on three different textbooks.

Figure 2 shows a part of the book generated for an existing textbook "Computer Networking: Principles, Protocols, and Practice". The left side of the figure is the table of contents of the original book. Our method of the subsections (e.g. 3.1.1, 3.1.2) show the top candidate Wiki article for each section on the right side and link the book chapter sections to the relevant Wiki article.

3.3 Indexing Subsystem

The indexing subsystem indexes all open educational resources collected by the system, including Wikipedia articles, open access textbooks, and Wiki-based open books cre-

[7]More details of the proposed candidate ranking method can be found in our recent work[12].

ated by BBookX. A Wikipedia dump of Dec 8, 2014 is used. Each Wikipedia article and book chapter indexed is first preprocessed, which includes tokenization, stop word and punctuation removal, conversion to lower case, and stemming. Then Solr/Lucene is used to build a full text index for the content of each document. To calculate the similarity score, key phrases of each document are also extracted and indexed [11]. Specifically, anchor texts are extracted from each Wiki article as key phrases. For book chapters, the Maui tool [6] is used to extract keyphrases. Besides full text and key phrases, the metadata of each document is also indexed, such as Wiki title, book chapter title, document source, etc.

4. INTERACTIVE BOOK CREATION

Parts of the user interface for the interactive book creation tool are shown in Figure 3.

4.1 User Interface

After logging into the system, users can choose either to edit a book in "My books", which stores books previously created, or start creating a new book. The book creation process can start with creating a book title and a short description (<50 words) of the book (as shown in Figure 3a). Next users add chapters. For each chapter, a title and a short description is also created. Figure 3b shows the interface for adding chapters. The system also allows users to add subchapters or remove an existing chapter. When users finish adding chapters, they can click the "Query" button to retrieve a list of candidate items for each chapter, which could be sections of open books or Wiki articles. Users can review the returned results, reorder the list, and delete items which are not appropriate (as shown in Figure 3c). If several items under a chapter are deleted, users can regenerate the results for that chapter by clicking "Query". Explicit feedback is utilized by BBookX to reformulate the query for improved ranking. Finally, all resources that users think are relevant are combined as a book, which can be stored in "My books". In addition, the built book can be exported to HTML or text files for offline editing.

4.2 Query Subsystem

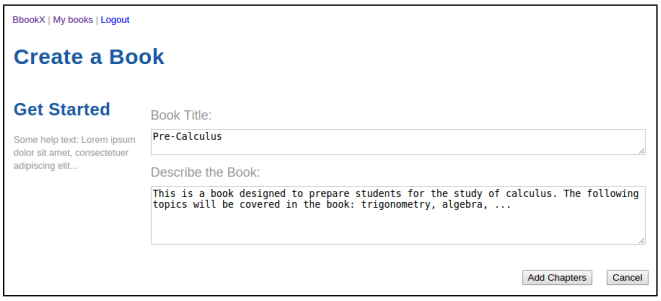
The query subsystem receives the information of a chapter/subchapter as a query and returns a ranked list of relevant educational resources, including Wiki articles and sections of open books. It mainly consists of the following three processes:

Querying: For a query $q = (t_q, c_q)$, where t_q is the chapter title and c_q is the associated descriptive text, BBookX retrieves a set of candidate relevant resources D by querying t_q and c_q in the pre-built full text index. The key phrases of q, denoted as k_q, are also extracted for the following ranking process.

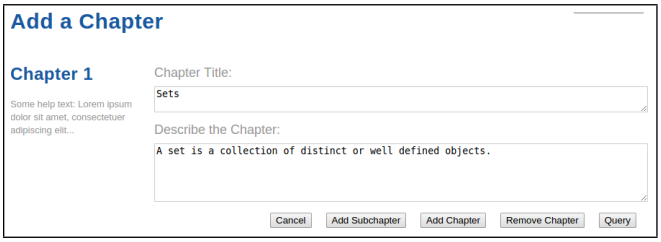
Ranking: For each candidate resource d in D, BBookX calculates the similarity score between q and d, denoted as $sim(q, d)$, by considering features similar to local features described in Section 3.2, such as title similarity and text similarity. D is sorted by similarity score in descending order as a ranked list.

In our system, $sim(q, d)$ is calculated as

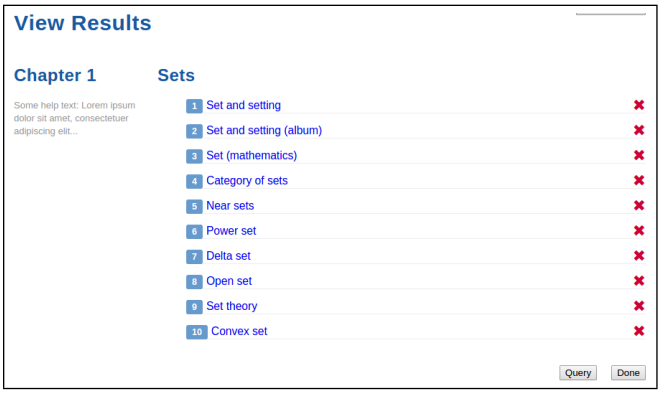
$$sim(q,d) = \alpha_1 \cdot cosSim(c_q, c_d) + \alpha_2 \cdot cosSim(k_q, k_d) + \alpha_3 \cdot Jaccard(t_q, t_d)$$



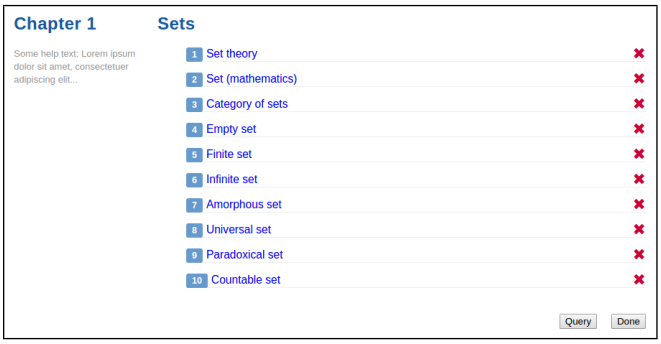
(a) Interface for creating a book title and a short description.



(b) Interface for creating chapters.



(c) Initial results for the chapter.



(d) New results using user feedback.

Figure 3: Interactive user interface.

where $cosSim(c_q, c_d)$ is the cosine similarity between the word vectors of the content of q and d, $cosSim(k_q, k_d)$ is the cosine similarity between key phrase vectors of q and d, and $Jaccard(t_q, t_d)$ is the Jaccard similarity between the title of q and d. Specifically, $\alpha_1 = 0.2$, $\alpha_2 = 0.2$, and $\alpha_3 = 0.6$.

Relevance feedback: Users can decide whether to keep a returned item or retrieve a new ranked list of items. BBookX

incorporates a relevance feedback mechanism, which includes three steps: 1) Store the results kept by users as relevant results; 2) Select the top 20 key phrases from these results using term frequency weights; 3) Perform query expansion by adding these selected key phrases to the description c_q. A new query q' is then used to retrieve relevant results. Since users usually will not write a detailed chapter description, the relevance feedback will help users to find relevant results. Of course, other methods can be used and are currently being explored.

4.3 Case Study

We use several case studies to test the performance of the proposed query subsystem. In general, BBookX can generate a satisfactory result using no more than two iterations of user feedback. Figure 3 shows an example of the creation process for the chapter "Sets" for a precalculus book. With the chapter title and descriptive text shown in Figure 3b, users first get 10 educational resources from BBookX, as shown in Figure 3c. However, some results are not relevant, such as "Set and setting" and "Set and setting (album)". If users decide to only keep the results "Set theory", "Set (mathematics)" and "Category of sets", the feedback is used to retrieve a new list of results (as shown in Figure 3d). In this case, the new returned resources are all related to "set" in mathematics.

5. RELATED WORK

FlexBook[8] is a textbook authoring platform where users can produce and customize the book content by re-purposing educational content. Wikibooks provides a wiki-based platform which allows different users to collaboratively create books [2]. To our knowledge, existing systems do not support the automatic retrieval and organization of relevant educational resources to facilitate the book creation process.

The method used in the interactive query subsytem of BBookx is related to relevance feedback [10]. Three types of feedback can be utilized: explicit feedback, which is collected by users explicitly marking relevant and irrelevant documents [13]; implicit feedback, which is inferred by the system based on users' observable behaviors such as clicks and votes [1, 5]; and pseudo feedback, which is gathered by assuming the top-k ranked documents are relevant [3]. Currently, BBookX uses only explicit feedback.

Our methods is also similar to wikification [7, 8], which automatically identifies concept mentions in text and links them to referents in Wikipedia. The difference is our focus is on extracting the most important concepts for each book chapter, not named entities mentioned in the general text.

6. CONCLUSIONS

BBookX is an automatic book creation system that assists users in creating personalized online books. by constructing an open book repository from existing open access textbooks, Wiki-based open books, and other open educational resources. We propose and evaluate methods for linking built book chapters to relevant Wiki articles. An interactive user interface supports real-time book creation from user queries which incorporates explicit relevance feedback.

To our knowledge BBookX is the first system that automatically and collaboratively creates books from open educational resources. Promising applications would be to reduce the cost of creating and maintaining learning resources and instructional content and readily incorporating rapidly changing information. New books entirely different from others can also be built. Students could also build books associated with their courses.

[8]CK-12 Foundation: http://www.ck12.org/

Acknowledgments

We gratefully acknowledge partial support from the National Science Foundation.

7. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of SIGIR*, pages 19–26, 2006.

[2] C. J. Bonk, M. M. Lee, N. Kim, and M.-F. G. Lin. The tensions of transformation in three cross-institutional wikibook projects. *The Internet and Higher Education*, pages 126–135, 2009.

[3] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using SMART: TREC 3. In *Proceedings of TREC*, pages 69–80, 1994.

[4] T. Joachims. Training linear svms in linear time. In *Proceedings of SIGKDD*, pages 217–226, 2006.

[5] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR*, pages 154–161, 2005.

[6] O. Medelyan, E. Frank, and I. H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of EMNLP*, pages 1318–1327, 2009.

[7] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of CIKM*, pages 233–242, 2007.

[8] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of CIKM*, pages 509–518, 2008.

[9] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of ACL*, pages 1375–1384, 2011.

[10] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Readings in information retrieval*, 24(5):355–363, 1997.

[11] R. Shams and R. E. Mercer. Investigating keyphrase indexing with text denoising. In *Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries*, pages 263–266, 2012.

[12] S. Wang, C. Liang, Z. Wu, K. Williams, B. Pursel, B. Brautigam, S. Saul, H. Williams, K. Bowen, and C. L. Giles. Concept hierarchy extraction from textbooks. In *Proceedings of ACM symposium on Document engineering*, 2015.

[13] R. W. White, I. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in web retrieval. In *Advances in Information Retrieval*, pages 93–109. Springer, 2002.