Technology, Knowledge and Learning
https://doi.org/10.1007/s10758-021-09544-z

ORIGINAL RESEARCH

Knowledge Annotation for Intelligent Textbooks

Mengdi Wang[1] · Hung Chau[1] · Khushboo Thaker[1] · Peter Brusilovsky[1] · Daqing He[1]

Accepted: 22 June 2021

Abstract
With the increased popularity of electronic textbooks, there is a growing interest in developing a new generation of "intelligent textbooks," which have the ability to guide readers according to their learning goals and current knowledge. Intelligent textbooks extend regular textbooks by integrating machine-manipulable knowledge, and the most popular type of integrated knowledge is a list of relevant concepts mentioned in the textbooks. With these concepts, multiple intelligent operations, such as content linking, content recommendation, or student modeling, can be performed. However, existing automatic keyphrase extraction methods, even supervised ones, cannot deliver sufficient accuracy to be practically useful in this task. Manual annotation by experts has been demonstrated to be a preferred approach for producing high-quality labeled data for training supervised models. However, most researchers in the education domain still consider the concept annotation process as an ad-hoc activity rather than a carefully executed task, which can result in low-quality annotated data. Using the annotation of concepts for the *Introduction to Information Retrieval* textbook as a case study, this paper presents a knowledge engineering method to obtain reliable concept annotations. As demonstrated by the data we collected, the inter-annotator agreement gradually increased along with our procedure, and the concept annotations we produced led to better results in document linking and student modeling tasks. The contributions of our work include a validated knowledge engineering procedure, a codebook for technical concept annotation, and a set of concept annotations for the target textbook, which could be used as a gold standard in further intelligent textbook research.

Keywords Knowledge engineering · Concept annotation · Concept mining · Annotation scheme · Intelligent textbook · Electronic textbook

1 Introduction

Modern textbooks have been developed and refined over many decades into well-organized tools for communicating knowledge and educating the next generation of professionals. Yet, the power of computing and the internet has caused textbooks to evolve even faster than

Mengdi Wang and Hung Chau have contributed equally to this work.

✉ Peter Brusilovsky
peterb@pitt.edu

Extended author information available on the last page of the article

before. The conversion of textbooks into an electronic format has created an opportunity to augment textbooks with novel functionalities based on applications or artificial intelligence. This direction of research, which is usually referred to as “intelligent textbooks,” has explored a range of novel ideas over the last 20 years. The explored approaches include adaptive navigation support (Henze et al., 1999), natural language question answering (Chaudhri et al., 2013), automatic link creation (Guerra et al., 2013), and personalized recommendations of external content (Agrawal et al., 2014).

Concepts in the form of keyphrases play an important role in empowering most of these intelligent textbook technologies because they act as the “knowledge behind pages.” Historically, concepts were connected with each page or a section of a textbook by a textbook author or another domain expert in the process known as *indexing*, which was an important component of intelligent textbook authoring (Brusilovsky, 2003). The rapid development of machine learning and natural language processing enabled a range of *keyphrase extraction* and *keyphrase generation* approaches (Meng et al., 2017) that can perform automatic indexing of textbook fragments with domain concepts represented by keyphrases. However, these automatic keyphrase extraction methods suffer insufficient extraction accuracy in specific application domains where high-quality training data is scarce. Education domain including intelligent textbooks is such a domain.

The emergence of automatic keyphrase extraction approaches changes the nature of human engagement in knowledge annotation. While in the past, with a large number of documents to be annotated, the goal of human work was quantity rather than quality, nowadays the large share of less-critical annotation work could be performed automatically, saving human time to focus on most critical cases where quality is essential. In turn, documents with high-quality annotation become vital to train automatic keyphrase extraction approaches based on machine learning.

This paper presents our attempt to design and evaluate a reliable and systematic annotation procedure to produce high-quality knowledge annotation for college textbooks where the demand is high, quality of the outcome is important, and high-quality training data is scarce. In contrast to traditional single-author indexing, we explored a codebook-driven annotation process performed by a team of experts. The use of codebooks to assist a team of experts has been explored in several areas, however, this approach has never been used to produce concept annotations for intelligent textbooks. We evaluated the quality of the designed annotation process against several manual and automatic baselines and also explored its use for some essential needs of intelligent textbooks such as student modeling. Our study demonstrated that our approach produces high-quality annotation results that can be directly used in intelligent textbook tasks for providing the “knowledge behind pages,” or utilized for training automatic models.

This paper is organized as follows. Section 2 reviews related work; Sect. 3 describes the design of a systematic textbook annotation procedure and it’s application on an online textbook; Sect. 4 provides the main outcomes of this concept annotation procedure; Sect. 5 presents the evaluation on the main outcomes; and Sect. 6 concludes this paper and discusses the future work.

2 Related Work

2.1 Intelligent Textbooks

Much of the research on intelligent textbooks could be traced back to early attempts to develop electronic textbooks using pre-Web hypertext systems. At the time, artificial intelligence (AI) approaches were used to automate link creation between hypertext pages, which was an essential process in creating a high quality hypertext (Bareiss & Osgood, 1993). Since these early attempts, "intelligent linking" has remained an integral part of hypertext research. A range of more advanced approaches to extract concepts and other semantic features from hypertext pages have been reported in Agrawal et al. (2014), Green (1999), Guerra et al. (2013), Lakkaraju et al. (2008).

The next generation of research on intelligent textbooks was motivated by the expanding World Wide Web and the migration of textbooks online. This generation focused on using adaptive hypermedia techniques to produce adaptive textbooks. By monitoring user reading and other activities (e.g., question answering) in adaptive online textbooks, these systems attempted to model user knowledge and to support users with adaptive navigation within a book (Brusilovsky & Pesin, 1998; Henze et al., 1999; Kavcic, 2004) as well as adaptive content presentation (Melis et al., 2001). This generation of adaptive textbooks has been based on relatively advanced models of content annotation by domain experts, frequently using domain ontologies (Brusilovsky, 2003). Similar to automatic linking research, the research on concept-based adaptive textbooks remains active and focuses on more advanced personalization technologies, as well as on automative domain model development and concept indexing.

The most recent generation of intelligent textbooks was fueled by the increased availability of user data and focused on combining artificial and collective intelligence. Starting with early attempts of using users' past behaviors to provide social navigation support for future learners (Brusilovsky et al., 2004), research in this direction has explored increasingly more complex approaches for mining users' past behaviors to guide new users (Lan & Baraniuk, 2016) and predict their success (Winchell et al., 2018). Modern research on intelligent textbooks also frequently combines the ideas of automatic linking, personalization, concept annotation, and data mining (Labutov et al., 2017; Lan & Baraniuk, 2016).

2.2 Data Annotation

Despite efforts to automate knowledge extraction for documents, manual annotation processes still play an important role in the construction of corpora in various domains, including the domains of scientific publication (Augenstein et al., 2017), biomedical literature (Wilbur et al., 2006), and clinical corpus (Xia and Yetisgen-Yildiz, 2012). In the beginning of these processes, domain experts are often recruited to perform initial annotations with initial guidelines. Next, the experts will iteratively refine the guidelines until the agreement reaches a pre-defined threshold. Finally, with the refined guidelines, the experts can annotate a large-scale document collection. During the annotation process, discussions among the annotators to review and refine the guidelines have shown to have positive impacts on the quality of annotations. Wilbur et al. (2006) found that the inter-annotator agreement could significantly increase among annotators who received additional training on the guidelines. A set of such guidelines is also called a coding schema that assigns an objective (e.g., morphemes, words, phrases, sentences) to a single category. Two considerations for a coding schema were identified in Bayerl et al. (2003): (1) the categories of the coding schema must enable people to differentiate among the categories; and (2) the coding schema should be consistent among different coders or within one coder over different time. A methodological framework was also proposed in Bayerl et al. (2003), which consisted of five successive steps for systematic schema development.

2.3 Concept Mining

There are a wide range of applications related to concept mining, such as *key-phrase or concept extraction*, *prerequisite-outcome concept prediction* (Labutov et al., 2017), or *concept hierarchy creation* (Wang et al., 2015). Among these applications, concept extraction is the most fundamental task that leads to the success of other tasks; i.e., in order to predict if a concept is a prerequisite or outcome concept, we first need to identify if it is a concept.

Dozens of studies have tried to automatically extract key-phrases by using different approaches, including rules-based, supervised learning, unsupervised learning, and deep neural networks. However, their performances are still very low, which makes them not effective enough to use for certain applications, such as explainable recommendation systems. Typically, automatic key-phrase extraction systems consist of two parts. First, they need to preprocess data and then extract a list of candidate keyphrases with lexical patterns or heuristics (Florescu & Caragea, 2017; Grineva et al., 2009; Le et al., 2016; Liu et al., 2009a; Medelyan et al., 2009; Mihalcea & Tarau, 2004). Then, the candidates are ranked or classified to identify correct keyphrases by using unsupervised methods or by using supervised methods with hand-crafted features. Candidates are scored based on some properties that show how likely a candidate keyphrase is to be a keyphrase in the given document. Many studies have formed this task as a binary classification problem to determine correct keyphrases (Hulth, 2003; Jiang et al., 2009; Rose et al., 2010; Wang et al., 2015; Witten et al., 2005).

For unsupervised learning, graph-based methods (Bougouin et al., 2013; Mihalcea & Tarau, 2004) try to find important keyphrases in a document. A candidate is important when it has relationships with other candidates and when those candidates are also important in the document. This forms a graph that represents the input document, where a node and edge of the graph represents a keyphrase candidate and the relationship between two related candidates, respectively. Each node in the graph is assigned a score, which can be calculated using ranking techniques such as *PageRank* (Page et al., 1999). Finally, they select the top-ranked candidates as keyphrases for the input document. On the other hand, topic-based clustering methods (Grineva et al., 2009; Liu et al., 2009b, 2010) group semantically similar candidates in a document as *topics*. Keyphrases are then selected based on the centroid of each cluster or the importance of each topic.

Although deep neural networks have successfully been applied to many NPL-related tasks, such as sequence tagging and named entity recognition, few studies have focused on the problem of keyphrase extraction, and none of them have evaluated a textbook dataset, which resulted from a lack of a large amount of available data to train a deep learning model. Meng et al. (2017) have developed a RNN-based generative model using encoder-decoder architecture to predict keyphrases. Though their performance was better than state-of-the-art methods, it was still not clear how it would be used in an educational setting since the datasets evaluated were scientific articles and paper abstracts an author-assigned keyphrases in scientific articles are often general topics rather than specific concepts that are taught in class (e.g., concept extraction vs. expectation maximization). Chau et al. (2020) show that CopyRNN does not perform better than some of the baselines in the context of the Introduction to Information Retrieval textbook.

Wang et al. (2015) proposed a method for mining concept hierarchies for textbooks, which is also required to extract a list of concepts. In this study, instead of focusing on the concept extraction task, they used Wikipedia titles as an external resource to identify concepts appearing in the textbook's table of contents that may not cover many other concepts discussed in the book's contents. As a result, there are only a few concepts extracted to build the hierarchy.

3 Textbook Knowledge Annotation

In the education domain, knowledge annotation has been performed in many studies because its results have often served as the primary input for methods being developed (Brusilovsky et al., 1998; Weber & Brusilovsky, 2001; Henze & Nejdl, 2001; Papanikolaou et al., 2003). However, researchers usually perform it as an ad-hoc task and it is known to be a very challenging task (Shamsfard & Barforoush, 2004; Wong et al., 2012). This is because it is hard to maintain consistency during the long process of annotation without clear rules and descriptions.

In order to overcome this challenge, we designed a systematic textbook annotation procedure and applied it to annotating a popular online textbook, *Introduction to Information Retrieval* (IIR) . The goal of our annotation is to add concepts to the book to turn it into an intelligent textbook, and this annotation task helped us to refine the proposed textbook annotation procedure.

3.1 The Case Study: Introduction to Information Retrieval

The ultimate goal of this research focuses on the development of intelligent textbooks, which could offer a rich set of support functionalities to readers, including automatic content linking and recommendations. The IIR textbook was one of our first targets. In order to support the expected functionalities, we have to produce a fine-grained annotation of concepts to this textbook. Before we introduce our systematic annotation approach, it is important to mention that in order to produce quality annotation for the IIR textbook, we previously explored traditional ad-hoc expert annotation, crowdsourcing, and automatic concept extraction, as well as other approaches. While the overall quality of the obtained results and the inter-rater agreement for both experts and crowdworkers were lower than expected, the results of our earlier work were useful to guide our work on systematic annotation and were also useful to offer evaluation baselines.

3.2 Initial Coding Procedure and Hiring Process

Our goal is to develop a systematic textbook annotation procedure so that high inter-annotator agreements can be achieved and maintained. As shown in Fig. 1, the initial annotation procedure contains several standard steps, including screening applicants' profiles, guiding annotators to perform the tasks, and building an annotation codebook.

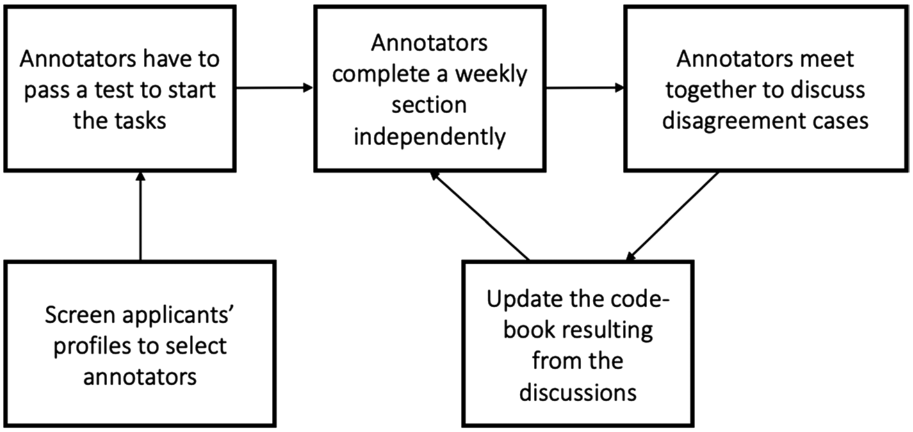
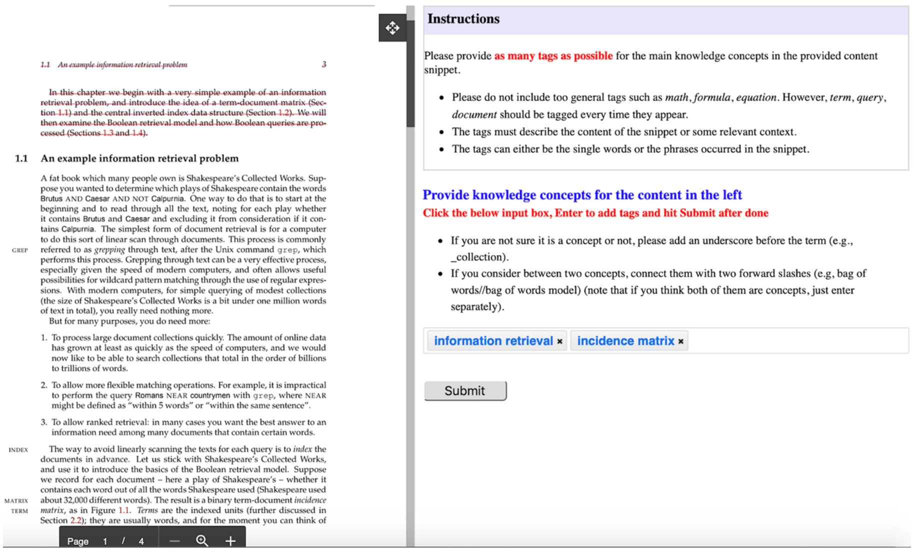

Annotators have to pass a test to start the tasks
Annotators complete a weekly section independently
Annotators meet together to discuss disagreement cases
Screen applicants' profiles to select annotators
Update the code-book resulting from the discussions

Fig. 1 Coding procedure diagram. The annotators follow the procedure until they complete the whole process



~~1.1 An example information retrieval problem~~ ~~3~~

~~In this chapter we begin with a very simple example of an information retrieval problem, and introduce the idea of a term-document matrix (Section 1.1) and the central inverted index data structure (Section 1.2). We will then examine the Boolean retrieval model and how Boolean queries are processed (Sections 1.3 and 1.4).~~

1.1 An example information retrieval problem

A fat book which many people own is Shakespeare's Collected Works. Suppose you wanted to determine which plays of Shakespeare contain the words Brutus AND Caesar AND NOT Calpurnia. One way to do that is to start at the beginning and to read through all the text, noting for each play whether it contains Brutus and Caesar and excluding it from consideration if it contains Calpurnia. The simplest form of document retrieval is for a computer to do this sort of linear scan through documents. This process is commonly referred to as *grepping* through text, after the Unix command `grep`, which performs this process. Grepping through text can be a very effective process, especially given the speed of modern computers, and often allows useful possibilities for wildcard pattern matching through the use of regular expressions. With modern computers, for simple querying of modest collections (the size of Shakespeare's Collected Works is a bit under one million words of text in total), you really need nothing more.

GREP

But for many purposes, you do need more:

1. To process large document collections quickly. The amount of online data has grown at least as quickly as the speed of computers, and we would now like to be able to search collections that total in the order of billions to trillions of words.
2. To allow more flexible matching operations. For example, it is impractical to perform the query Romans NEAR countrymen with `grep`, where NEAR might be defined as "within 5 words" or "within the same sentence".
3. To allow ranked retrieval: in many cases you want the best answer to an information need among many documents that contain certain words.

INDEX The way to avoid linearly scanning the texts for each query is to *index* the documents in advance. Let us stick with Shakespeare's Collected Works, and use it to introduce the basics of the Boolean retrieval model. Suppose we record for each document – here a play of Shakespeare's – whether it contains each word out of all the words Shakespeare used (Shakespeare used about 32,000 different words). The result is a binary term-document *incidence matrix*, as in Figure 1.1. *Terms* are the indexed units (further discussed in Section 2.2); they are usually words, and for the moment you can think of

MATRIX
TERM

Instructions

Please provide **as many tags as possible** for the main knowledge concepts in the provided content snippet.

- Please do not include too general tags such as *math*, *formula*, *equation*. However, *term*, *query*, *document* should be tagged every time they appear.
- The tags must describe the content of the snippet or some relevant context.
- The tags can either be the single words or the phrases occurred in the snippet.

Provide knowledge concepts for the content in the left

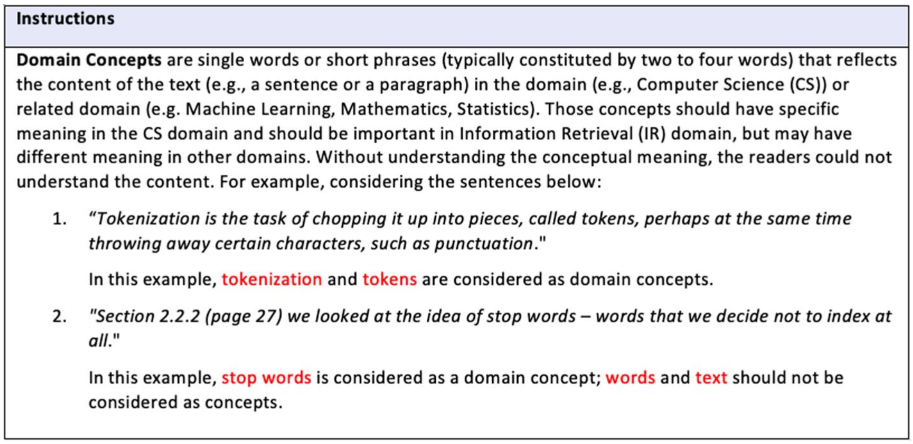
Click the below input box, Enter to add tags and hit Submit after done

- If you are not sure it is a concept or not, please add an underscore before the term (e.g., _collection).
- If you consider between two concepts, connect them with two forward slashes (e.g, bag of words//bag of words model) (note that if you think both of them are concepts, just enter separately).

information retrieval × incidence matrix ×

Fig. 2 The main interface for annotating concepts

In order to perform textbook annotation following the developed procedure, we hired three experts: one PhD student working in the IR domain and two master's students who completed a graduate IR course with high final class scores. After eleven weeks, we replaced one master's student with another master's student who also completed an IR course with a high class performance to see how the codebook could help to achieve a good agreement rate with a new annotator. The PhD student was paid by the project and the three master's students were paid a stipend of $12 per hour. The annotators were given task descriptions and the initial codebook for annotating concepts (discussed in the next



Instructions

Domain Concepts are single words or short phrases (typically constituted by two to four words) that reflects the content of the text (e.g., a sentence or a paragraph) in the domain (e.g., Computer Science (CS)) or related domain (e.g. Machine Learning, Mathematics, Statistics). Those concepts should have specific meaning in the CS domain and should be important in Information Retrieval (IR) domain, but may have different meaning in other domains. Without understanding the conceptual meaning, the readers could not understand the content. For example, considering the sentences below:

1. *"Tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation."*

   In this example, tokenization and tokens are considered as domain concepts.

2. *"Section 2.2.2 (page 27) we looked at the idea of stop words – words that we decide not to index at all."*

   In this example, stop words is considered as a domain concept; words and text should not be considered as concepts.

Fig. 3 The initial codebook for textbook concept annotation task

sections). Before starting the process, the annotators had to pass an annotation test to help familiarize themselves with both the task and the annotation interface (see Fig. 2).

3.3 Task Description

Annotators were expected to work on one chapter per week for the first 16 chapters of the IIR textbook (i.e., we only processed these chapters because they are used in a real classroom where students need to read them through an intelligent textbook interface). Each chapter includes multiple sections, which were considered as units for annotation. The sections were identified according to the headings in the table of contents of the book (unless a section was too short and could be combined with a section that follows). The annotators were required to annotate all possible concepts that appeared in the text of each section. Within each week, after completing annotating concepts, the experts sat down together to discuss cases where they did not agree with one another and came up with possible rules that could help to increase the agreement.

3.4 Initial Codebook

The annotators initially started performing the tasks by following the concept annotation instructions. The instructions shown to the annotators are shown in Fig. 3. The instructions were developed by a group of experts in the field for the tagging task, and we consider it to be the initial codebook of our coding procedure. Throughout the coding process, the codebook was updated and eventually became an outcome of the annotation procedure.

3.5 Annotating Process for the First Two Chapters

The annotators started the annotation process by following the procedures described above. They completed one chapter every week (called a "round") via the annotation interface. At the beginning of each round, the annotators tagged concepts section by section, which took about 2–3 h in total. The results (3 independent sets of annotations) were processed

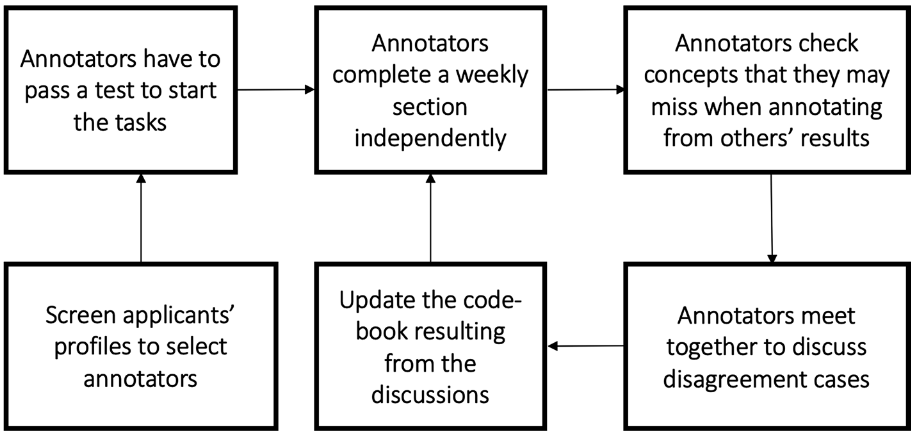

Annotators have to pass a test to start the tasks
Annotators complete a weekly section independently
Annotators check concepts that they may miss when annotating from others' results
Screen applicants' profiles to select annotators
Update the code-book resulting from the discussions
Annotators meet together to discuss disagreement cases

Fig. 4 Modified coding procedure diagram

to identify agreed cases (i.e., the concepts tagged by all three of the experts) and disagreed cases (concepts that were tagged by only one or two experts). We measure the level of agreement by calculating the proportionate inter-annotator agreement (i.e., the total number of agreed cases divided by the total number of cases). The annotators discussed the disagreement cases with one another to modify the results, which took 2–3 more hours. Based on the discussion and the analysis of disagreement cases, the codebook was updated by adding or modifying the rules and the new agreement score was re-calculated after the discussion. In the next round, annotators performed the annotation task, based on the current codebook.

3.6 Process Modification

After the first two rounds, we found out that the key reason of the low agreements before discussion is that the annotators unintentionally missed tagging particular concepts, although they agreed that those concepts should be tagged. In order to resolve this problem, we refined our annotation process by adding one more step: after completing their own annotations, the experts were required to check for missed concepts (see Fig. 4). It was done by reviewing a file where the experts could see each other's annotation results and decide whether they wanted to change their own annotations. The experts were asked to locate the missing concepts in the original context to make the decision. After checking for the missing concepts, the new agreement was calculated and the annotators discussed and updated the codebook, as described in the previous section.

3.7 Improvements from the Modified Process and Codebook

In order to see the improvements after refining the coding procedure and to demonstrate the benefit of the incrementally improving codebook, we report the inter-annotator agreement among the three annotators and also the average agreement of the pairs in Fig. 5.

In an annotation process, discussions typically help experts to correct human errors when performing their task and come up with rules that aim to resolve their conflicts. As

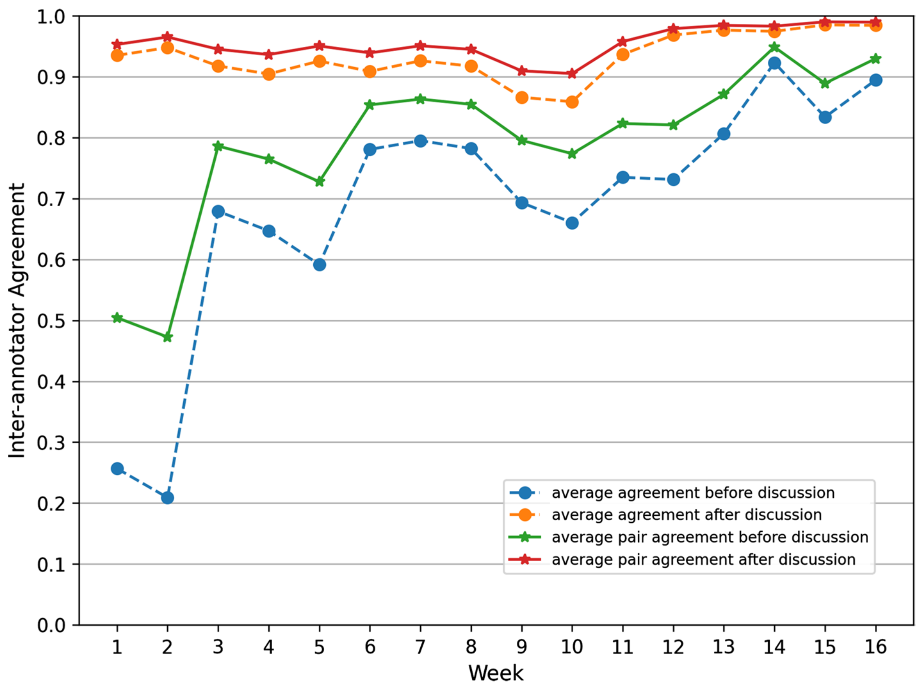


Fig. 5 Proportionate Inter-annotator Agreement Results (week by week)

shown in Fig. 5, the agreement after the discussions was very high from the start, above 0.9. However, for the first two rounds, the annotation process following the initial coding procedure resulted in a very low inter-annotator agreements of 0.25 and 0.2 before the discussion. As previously mentioned, after investigating the reason for this low agreement, we found out that although the annotators said that some concepts should be annotated, they unintentionally missed them while annotating.

From the third round on, the annotation process followed the refined coding procedure, which requires the experts to check the concepts that they missed (explained in Sect. 3.6). This refinement resulted in much higher inter-annotator agreements of above 0.6 before the discussions. Moreover, strictly following the codebook helped the experts to become more consistent in annotating particular concepts. The inter-annotator agreements before discussion had gradually been increasing, from 0.68 at week 3 up to 0.9 at the end. The after-discussion agreements also increased during the last few rounds, in which the annotators almost always agree with each others for all of the annotated concepts.

4 The Outcomes

In this section, we present the main outcomes of our attempts to develop a systematic concept annotation procedure. The outcomes include the final annotation procedure, the concept annotation codebook, and the Information Retrieval corpus, including the text of the first 86 sections from the selected *IIR* book and the list of concepts that are associated with each of the sections.

4.1 Final Coding Procedure

The final procedure for systematic concept annotation was developed in the process of full-scale practical testing of the initial procedure. While the initial procedure already integrated the best practices reported in earlier publications, our thorough testing led to an important modification explained in the previous section. The final coding procedure shown in Fig. 4 includes the following steps:

- *Step 1* The project leader screens candidate profiles to choose annotators who satisfy specific criteria; for example, background knowledge.
- *Step 2* The annotators familiarize themselves with the interface that is used to annotate knowledge components. The annotators also study the instructions that they need to follow during the annotation process. To ensure that they understood what they are asked to do and how to do it, the annotators have to pass a test that is related to the main task.
- *Step 3* The annotators complete one round of the annotation process independently for an assigned portion of text (in our case, one chapter every week), strictly following the codebook.
- *Step 4* An annotator checks for potentially missed concepts by reviewing the annotation results produced by the others. They are required to locate the missing concepts in the original text to make decisions.
- *Step 5* The annotators meet up after finishing the annotation round to discuss disagreement cases and to come up with new rules to prevent the identified conflicts in the future.
- *Step 6* The new rules from Step 5 are added to the codebook (if necessary).
- *Step 7* Move to the next assignment and repeat the process from Step 3 until all assignments are completed.

4.2 The Codebook

Table 1 lists the coding schema and detailed rules with examples of concepts and explanations. Following the coding procedure, we added one or more rules after each round. In total, we have ten rules. Most of the rules were added after the first few rounds (e.g., round 1,2,3). After round 9, no new rules were added. It indicates that the resulting table might be sufficiently complete and would be recommended for broader use.

4.3 The Corpus

The important practical outcome of our work is the IR Corpus, which is the full set of annotations for the first 16 chapters (i.e., 86 sections) of the *Introduction to Information Retrieval* textbook. We will make this data available for public use,[1] called the Systematic Knowledge Annotation (SKA) corpus. The outcome statistics for this corpus are shown in Table 2. To stress the importance of the systematic annotation process, along with the data about the final concepts (agreed by all of the three experts after their discussions, see column 4&5 in Table 2), we also report the statistics for concepts that are annotated by

[1] https://github.com/PAWSLabUniversityOfPittsburgh/Concept-Extraction.

Table 1 Coding schema for concept annotation

Rule	Description (bold text) with Examples and Explanations
1. (Round 1)	**Only noun/noun phrases are considered** Examples: Concept: sorting algorithm, wildcard pattern matching, boolean retrieval model Not concept: merging postings list, ranking documents In the examples above, *merging postings list* and *ranking documents* are not concepts, because they are not nouns or noun phrases
2. (Round 1)	**Abbreviation of a concept is also a concept** Examples: “IR (information retrieval)", “EM (expectation maximization)" *IR* and *EM* are all concepts, because *information retrieval* and *expectation maximization* are concepts
3. (Round 1)	**Annotate the whole noun/noun phrases,** **but ignore the general adj. (e.g., long, big etc.)** Examples: Concept: latent linguistic structure, hidden variables Not concept: long query, big document collection In the examples above, long and big are too general. Only *query* and *document collection* are concepts
4. (Round 2)	**If two noun phrases are concepts, the combination should be the concept** Examples: Concept: postings list, data structure, postings list data structure In the example above, *postings list* and *data structure* are concepts, so *postings list data structure* is a concept.
5. (Round 3)	**The concepts combined with conjunctions should be separated (e.g., and, or)** Examples: “boolean and proximity queries" In the example above, you need to annotate the two concepts *boolean queries* and *proximity queries*
6. (Round 5)	**All variations of the concepts should be annotated** Examples: Concept: Multi-term query, Bi-term query, Three-term query The examples above are variations of the concept query, therefore they should be annotated.
7. (Round 6)	**Annotate all special / not general phrases in the** **Computer Science related domain e.g., Statistics, mathematics** Examples: Concept: quadratic function, binomial distribution *Quadratic function* and *binomial distribution* are concepts, because they are important phrases in the Statistics domain.
8. (Round 6)	**Ignore the Abbreviation in brackets** Examples: “inverse document frequency (idf)" “variable byte (vb)" “encodingmegabytes (mb)" In the examples above, idf, vb, and mb should be ignored
9. (Round 8)	**If the concept term has punctuations, keep them** Examples: “(query, document) pairs" The example above should be annotated as a concept including the bracket and comma.
10. (Round 9)	**The well-known and important examples should be annotated** Examples: “A well-known example is the Unified Medical Language System..." In the example above, *Unified Medical Language System* should be annotated as a concept.

Table 1 (continued)

The first column shows the rule numbers and in which round this rule was added. The second column shows the description of each rule (bold text) with examples and explanations

Table 2 Data statistics of IR corpus

Characteristic	Concepts no. (before discussion)	Unique concepts no. (before discussion)	Concepts no. (after discussion)	Unique concepts no. (after discussion)
1-g	958 (36.19%)	236 (18.60%)	1121 (35.31%)	278 (18.02%)
2-g	1291 (48.77%)	8719 (56.66%)	1565 (49.29%)	871 (56.45%)
3-g	351 (13.26%)	270 (21.27%)	422 (13.29%)	330 (21.39%)
4-g	41 (1.55%)	38 (2.99%)	58 (1.83%)	55 (3.56%)
5+6-g	6 (0.23%)	6 (0.47%)	9 (0.28%)	9 (0.58%)
All grams	2647	1269	3175	1543

The concepts included in the final results are those that are agreed upon by all the three experts before the discussions (i.e., column 1 & 2) and after the discussions (i.e., column 3 & 4)

all the experts before discussions (see columns 2&3 in Table 2). Note that the numbers of concepts and unique concepts after discussions are larger than those that occur before discussions.

The distribution of n-grams is very similar both before and after discussions, which can be seen in Table 2. For the final concept list, bi-grams contribute to about *50%* of all the concepts for both cases (i.e., *number of concepts* and *number of unique concepts*). The longer a concept is, the less frequently it appears in the corpus. Unique 1-g account for 18.02% of all the unique concepts, while 1-g alone account for 35.31% of all the concepts. On the other hand, unique 3-g account for 21.39% of all of the unique concepts, while 3-g only contribute to 13.29% of all the concepts. This statistics could be helpful for designing automatic concept extraction; for instance, instead of trying to predict all the concepts, one just needs to focus on 1 to 4 g, which contribute to about 99.5% of improvements to the model performance.

4.4 Error Analysis

To show the necessity of adding rules in Table 1 for each round, we performed error analysis for each rule. During the discussion procedure in each round, the annotators found errors that either included bad concepts or that missed some good ones because of the lack of a particular rule. For example, without Rule 1, one or more annotators annotated phrases such as “sort” and “normalize”, which are important domain verbs, but are not concepts. Another example is that without Rule 2, annotators might miss some good concepts that are important abbreviations of a concept like “EM” (expectation maximization). Table 3 shows the error analysis for all rules in each round. The first column shows the rule number in Table 1 and the second column shows the problem caused without the corresponding rule. The third column shows the proportion of the errors caused by the lack of this rule among all the errors. For example, in all errors that include bad concepts, 10.61% of them occurred in Round 1 because of the lack of Rule 1. In other words, the annotators could delete 10.61% of bad concepts annotated in Round 1 with this additional rule (i.e., Rule 1).

Table 3 Error analysis for each rule in the codebook

Rule	Problem	Percentage
Rule 1 (Round 1)	Include bad concepts	10.61
Rule 2 (Round 1)	Miss good concepts	17.90
Rule 3 (Round 1)	Include bad concepts	18.18
Rule 4 (Round 2)	Miss good concepts	6.97
Rule 5 (Round 3)	Include bad concepts	9.25
Rule 6 (Round 5)	Miss good concepts	11.10
Rule 7 (Round 6)	Miss good concepts	12.50
Rule 8 (Round 6)	Include bad concepts	13.30
Rule 9 (Round 8)	Miss good concepts	6.25
Rule 10 (Round 9)	Miss good concepts	10.00

5 Evaluation

In this section, we evaluate our SKA corpus and compare it against several other corpora produced by alternate annotation approaches, including crowd-sourcing annotation and automatic annotation techniques. We first compared the SKA corpus with other corpora in terms of the statistics and other metrics (e.g., precision, recall and F1). In addition, we performed the comparisons on two other tasks: document linking and student modeling, since the quality of the annotation could lead to better support for intelligent textbooks which rely on concept annotation.

5.1 Comparision with Other Corpora

In order to demonstrate the effectiveness of our annotation procedure, we compare our SKA courpus against the baseline corpora that was obtained with alternative annotation procedures. In order to understand the importance of the discussed phrases, we also compare it against the intermediate results of the SKA process; i.e., concepts identified by each of the three expert annotators before discussion.

- *Crowd-sourcing Amazon MTurk (MTurk)* concept annotations produced by non-expert crowdworkers. In order to produce this corpus, we recruited three crowd workers from Amazon Mechanical Turk.[2] The annotators were assigned to chapters 6 and 8 of the IIR textbook (we chose these two chapters based on a reasonable amount of text for the annotation assignments), annotating 13 sections in total. We used the same interface (shown in Fig. 2) to collect the data. The workers performed their assignments independently.
- *Expert* concepts annotated by one expert. In order to model a traditional ad-hoc annotation process, one PhD student working in the IR domain (who was treated as the expert) was asked to annotate the concepts using our interface, but without any explicit guidelines or codebook.

[2] https://www.mturk.com.

Table 4 Data Statistics of different concepts corpora for IIR textbook

Corpus	Concepts no. per chapter	Unique concepts no. per chapter
MTurk	96.67	88.25
Expert	87.31	46.84
IBM	253.81	191.56
Anno w/o Discussion 1	113.81	76.06
Anno w/o Discussion 2	118.19	80.06
Anno w/o Discussion 3	127.13	85.93
SKA	198.44	96.44

Table 5 Corpora comparison with SKA

Corpus	Precision	Recall	F1
MTurk	0.41	0.26	0.32
Expert	0.60	0.34	0.42
IBM	0.21	0.39	0.25
Anno w/o Discussion 1	0.94	0.89	0.91
Anno w/o Discussion 2	0.91	0.89	0.90
Anno w/o Discussion 3	0.91	0.95	0.93

- *IBM Natural Language Understanding API (IBM)* we used the client library *watson_developer_cloud*[3] provided for Python. IBM Watson was selected as one of the most advanced examples of automatic annotation. When given a text document, the API returns a list of keywords or entities. The total number of concepts and total number of unique concepts extracted by IBM API for the first 16 chapters of the IIR book are 4061 and 3065, respectively.
- *Annotators without discussion (Anno w/o Discussion)* Three concept datasets annotated by three expert annotators following the codebook, but before the discussion stage.

Table 4 shows the basic statistics of the baselines and the SKA corpus. We observed that the corpus extracted by IBM Natural Language Understanding API has the largest number of concepts and unique concepts per chapter. It is also interesting to observe that even a single expert annotator following our annotation procedure can identify considerably more concepts than an expert performing ad-hoc annotation, who, in turn, can find fewer than half of the concepts produced by the SKA procedure. Table 5 shows the baseline corpora comparison with the SKA corpus in terms of precision, recall and F1, where we treat SKA as "ground-truth". The high number means there is a high degree of similarity between the baseline and the SKA corpus. Not surprisingly, the three datasets annotated by three annotators are the ones closest to the SKA corpus. Annotations by an expert without the codebook is more similar to the SKA corpus than the MTurk and the IBM corpus.

[3] https://github.com/watson-developer-cloud/python-sdk.

5.2 Document Linking

In this section, we evaluate the SKA corpus on the task of textbook linking. To be more specific, we attempt to use the concepts as the textbook content representation to identify similar book sections from different textbooks. We believe that textbooks are carefully designed by their authors to organize knowledge or concepts for a given field, as each section of the book contains certain knowledge that is hidden behind the concepts. Therefore, concept annotations of better quality could help to better link the textbook sections. We follow the content linking problem defined in Meng et al. (2016), which is to match the subsections in BOOK1 and the corresponding subsections in BOOK2. As it is a one-to-many match (e.g., one subsection in BOOK1 can be matched to many subsections in BOOK2), we rank all subsections in BOOK2 based on their similarity to subsections in BOOK1. We first use the concepts to represent each book subsection as a vector and then compute the similarities between sections as similarities between their vectors (using cosine similarity).

5.2.1 Ground-Truth and Experiment Design

We used the ground-truth data on subsection mapping in the information retrieval textbooks. The data includes mapping of subsections from the textbook that we used for annotation (IIR) in this work to another textbook (Baeza-Yates et al. Modern Information Retrieval; in short, *MIR*). Two experts were asked to provide the mapping score for each subsection pair. The final relevance score was computed as the average of the scores. The ground-truth dataset contains four chapters with 47 subsections from IIR, which are mapped to 88 subsections in MIR.

We used the SKA and baselines to link the two textbooks. If one of these concepts is mentioned in a book subsection, this concept will be used to represent the given book subsection. We also consider the number of occurrences of each concept. To be specific, we use the concept frequency to create a vector as the knowledge representation of each book subsection. The similarity between two book subsections is measured by cosine similarity.

5.2.2 Document Linking Evaluation

As discussed in the previous sections, one subsection in IIR may map to more than one subsection in MIR. In the ground-truth dataset, 55.3% are one-to-one relationships, 21.3% are one-to-two mapping relationships, and the rest are one-to-N ($N > 2$) mapping relationships. The well-known ranking-based evaluation metrics NDCG (Normalized Discounted Cumulative Gain)@N (Järvelin & Kekäläinen, 2002) was adopted for evaluation. As shown in Eq. 1, NDCG@p measures the quality of top p ranked book sections Discounted Cumulative Gain (DCG@p) normalized by top p Ideal Discounted Cumulative Gain (IDCG@p). DCG@p and IDCG@p are defined by Eqs. 2 and 3, where rel_i is the relevance score in the ranked book sections by different methods and REL_i is the relevance score in the ideal ranked order. NDCG@p compares the target ranking to the positions that documents occupy in the ideal list and penalizes any mismatches. NDCG@1 measures the effectiveness of the model to find the top relevant document. In the same way, NDCG@p measures the quality of ranking the first p items respectively.

Table 6 Document linking results under SKA corpus and the baselines

Corpus	NDCG@1	NDCG@3
MTurk	0.19	0.24
Expert	0.21	0.28
IBM	0.20	0.32
Anno w/o Discussion 1	0.24	0.32
Anno w/o Discussion 2	0.22	0.30
Anno w/o Discussion 3	0.23	0.30
SKA	**0.26**	**0.35**

Bold font indicates the best result

As in our dataset more than half of the mappings are either one-to-one relationships and 91.5% of them are one-to-p ($1 \leq p \leq 3$) relationships, p was set to be 1 and 3.

$$NDCG@p = \frac{DCG@p}{IDCG@p} \tag{1}$$

$$DCG@p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{2}$$

$$IDCG@p = \sum_{i=1}^{p} \frac{2^{REL_i} - 1}{\log_2(i+1)} \tag{3}$$

In this section, we provide our results with baseline corpora (refer to Sect. 5.1). We observed that the results in terms of both NDCG@1 and NDCG@3 in Table 6 are small. Similar observations that term-based methods (e.g., those only using term frequency) always perform much worse than other sophisticated methods was also found in the previous work (Guerra et al., 2013). In their work, when only applying term-based methods using all terms in the content to link the sections of five information retrieval textbooks, two of which are used in this study, the average NDCG@1 and NDCG@3 can only reach 0.057 and 0.186 respectively. In our study, we observed from Table 6 that when using concepts as terms, the results are much better, which shows concepts that can better represent the underlying knowledge. In addition, we observed that the SKA corpus performs better than both Expert and IBM concepts in terms of both NDCG@1 and NDCG@3. This shows that with the systematic annotation procedure, a team of experts can better extract hidden knowledge in the textbook. Among the three baseline corpora of MTurk, Expert, and IBM, Expert performs best at point 1. This may be because the Expert dataset is more similar to the SKA corpus (see Table 5). To see how the discussed phrases help improve the quality of the corpus, we also compared the SKA corpus with individual annotations from before the discussion. The results in Table 6 show that the SKA corpus produces better results than all three datasets produced by expert annotators before the discussion. This provides the evidence in favor of the discussion phase, which tries to bring together knowledge from different experts and combine their views to annotate the text with concepts. We also observed that each of the three datasets produced by annotators before the discussion perform better than the single expert performing an ad-hoc annotation without

Table 7 SM dataset statistics

Number of documents (sections)	394
Number of questions	158
Number of students	22
Median per student of reading time (minutes)	104
Average per student questions attempted	126
Median reading speed (words per minutes)	773
Percentage of skimming activities	33%
Percentage of reading activities	67%
Total interactions	22,536

the codebook and the discussion. This demonstrates that the codebook can guide the expert to extract a better knowledge representations from within the textbooks and thus improve the overall quality of the annotations.

5.3 Student Modeling

Student models (SMs) are used to track student learning in online-learning platforms like massive open online Courses (MOOCs) and intelligent tutoring systems (Corbett & Anderson, 1995; Pavlik et al., 2009). SMs are maintained by observing students working with learning materials and are used to adapt the system behaviors to individual students; i.e., to recommend the most relevant materials or practice activities. Modern SMs are able to maintain the level of student knowledge for a set of knowledge units (KUs). KUs, also known as knowledge components or skills, are the fundamental units upon which students' knowledge is measured. For example, a student practicing an elementary mathematics problem might have to understand knowledge units like "Addition", "Subtraction", "Mulitplication", and "Division". Traditionally, experts annotate practice activities or learning resources with KUs. To evaluate and understand the quality of annotated concepts, we used them as knowledge units for SMs and measured the predictive power of the obtained SMs. In the following subsections, we will discuss the system used, the data collection procedure, and the experiment details.

5.3.1 Ground-Truth and Experiment Design

The dataset used for this experiment is collected from an online reading platform, *System C* (anonymized). This system was used in a graduate-level Information Retrieval course. The system provides an active reading environment to the student where they read the assigned textbook sections to prepare for the next class. Each section of textbook is followed by a quiz, which allows students to assess how well they learned the content. There is no restriction on the number of attempts to answer the questions. *System C* logs all attempts made by the student. The dataset contains students' time spent on reading sections and quiz performance. The dataset includes interactions from 22 students collected from the Spring 2016 semester. Details of the dataset are listed in Table 7.

To assess the quality of each corpus, we used their concepts as KUs to model students' reading and quiz attempt behavior and predict their future performances. To perform this, we used a comprehensive factor analysis Model (CFM) (Thaker et al., 2019). CFM is a logistic regression based model that takes students' previous performances and reading

Table 8 Results of student performance prediction with SKA corpus and the baselines

Corpus	AUC	RMSE
Expert	0.541	0.475
IBM	0.624	0.385
Anno w/o Discussion 1	0.618	0.386
Anno w/o Discussion 2	0.602	0.401
Anno w/o Discussion 3	0.584	0.421
SKA	**0.633**	**0.363**

Bold font indicates the best result

behaviors to predict their success rate for a given question. We selected CFM to model student performances, as it performs better on intelligent textbooks than other state-of-the-art student modeling approaches and also incorporates student reading behavior, which has proved to be beneficial in cases of online textbook-based learning systems (Huang et al., 2016; Thaker et al., 2018).

5.3.2 Student Modeling Evaluation

To evaluate the performance of CFM on student performance, we performed 5-fold cross-validation with student-stratified folds. First, we randomly selected 80% of students and put all their reading and quiz activity data into a training set. Then for the remaining 20% of students, all of their reading and quiz activity data was put into a test set. The predictions are reported for quiz performances. The 5-fold cross-validation is performed from the generated folds and area under the receiver operating characteristic curve (AUC) and root mean squared error (RMSE) are reported. Larger AUC and lower RMSE numbers indicate better results.

- *Comparison with other annotation methods*

  In this section, we report our results with baseline corpora (refer to Sect. 5.1) to understand the importance of the SKA approach over other annotation methods. As shown in Table 8, SKA performs better than both expert annotation and IBM concepts. This shows that the codebook-based annotation method is better at extracting KUs than the simple expert annotation (Expert) and the automatic concept extraction method (IBM). An important insight was the small gap in the differences between the performance of IBM and SKA. The previous research has shown that small improvements in student modeling are acceptable and provide a significant improvement in student activity adaptation, based on student models (Thaker et al., 2018, 2020)

  We ignored the MTurk baseline, as crowdsourced annotations were collected for only two chapters, which is not sufficient to model student performance. In order to train student models, we need datasets from several chapters in a sequence. Currently we don't have the MTurk data from all the chapters; we only have a dataset for chapter 2 and 8, which don't have any overlap needed to train and test students' knowledge. Collecting new data will take a lot of effort, which is not the focus of this work, so we left it for future study.
- *Comparison with expert annotations before discussion*

  To understand the importance of the discussion phase of the annotation process, we also compared our final annotations with annotations without different discussion phases. The results of this phase are listed in Table 8. As the results show, SKA annotations perform better than each of the annotations where any of the discussion phases are missing; with an increase in AUC, this depicts the effectiveness of the discussion phase. The results also make it evident that removing *Discussion 3* affects the results more drastically, and that this shows the importance of each discussion phase. The results provide an important insight that *Anno w/o Discussion* in SKA approach was able to come up with better concepts than the Expert. This is another piece of evidence of the effectiveness of using the codebook and the discussion phases.

6 Conclusion and Future Work

In this paper, we present a reliable systematic knowledge engineering approach for fine-grained annotation of textbooks with underlying knowledge in the form of concepts. We explored this approach by performing a full-scale annotation procedure on a popular open source textbook *Introduction to Information Retrieval* (IIR). In the process of working with IIR, we refined and finalized the proposed approach. The low inter-agreement among annotators, in the beginning, shows that it is hard to annotate concepts from a textbook, even by domain experts, without proper procedure. The observation that the inter-agreement among annotators gradually increased by following our proposed procedure shows the effectiveness of this procedure.

In order to evaluate the quality of our SKA corpus, we compared our SKA corpus against alternately produced annotation corpora in terms of their performance on two target tasks performed by intelligent textbooks: document linking and student modeling. The results demonstrate that our SKA corpus can achieve better performance in both tasks, as compared with other annotation corpora. Altogether, our data indicates that the annotation process presented in this paper could be used to augment textbooks with “knowledge behind pages” that could effectively support several needs of intelligent textbooks.

Besides this approach itself, the outcomes of our work include a codebook, which can be used to annotate similar textbooks, and a public dataset. The dataset includes the textbook content and a full set of section-level annotation (SKA corpus) and could be used by the document engineering community to refine and evaluate keyphrase extraction and generation approaches. In our own research, we were able to apply the produced dataset to train a well-performing automatic keyphrase annotation approach (Chau et al., 2020). We believe that the presented team-based systematic knowledge annotation approach could be used to produce a broader collection of high-quality datasets for training automatic models.

In our future work, we plan to continue our exploration of quality-focused concept annotation approaches for textbooks addressing a number of challenges and opportunities that we were not able to explore in the first round of our research. In particular, we are interested to explore more opportunities to engage a team of crowdworkers in the process of annotation. While we demonstrated that a team of experts working with a codebook could produce a higher-quality annotation than a single expert of a crowdworker, once a codebook is developed by experts, it might also empower crowdworkers. It remains to be seen whether the annotation produced by crowdworkers *with* the codebook could reach the quality produced by a team of experts. Second, the observed good results produced by the IBM automatic approach in both assessment tasks encourage us to explore a hybrid approach that combines the automatic extraction method and the systematic expert procedure. The automatic extraction method may have potential power in improving the quality of the annotation, as well as reducing the overall annotation load.[4]

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. 1525186. This research is also supported in part by the University of Pittsburgh Center for Research Computing through the resources provided.

References

Agrawal, R., Gollapudi, S., Kannan, A., & Kenthapadi, K. (2014). Study navigator: An algorithmically generated aid for learning from electronic textbooks. *Journal of Educational Data Mining, 6*(1), 53–75.

Augenstein, I., Das, M., Riedel, S., Vikraman, L., & McCallum, A. (2017). Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications. Preprint arXiv:1704.02853

Bareiss, R., & Osgood, R. (1993). Applying ai models to the design of exploratory hypermedia systems. In *Fifth ACM Conference on Hypertext* (pp. 94–105). ACM Press.

Bayerl, P. S., Lüngen, H., Gut, U., & Paul, K. I. (2003). Methodology for reliable schema development and evaluation of manual annotations. In *Proceedings of the workshop on knowledge markup and semantic annotation at the second international conference on knowledge Capture (K-CAP. (2003)*. ACM.

Bougouin, A., Boudin, F., & Daille, B. (2013). Topicrank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the sixth international joint conference on natural language processing* (pp. 543–551). Asian Federation of Natural Language Processing.

Brusilovsky, P. (2003). Developing adaptive educational hypermedia systems: From design models to authoring tools. In T. Murray, S. Blessing, & S. Ainsworth (Eds.), *Authoring tools for advanced technology learning environments: Toward cost-effective adaptive, interactive, and intelligent educational software* (pp. 377–409). Kluwer.

Brusilovsky, P., Chavan, G., & Farzan, R. (2004). Social adaptive navigation support for open corpus electronic textbooks. Lecture Notes in Computer Science. In P. De Bra & W. Nejdl (Eds.), *Third international conference on adaptive hypermedia and adaptive web-based systems (AH'2004)* (Vol. 3137, pp. 24–33). Springer-Verlag.

Brusilovsky, P., Eklund, J., & Schwarz, E. (1998). Web-based education for all: A tool for development adaptive courseware. *Computer Networks and ISDN Systems, 30*(1), 291–300. https://doi.org/10.1016/S0169-7552(98)00082-8.

Brusilovsky, P., & Pesin, L. (1998). Adaptive navigation support in educational hypermedia: An evaluation of the isis-tutor. *Journal of Computing and Information Technology, 6*(1), 27–38.

Chau, H., Labutov, I., Thaker, K., He, D., & Brusilovsky, P. (2020). Automatic concept extraction for domain and student modeling in adaptive textbooks. *International Journal of Artificial Intelligence in Education*.

Chaudhri, V. K., Cheng, B., Overtholtzer, A., Roschelle, J., Spaulding, A., Clark, P., et al. (2013). Inquire biology: A textbook that answers questions. *AI Magazine, 34*(3), 55–72.

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*(4), 253–278.

Florescu, C., & Caragea, C. (2017). Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th annual meeting of the association for computational linguistics (Volume 1: Long Papers)* (pp. 1105–1115). Association for Computational Linguistics.

Green, S. J. (1999). Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering, 11*(5), 713–730.

Grineva, M., Grinev, M., & Lizorkin, D. (2009). Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World Wide Web, WWW '09* (pp. 661–670). ACM.

[4] http://crc.pitt.edu.

Guerra, J., Sosnovsky, S., & Brusilovsky, P. (2013). When one textbook is not enough: Linking multiple textbooks using probabilistic topic models. In *European conference on technology enhanced learning* (pp. 125–138). Springer.

Henze, N., Naceur, K., Nejdl, W., & Wolpers, M. (1999). Adaptive hyperbooks for constructivist teaching. *Künstliche Intelligenz, 13*(4), 26–31.

Henze, N., & Nejdl, W. (2001). Adaptation in open corpus hypermedia. *International Journal of Artificial Intelligence in Education, 12*(4), 325–350.

Huang, Y., Yudelson, M., Han, S., He, D., & Brusilovsky, P. (2016). A framework for dynamic knowledge modeling in textbook-based learning. In *Proceedings of 24th conference on user modeling, adaptation and personalization* (pp. 141–150). ACM.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on empirical methods in natural language processing, EMNLP '03* (pp. 216–223). Association for Computational Linguistics.

Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS), 20*(4), 422–446.

Jiang, X., Hu, Y., & Li, H. (2009). A ranking approach to keyphrase extraction. In *Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval, SIGIR '09* (pp. 756–757). ACM.

Kavcic, A. (2004). Fuzzy user modeling for adaptation in educational hypermedia. *IEEE Transactions on Systems, Man, and Cybernetics, 34*(4), 439–449.

Labutov, I., Huang, Y., Brusilovsky, P., & He, D. (2017). Semi-supervised techniques for mining learning outcomes and prerequisites. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, KDD '17* (pp. 907–915). ACM.

Lakkaraju, P., Gauch, S., & Speretta, M. (2008). Document similarity based on concept tree distance. In *The 19th ACM conference on hypertext & hypermedia* (pp. 127–131). ACM.

Lan, A. S., & Baraniuk, R. G. (2016). A contextual bandits framework for personalized learning action selection. In T. Barnes, M. Chi, & M. Feng (Eds.), *The 9th international conference on educational data mining (EDM 2016)* (pp. 424–429). Raleigh.

Le, T. T. N., Nguyen, M. L., & Shimazu, A. (2016). Unsupervised keyphrase extraction: Introducing new kinds of words to keyphrases. In B. H. Kang & Q. Bai (Eds.), *AI 2016: Advances in artificial intelligence* (pp. 665–671). Springer International Publishing.

Liu, F., Pennell, D., Liu, F., & Liu, Y. (2009). Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics, NAACL '09* (pp. 620–628). Association for Computational Linguistics.

Liu, Z., Huang, W., Zheng, Y., & Sun, M. (2010). Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing, EMNLP '10* (pp. 366–376). Association for Computational Linguistics.

Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 1–Volume 1, EMNLP '09* (pp. 257–266). Association for Computational Linguistics.

Medelyan, O., Frank, E., & Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 3–Volume 3, EMNLP '09* (pp. 1318–1327). Association for Computational Linguistics.

Melis, E., Andrès, E., Büdenbender, J., Frishauf, A., Goguadse, G., Libbrecht, P., et al. (2001). Activemath: A web-based learning environment. *International Journal of Artificial Intelligence in Education, 12*(4), 385–407.

Meng, R., Han, S., Huang, Y., He, D., & Brusilovsky, P. (2016). Knowledge-based content linking for online textbooks. In *2016 IEEE/WIC/ACM international conference on web intelligence (WI)* (pp. 18–25). IEEE.

Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., & Chi, Y. (2017). Deep keyphrase generation. In *ACL2017, annual meeting of the association for computational linguistics* (pp. 836–845). ACL. arxiv:1704.06879

Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of EMNLP 2004* (pp. 404–411). Association for Computational Linguistics. https://www.aclweb.org/anthology/W04-3252

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Stanford InfoLab: Tech. rep.

Papanikolaou, K. A., Grigoriadou, M., Kornilakis, H., & Magoulas, G. D. (2003). Personalising the interaction in a web-based educational hypermedia system: The case of inspire. *User Modeling and User Adapted Interaction, 13*(3), 213–267.

Pavlik, P., Cen, H., & Koedinger, K.R. (2009). Performance factors analysis–a new alternative to knowledge tracing. In *Proceedings of the 2009 conference on artificial intelligence in education: Building learning systems that care: From knowledge representation to affective modelling* (pp. 531–538). IOS Press.

Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. In M. W. Berry & J. Kogan (Eds.), *Text mining: Applications and theory* (pp. 1–20). John Wiley and Sons Ltd.

Shamsfard, M., & Barforoush, A. A. (2004). Learning ontologies from natural language texts. *International Journal on Human-Computer Studies, 60,* 17–63.

Thaker, K., Carvalho, P., & Koedinger, K. (2019). Comprehension factor analysis: Modeling student's reading behaviour: Accounting for reading practice in predicting students' learning in moocs. In *Proceedings of the 9th international conference on learning analytics and knowledge* (pp. 111–115). ACM.

Thaker, K., Huang, Y., Brusilovsky, P., & He, D. (2018). Dynamic knowledge modeling with heterogeneous activities for adaptive textbooks. In *The 11th international conference on educational data mining* (pp. 592–595). ACM.

Thaker, K., Zhang, L., He, D., & Brusilovsky, P. (2020). Recommending remedial readings using student knowledge state. In *Proceedings of The 13th international conference on educational data mining (EDM 2020)* (pp. 233–244).

Wang, S., Liang, C., Wu, Z., Williams, K., Pursel, B., Brautigam, B., Saul, S., Williams, H., Bowen, K., & Giles, C. L. (2015). Concept hierarchy extraction from textbooks. In *Proceedings of the 2015 ACM symposium on document engineering, DocEng '15* (pp. 147–156). ACM.

Weber, G., & Brusilovsky, P. (2001). ELM-ART: An adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education, 12,* 351–384.

Wilbur, W. J., Rzhetsky, A., & Shatkay, H. (2006). New directions in biomedical text annotation: Definitions, guidelines and corpus construction. *BMC Bioinformatics, 7*(1), 1–10.

Winchell, A., Mozer, M., Lan, A., Grimaldi, P., & Pashler, H. (2018). Can textbook annotations serve as an early predictor of student learning? In *The 11th international conference on educational data mining* (pp. 431–437). ERIC.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (2005). Kea: Practical automated keyphrase extraction. *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific* (pp. 129–152). IGI Global.

Wong, W., Liu, W., & Bennamoun, M. (2012). Ontology learning from text: A look back and into the future. *ACM Computer Survey, 44*(4).

Xia, F., & Yetisgen-Yildiz, M. (2012). Clinical corpus annotation: Challenges and strategies. In *Proceedings of the third workshop on building and evaluating resources for biomedical text mining (BioTxtM'2012) in conjunction with the international conference on language resources and evaluation (LREC), Istanbul, Turkey*.

Authors and Affiliations

Mengdi Wang[1] · **Hung Chau**[1] · **Khushboo Thaker**[1] · **Peter Brusilovsky**[1] · **Daqing He**[1]

Mengdi Wang
mew133@pitt.edu

Hung Chau
hkc6@pitt.edu

Khushboo Thaker
k.thaker@pitt.edu

Daqing He
dah44@pitt.edu

[1] School of Computing and Information, University of Pittsburgh, Pittsburgh, PA 15260, USA