Using Knowledge Graph for Explainable Recommendation of External Content in Electronic Textbooks

 $\begin{array}{c} \mbox{Behnam Rahdari}^{[0000-0001-6514-912X]}, \mbox{Peter Brusilovsky}^{[0000-0002-1902-1464]}, \\ \mbox{Khushboo Thaker}^{[0000-0003-3619-9376]}, \mbox{ and Jordan} \\ \mbox{Barria-Pineda}^{[0000-0002-4961-4818]} \end{array}$

University of Pittsburgh, Pittsburgh PA 15260, USA ber58, peterb, k.thaker and jab464 @pitt.edu

Abstract. Over the last 10 years, the world experienced a rapid increase in volume and diversity of digital learning resources. The abundance of digital resources could support a range of powerful educational scenarios, which were not available before. In this paper, we introduce a novel approach that combines fully automatic knowledge modeling, student modeling, and content recommendation approaches to recommend relevant Wikipedia articles for students working with online electronic textbooks. An assessment of our approach with real classroom data indicated several benefits of our approach over the baseline and revealed interesting patterns of students' behavior while using the system.

Keywords: Recommender Systems · Personalization · Knowledge Graph · Student Model · Electronic Textbooks · Concept Extraction.

1 Introduction

Over the last 10 years, the world experienced a rapid increase in volume and diversity of digital learning resources. On the one hand, a variety of tutorials, online textbooks, educational videos, and other open educational resources were posted online to complement traditional textbooks. On the other hand, almost all traditional textbooks have migrated to digital format and become available online [6]. The abundance of digital resources could support a range of powerful educational scenarios, which were not available before. For example, if a textbook section is challenging for a learner to comprehend, she could be recommended some useful external materials, which explains the same topics in a way that is more adapted to her knowledge and interests. If the student fails to solve problems or answer questions due to the lack of prerequisite knowledge, she could be guided to the readings that introduce or review the missing knowledge.

Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

The ideas of this "smart" learning have been explored in early projects focused on adaptive textbooks [4], which demonstrated both the feasibility and the value of knowledge-driven personalized reading support. However, these early attempts focused mostly on so-called *closed corpus* personalization, i.e., guiding readers to most relevant parts of the textbook itself. A few attempts to offer *open corpus* personalization [3], i.e., recommending most relevant external resources, failed to scale up because it required expensive expert-driven knowledge analysis of every external resource [7]. The goal of the project presented in this paper was to move the idea of open corpus personalization in user-adaptive textbooks closer to reality using fully automatic knowledge modeling, student modeling, and content recommendation approaches. As a test-bed for exploring this idea, we selected the case of recommending relevant Wikipedia pages for a textbook user - both proactively, when she starts reading a new section and remedially following an attempt to answer textbook questions.

Following a brief review of related work, this paper introduces the interface of our digital textbook reading system with embedded recommendations. The next four sections introduce the underlying mechanisms of our intelligent textbook: the domain and student modeling approaches, the knowledge graph, and the recommendation approach based on this infrastructure. The following section presents the evaluation of the recommendation approach based on real classroom data. We conclude with a discussion and future work plans.

2 Related Works

Research on recommendation of related reading sources has deep roots in research on educational hypertext and hypermedia. Historically, it has been performed under the name of "intelligent hypertext", since this approach recommended resources that were not connected by a human-authored link. Research on intelligent hypertext started in the early days of the educational hypertext field and originally focused on linking resources using term-based resource similarity [10]. Simple keyword-based approaches have been gradually replaced by semantic-level similarity based on the Semantic Web ideas and domain ontology [5, 11] and, later, by modern text-processing approaches such as topic modeling and concept extraction [1, 12].

The emergence of MOOCs and the accumulation of large volume of educational content online encouraged a new wave of research on "intelligent" linking focused on connecting primary learning content such as textbooks and MOOCs with several kinds of external learning resources such as videos, Wikipedia pages, or research papers [1,9].

3 Explainable Wikipedia Recommendations in a Digital Textbook

We implemented Wikipedia recommendation interface in the context of a digital textbook system *Reading Mirror* [2]. Reading Mirror is an online reading system specifically focused on supporting student learning from modern digital textbooks (Figure 1). The system supports textbooks in PDF and HTML formats augmenting the reading process with a range of advanced features such as self-assessment, student knowledge modeling [13], and reading progress tracking with social comparison (Figure 1D).



Fig. 1. Reading Mirror interface. (A) Table of contents (B) Progress bar with mirrored social comparison interface; (C) Navigation bar; (D) Reading area; (E) Recommended Wikipedia articles

Automatic knowledge-driven linking (recommendation) of Wikipedia articles is one of the newest features of the system. A new set of five most relevant Wikipedia articles are generated for the target user in two cases. First, when a user starts reading of a new textbook unit (section or subsection), a set of best *supportive* articles is generated. These articles attempt to provide some alternative reading sources for the knowledge which the unit aims to present as well as *prerequisite* knowledge which are required to understand the content of this subsection, but not yet mastered by the target user (as evidenced by her knowledge model). Second, when the student answers a question incorrectly a set of best *remedial* articles is generated. Remedial articles focus on alternative presentation of knowledge that the student failed to master (as evidenced by the wrong answer).



(a) Recommended Item Dialog box

(b) Explanations Dialog box

Fig. 2. The preview of recommended item and explanations dialog boxes.

As shown in Figure 1E, the recommended articles are presented on the left side of the interface along with internal table-of-contents links (Figure 1A). The links are ranked by the expected value of each article (importance) in the current context. A colored heat-bar visualizes this importance: here "Green" means more relevant and "Red" means less relevant. When student clicks on a recommended item, the summary of the Wikipedia article will appear first (Figure 2a). After clicking on "Read the Full Article" button, the complete version of the Wikipedia article is being presented to the user.

To make the recommendation more transparent, we offered a brief explanation for each recommended item, which could be obtained by clicking on "(Why)" link at the right-hand side of the item. The goal of explanation is helping students to understand the reason for recommending the article. The explanation dialog (Figure 2b) consists of two parts. The first part lists top three domain model concepts that user learns by reading this article. These concepts are top three items (with highest value) in the list of "useful Knowledge" (see section 5) when this recommendation is generated. The second part explains the reason why the presented concepts are specifically important for the target user. These reasons are presented as a bullet list and are generated using the current state of the user knowledge reflected in the student model.

4 Building the Knowledge Graph

We built a graph structure to represent the underlying knowledge layer of our system. The entities and relationships in this graph demonstrate the connection between the textbook content, Wikipedia and the student model. The knowledge graph is hosted on a native graph database (Neo4j) and used for both storing the data and generating the recommendations. The overall schema of our knowledge graph is presented in Figure 3.

In the following, we will describe the process of building the knowledge graph.



Fig. 3. Graph Schema representing the entities of the knowledge graph and the relationship between them

4.1 Wikipedia Entities Representation

Wikipedia contains a large number of articles. Only a small number of them are related to the context of any given textbook. To ensure the level of relatedness and to increase the overall performance of our system we generated a subset of Wikipedia articles to be recommended to the students. In order to find the most relevant articles to the context of a textbook in the domain of computer and information science, we used Wikipedia API and started from a high-level Wikipedia category, namely "Category:Subfields of computer science" and recursively extracted the subcategories and all the articles associated with them. Since the Wikipedia category structure is not loop-free, we manually stopped the recursion after three steps. For each Wikipedia article, we extracted the following information using the Wikipedia API:

- *Title*: title of the Wikipedia page
- *Summary*: a brief description of the article that appear at the top of the page.
- Full Text: the complete textual content of the page

The total number of **1141** categories and **47772** articles are extracted and added to the graph during this step. We then connect these entities in the graph using "Has_Page" (when an article belongs to a category) and "has_Child" (when a sub-category belongs to a category) relationships.

4.2 Textbook Entities Representation

The content of the textbook is represented using three main entities: sections, questions, and concepts. For simplicity, we consider all the variation of the section (i.e, sub-sections and sub-sub-sections) as one entity (*Section*). Each section or question is associated with a set of concepts that it presents or assesses using "Includes" relationship. Each question is connected to a section with the "Belongs_to" relationship. During our calculations, we represent a union of concepts associated with a question and its corresponding section as relevant concepts to the question. Sections and questions are connected to their matched concepts via "Includes" relationship.

4.3 Linking Concepts and Wikipedia Articles

In order to create a relationship between the content of the textbook and extracted Wikipedia articles, we perform a full-text search on the textual representation of the articles using each concept as a query. The graph database (Neo4J) provided us with the full-text indexing functionality which we used to create the index for the combination of article title, summary, and full-text. To find the most relevant articles for each concept we used the Neo4J internal full-text search algorithm (Lucene). This algorithm provides us with a ranked list of relevant articles as well as a relative score that shows the relevance of each result to the input query. We used this information to connect each concept with the top 100 relevant articles alongside with their relevance score. The "Related_to" relationship is representing this connection in the graph schema.

4.4 Student Model Representation

Student models utilize a log of student actions as the input, and predict student performance with practice activities. To generate and maintain students' knowledge state for each domain model concept, we used a Comprehension Factor Analysis framework (CFM) [13]. CFM incorporates student reading behaviour along with activity performance which has proved to be beneficial in case of learning systems based on online textbooks [14]. At each student practice opportunity CFM provides the probability of student's success at that point. For our case we require probability on *each domain concept* associated with that opportunity (reading as well as questions). To generate this opportunity we generate probability of success for each concept at that opportunity (details in [15]).In the graph representation, student model maintains the level of knowledge of student with the concepts at each interaction. This information is represented by a link (called "knows") between the "user" node and "concept" node and contains the following properties:

- Interaction ID: specifies the interaction which the user gained some level of knowledge with respect to the target concept.
- Type: determines the type of activity (reading a section or answering a question) that lead to learning the concept.
- Name: stores the name of the section/question.
- *Results*: if the type is question, represents whether student answer that question correctly or not.
- Level: shows the normalized value of student's knowledge (between 0 and 1) on a given concept for a specific section or question

This implementation of the student model allows us to retrieve the students' level of familiarity with the concepts represented in a section/question after each interaction of the user with the system.

5 Recommendation Approach

Our system distinguishes two instances for recommending Wikipedia articles: (1)when student moves to a new section of the book and starts reading and (2) when students fails to answer a question at the end of the section. These instances could appear in any order: the students can move to any given section in the book at any time. Similarly, students can jump right to question section and start answering the questions before reading previous sections. Students are also able to return to a section that they previously read or a question that they already tried. To generate meaningful recommendation that could support this level of freedom (which is natural for reading a paper or electronic book) the use of student knowledge level is essential. In the following, we describe the recommendation approach for both reading and question answering instances.

In order to find the most relevant Wikipedia articles for a given reading instance, we define two overlapping sets of KCs: (1) the knowledge required to fully understand the content of the section (Required Knowledge) and (2) the current level of student mastery that has been predicted by the student model (Obtained Knowledge). The "Required Knowledge" for a given section can be defined by combining all of the concepts associated with the current and previous sections of the book. This assumption has been made based on the linear organization used in most textbooks (i.e., all the previous sections in the textbook are perquisites of the current section).

Having the set of "Required knowledge" for a given section of the textbook and the set of "Obtained Knowledge" by student while reading that section, we use set difference to form the "Useful Knowledge" set. The concepts presented in this set are the ones that are required to understand the section but has not, or only partially mastered by the student.

Since the student model predicts the level of student knowledge for each concept as a number between 0 and 1, we consider two conditions for calculating the importance of each concept in the "Useful Knowledge" set.

- MissingKnowledge: If a concept exists in "Required Knowledge" set but not in "Obtained Knowledge" set, then its important is equal to 1
- PartialKnowledge: If a concept in the "Required Knowledge" set also exist in "Obtained Knowledge" set with the predicted value of s, then its importance is equal to 1-s

As mentioned in section 4.3, we calculated the relevance of each concept to top 100 Wikipedia articles in our graph. In order to find the most relevant articles for a reading instance, we multiplied the importance of each concept in "Useful Knowledge" set by its relevance score to all Wikipedia articles connected to that concept. Then by aggregating the list for all the concepts presented in "Useful Knowledge" set over the sum of the final score, we build a ranked list of Wikipedia articles that are both relevant to a given section and take the current level of student knowledge into the account. Finally, we select the top 5 ranked articles in the list and present them as recommendation for that reading instance.

We follow the above approach with a small modifications in recommendations for question answering instances, which are generated only when the student failed to answer the question correctly. Main difference is, as mentioned in section 4.2, that the "Required Knowledge" set for a question includes not just concepts directly associated with the question, but also all "Required Knowledge" in its corresponding section.

6 The Assessment Process

To assess the potential value of our personalized recommendation approach, investigated the impact of considering the current level of knowledge represented in the student-model to generate knowledge-adaptive recommendations. This sections reviews the details of our evaluation design.

6.1 Data Source

To assess our recommendation approach in a realistic context, we used log data collected from the interaction of students with the reading system in a real semester-long course on Information Retrieval. In this course, the students were required to read 43 sections of the book and answer questions at the end of each section (75 questions in total). The log includes data of 22 students who used the reading system during this course. The students made 9494 interactions with the system (Average: 431.5, Median: 411.5, SD: 108.2). We followed these interactions reconstruct the state of their student models at every recommendation opportunity as described in Section 4.4.

6.2 Baseline

To highlight the value of using student knowledge in the recommendation process, we compare our knowledge-adaptive recommendation with a baseline that only considers the content of a given section/question to generate the recommendations. This baseline represents the current state of the art for generating recommendations of external content [1, 9]. In parallel with adaptive recommendations, we created a set of baseline recommendations for every reading or question-answering instance.

In order to find the most relevant article with respect to a given section or question, we first created a list of all articles that are related to the concepts which represent that section or question. We then aggregated that list over the sum of the scores for each concept in the list. Finally we re-rank the list based on the aggregated-sum of scores and selected the top 5 relevant article to each section or question. This connection is illustrated as "Related_to" relationship in the graph schema (Figure 3). The "relevance" property of this relationship represents the relatedness of the section/question to the target Wikipedia article.

7 Results

To determine the effectiveness of our experimental system, we investigated the following key factors: (1) To what extent the recommended items are affected by involving the student model into the calculations, (2) Whether including the student model improved the quality and coverage of the recommended items, (3) Are there any conspicuous patterns in changes caused by including student model in the process of recommendations and (4) In what ways the proposed approach can facilitate the reading process for the students.

7.1 Measure of Ranking Quality - Expected Knowledge Value

In order to compare the results of recommendations between our proposed method (combination of section/question context and the student model) and the baseline (only the context of section/question) we calculated the Discounted Cumulative Gain (DCG) [8] of every set of recommended items for each instances of user interaction with the reading system.

$$DCG = \sum_{n}^{i=1} \frac{relevance_i}{\log_2(i+1)} \tag{1}$$

The DCG equation, as it shown in 1, takes into account both the relevance score and the order of items in the recommendations list. The relevance score for each item is calculated by averaging the similarity score (using Lucene search) of all the linked concepts to a given section/question and their corresponding Wikipedia article. This relevance score is being discounted by dividing it with the log of the corresponding position.

7.2 Overall Expected Knowledge Value of the Recommendations

Figure 4 illustrate the overall quality of recommended items for sections and questions in the textbook. The x axis shows the normalized average value of Discounted Cumulative Gain for every given section (Figure 4a) and question (Figure 4b). As the data shows, the average DCG value is always higher when the student model is being involved in the process of recommendation. The proposed approach produced recommendations with in average 23.29% higher DCG value among all sections and 30.27% among the questions. The higher DCG values suggests that more concepts were engaged in the process of recommendation and the recommended items using the proposed approach have higher expected knowledge values.



Fig. 4. The Comparison Between Average DCG values of the Sections/Questions of the Textbook

It would be expected that adaptive recommendations will suggest different sections to different students at the start of the same section since their knowledge are likely to be different due to differences in reading paths. To investigate the effect of including the "student model" in the process of recommendations,



Fig. 5. The Comparison Between Average DCG values of the Sections/Questions of the Textbook

we visualized the difference between expected knowledge values (DCG) amongst all the students for every section (Figure 5a) and question (Figure 5b) in the textbook. The apparent fluctuations in expected knowledge value suggests that every student received a different set of recommendations for a given instance of interaction with the reading system. This can be considered as the evidence for the necessity of student-level personalization considering each student has a different level of mastery for each concept and required to learn divergent set of concepts at each instance of interaction.

7.3 Predicting User's Knowledge Requirements

To examine how well the personalized recommendation could help the users, we examined the "jumping-back" behavior in their reading log. The Reading Mirror system provides students with the ability to jump between sections using the textbook's table of content (Figure 1A). Frequently, this functionality is used by the students to jump back to a certain section of the textbook in order to learn or refresh their memory on a specific concepts that they need to understand the current section. Data analysis of student navigation behavior in our class shows. this jumping-back behavior was quite frequent taking at average 17.27% of all student navigation steps in the textbook. We believe that in many cases, the adaptive knowledge-based recommendation of Wikipedia articles could prevent this unproductive behavior. Unlike non-adaptive Wikipedia article recommendation (which focuses on the concepts presented in the current page), the adaptive recommendation attempts to *proactively* offers information about concepts that might be necessary to understand the current page or question, but are not yet known sufficiently by the target user. These recommendation might present the missing information right in place – eliminating the need of jumping-back behavior and helping students to integrate the past and the current knowledge.

To assess to what extent the proactively generated adaptive recommendations could help in this context, we examined each jumping-back case and recorded all concepts presented on the page that the student selected to jump back. We then compared this set of concepts with the concepts covered by adaptive recommendation of Wikipedia articles presented for the given student on the last page visited before jumping back. As the Figure 6 shows, the proactive recommendations cover a remarkable fractions of concepts that were the target of these back-jumps, 86.63% at average. This result indicates that our adaptive recommendation approach can accurately predicts the missing knowledge and considerably reduce the need of jumping through sections to acquire or refresh these knowledge. In contrast, non-adaptive baseline recommendation focused on the current page would cover less than a quarter (24.13%) of student background knowledge needs (Figure 6). This data stresses the importance of considering potentially missing background knowledge in the real classroom context and shows the value of adaptive knowledge-based recommendation.



Fig. 6. Percentage of potentially missing previous concepts targeted by jumping-back behavior that are covered by the adaptive and baseline recommendations

8 Summary and Discussion

In this paper we present a novel approach to generate personalized recommendations of external content for online electronic textbooks. We construct a knowledge graph that represents all three components of "relevant Wikipedia articles", "textbook content" and the "student model". We used this knowledge graph to generate personalized recommendations based on the relevance to a specific section/question but also taking into account the state of the student model in every instance of interaction with the reading system. The experimental evidence of our data-driven analysis shows that including the student model in the process of generating the recommendation results in higher expected knowledge value in the recommendations. Furthermore, we demonstrate that our proposed approach can accurately predicts the missing knowledge components and potentially reduce the need of jumping-back behaviour amongst students and provides a smoother reading experience. We are aware of possible limitations of our proposed approach and findings. First, the total number of students in our experimental dataset was limited; including more student data could results in more accurate conclusion. Second, despite of popularity of data-driven studies in the

field of recommender system, it has been argued that these studies should be complemented by controlled user studies where students could observe and rate generated recommendations. And finally, there are other important components such as the difficulty of learning concepts, forgetting factor, etc. that we can potentially incorporate in our recommendation algorithm to produce better and more accurate results. We hope to address these issues in our future work.

References

- Agrawal, R., Gollapudi, S., Kannan, A., Kenthapadi, K.: Study Navigator: An algorithmically generated aid for learning from electronic textbooks. Journal of Educational Data Mining 6(1), 53–75 (2014)
- Barria-Pineda, J., Brusilovsky, P., He, D.: Reading mirror: Social navigation and social comparison for electronic textbooks. In: First Workshop on Intelligent Textbooks at 20th International Conference on Artificial Intelligence in Education. pp. 30–37 (2019), http://ceur-ws.org/Vol-2384/paper03.pdf
- Brusilovsky, P., Henze, N.: Open corpus adaptive educational hypermedia. In: The Adaptive Web, pp. 671–696. Springer, Berlin (2007)
- Bursilovsky, P., Eklund, J., Schwarz, E.: Web-based education for all: A tool for developing adaptive courseware. In: 7th International World Wide Web Conference. pp. 291–300 (1998)
- Carr, L., Hall, W., Bechhofer, S., Goble, C.: Conceptual linking: Ontology-based open hypermedia. In: 10th International World Wide Web conference. pp. 334–342 (2001)
- Crestani, F., Landoni, M., Melucci, M.: Appearance and functionality of electronic books. International Journal on Digital Libraries 6(2), 192–209 (2006)
- Henze, N., Nejdl, W.: Adaptation in open corpus hypermedia. International Journal of Artificial Intelligence in Education 12(4), 325–350 (2001)
- Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems 20(4), 422–446 (2002)
- Kokkodis, M., Kannan, A., Kenthapadi, K.: Assigning educational videos at appropriate locations in textbooks. In: Seventh International Conference on Educational Data Mining. pp. 201–204 (2014)
- Mayes, J.T., et al.: Strathtutor: The development and evaluation of a learning-bybrowsing on the macintosh. Computers and Education 12, 221–229 (1988)
- Milne, D., Witten, I.H.: Learning to link with wikipedia. In: 17th ACM Conference on Information and Knowledge Management. pp. 509–518 (2008)
- Rihák, J., Pelánek, R.: Measuring similarity of educational items using data on learners' performance. In: 10th International Conference on Educational Data Mining (2017)
- Thaker, K., Carvalho, P., Koedinger, K.: Comprehension factor analysis: Modeling student's reading behaviour: Accounting for reading practice in predicting students' learning in moocs. In: 9th International Conference on Learning Analytics and Knowledge (2019)
- Thaker, K., Huang, Y., Brusilovsky, P., He, D.: Dynamic knowledge modeling with heterogeneous activities for adaptive textbooks. In: the 11th International Conference on Educational Data Mining. pp. 592–595 (2018)
- Thaker, K., Zhang, L., He, D., Brusilovsky, P.: Recommending remedial readings using student's knowledge state. In: 13th International Conference on Educational Data Mining. pp. 233–244 (2020)