Knowledge Engineering for Intelligent Textbooks

Hung Chau*, Mengdi Wang*
Khushboo Thaker
School of Computing and Information
University of Pittsburgh
Pittsburgh, PA, USA
{mengdi.wang,hkc6,k.thaker}@pitt.edu

ABSTRACT

With the increased popularity of electronic textbooks, there is a growing interests in developing a new generation of "intelligent textbooks", which have the ability to guide the readers according to their learning goals and current knowledge. The intelligent textbooks extend regular textbooks by integrating machine-manipulatable knowledge such as a knowledge map or a prerequisite-outcome relationship between sections, among which, the most popular integrated knowledge is a list of unique knowledge concepts associated with each section. With the help of these concept, multiple intelligent operations, such as content linking, content recommendation or student modeling, can be performed. However, annotating a reliable set of concepts to a textbook section is a challenge. Automatic unsupervised methods for extracting key-phrases as the concepts are known to have insufficient accuracy. Manual annotation by experts is considered as a preferred approach and can be used to produce both the target outcome and the labeled data for training supervised models. However, most researchers in education domain still consider the concept annotation process as an ad-hoc activity rather than an engineering task, resulting in low-quality annotated data. In this paper, we present a textbook knowledge engineering method to obtain reliable concept annotations. The approach has been applied to produce annotated concepts for *Introduction to Information Retrieval* textbook. As shown 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 outcomes of our work include a validated knowledge engineering procedure, a code-book for technical concept annotation, and a set of concept annotations for the target textbook, which could be used as gold standard in further research.

CCS CONCEPTS

• Information systems → Information extraction; **• Applied computing → E-learning**.

*Both authors contributed equally to this research.

DocEng '19, Sep. 23-26, 2019, Berlin, Germany

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

KEYWORDS

Knowledge engineering, concept annotation, concept mining, annotation scheme, intelligent textbook, electronic textbook

ACM Reference Format:
Hung Chau*, Mengdi Wang and Khushboo Thaker. 2018. Knowledge Engineering for Intelligent Textbooks. In *DocEng '19: ACM Symposium on Document Engineering, Sep. 23-26, 2010, Berlin, Germany*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Modern textbooks have been developed and refined over many decades to evolve into well-organized tools for communicating knowledge and educating the next generation of professionals. Yet, the power of computing and internet caused the textbooks to evolve even faster than before. The conversion of textbooks into electronic format created an opportunity to augment textbooks with novel functionalities based on application or Artificial Intelligence. This direction of research, usually referred as "intelligent textbooks" explored a range of novel ideas over the last 20 years. The explored approaches include adaptive navigation support [17], natural language question answering [9], automatic link creation[16], and personalized recommendation of external content [1].

The key to the power of most of the intelligent textbook technologies is "knowledge behind pages", which this technologies need to operate. These knowledge are usually extracted using a combination of machine learning, automatic natural language processing, and human knowledge engineering, i.e., annotation by human experts. Expert annotation is known to be of higher quality and is frequently used as the "gold standard" to assess the quality of automatic approaches. For some easier tasks such as content linking or content recommendation, automatic processing could support sufficient levels of quality. For more challenging tasks, such as personalization, the use of expert annotation in some form is essential. The problem is, however, that even an expert-level knowledge annotation might not achieve a quality required by intelligent approaches, unless it is guided by a reliable systematic procedure. In this paper we present our work on developing and evaluation of a systematic knowledge engineering approach for fine-grained annotation of textbooks with underlying knowledge in the form of concepts. Our study demonstrates that this approach produces better results in performance-based evaluation.

2 RELATED WORK

2.1 Intelligent Textbooks

The research on intelligent textbooks could be traced back to the early attempts to develop electronic textbooks using pre-Web hypertext systems. At that time, artificial intelligence (AI) approaches were used to automate link creation between hypertext pages, which is an essential process to create a high quality hypertext [2]. Since these early attempts, "intelligent linking" remained as an integral part of hypertext research. A range of increasingly more advanced approaches to extract concepts and other semantic features from hypertext pages have been reported [1, 13, 16, 23].

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 in adaptive online textbooks, these systems attempted to model user knowledge and support the users with adaptive navigation within a book [8, 17, 21] as well as adaptive content presentation [30]. This generation of adaptive textbooks has been based on relatively advanced models of content annotation by domain experts, frequently using domain ontologies [6]. Similar to automatic linking research, the research on concept-based adaptive textbooks remains active and focus on more advanced personalization technologies as well as automative domain model development and concept indexing.

The most recent generation of intelligent textbook was fueled by the increased availability of user data and focused on combining artificial and collective intelligence. Started with early attempts of using past users' behavior to provide social navigation support for future learners [7], the research of this direction explored increasingly more complex approaches for mining past users' behavior to guide new users [24] and predict their success [40]. Modern research on intelligent textbooks also frequently combines the ideas of automatic linking, personalization, concept annotation, and data mining [22, 24].

2.2 Ground Truth Annotation

Despite efforts to automate annotations of documents, manual annotations still play an important role in the construction of corpora for document engineering. The quality of such manual annotations depends on a reliable coding schema. A coding schema can be seen as a set of guidelines to assign an objective (e.g., morphemes, words, phrases, sentences) to a single category. [3] identified two considerations for a coding schema: 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. [3] also proposed a methodological framework consisting of five successive steps for systematic schema development. Various schemata for ground truth annotation of documents were developed for different applications. For example, [11] explored sentiment annotation tools for sentiment analysis, which has gain high popularity and several academic projects emerged in this field. [38] proposed a manual annotation framework to link short fragments of text within a document for entity linking. [4] used several knowledge bases for a semantic annotation strategy.

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* [22], or *concept hierarchy creation* [39]. Among these applications, concept extraction is the most fundamental task that leads to the success of other tasks; i.e., in order to predict a concept is a prerequisite or outcome concept we first need to identify if it is a concept.

Dozens of studies have tried to extract key-phrase automatically with different kinds of approaches including rules-based, supervised learning, unsupervised learning, and deep neural networks. However, their performance is still very low, making them are not effective enough to use for certain applications; for example, explainable recommendation systems. Typically, automatic keyphrase extraction systems consist of two parts. Firstly, they need to preprocess data and then extract a candidate keyphrase list with lexical patterns and heuristics [5, 12, 14, 25, 26, 28, 29, 33, 39]. Secondly, the candidates are ranked or classified to identify correct keyphrases using unsupervised methods or supervised with handcrafted features. Candidates are scored based on some properties that show how likely a candidate being a keyphrase in the given document. Many studies have formed this task as a binary classification problem to determine correct keyphrases [19, 19, 20, 35, 39, 41].

For unsupervised learning, graph-based methods [5, 33] try to find important keyphrases in a document. A candidate is important when it has relationships with other candidates and those candidates are also important in the document, forming a graph representing 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*. Finally, they select the top-ranked candidates as keyphrases for the input document. On the other hand, topic-based clustering methods [14, 27, 28] 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 applied to many NPL-related tasks, sequence tagging, named entity recognition, to name a few, few studies have focused on keyphrase extraction problem; and none of them have evaluated on textbook datasets. Meng et al.[32] built 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 to use in the educational setting since the datasets used to evaluated were scientific articles and paper abstracts.

Wang et al. [39] 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 concept extraction task, they use Wikipedia titles as a external resource to identify concepts appearing the textbook's table of content. As a result, there are only a few important extracted concepts considered as topic levels for building a hierarchy.

3 TEXTBOOK KNOWLEDGE ANNOTATION

In education domain, knowledge annotation has been perform in many studies because its results often served as the primary input

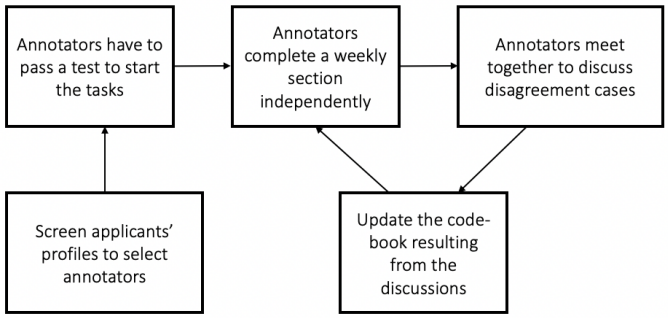

Figure 1: Coding procedure diagram. The annotators follow the procedure until they complete the whole process.

for the methods being developed. However, researchers usually perform it as an ad-hoc task and it is known to be a very challenging task. This is because it is hard to maintain consistency during the long process of annotation without clear rules and descriptions.

To overcome this challenge, we designed a systematic textbook annotation procedure, and applied it in the annotation of a popular online available textbook *Introduction to Information Retrieval* (IIR) [1]. The goal of our annotation is to add concepts to the book so that to turn it into an intelligent textbook, and this annotation task help us to refine the proposed textbook annotation procedure.

3.1 The Case Study: Introduction to Information Retrieval

The ultimate goal of our research project is the development of intelligent textbooks, which could offer a rich set of support functionalities to their readers, including automatic linking and content recommendation. IIR textbook was one of our first targets. 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 IIR textbook, we previously explored traditional ad-hoc expert annotation, crowdsourcing, concept extraction, and 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 to offer evaluation baselines.

3.2 Initial Coding Procedure

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

3.3 Hiring Process

To perform textbook annotation following the developed procedure, we hired three experts, one PhD student working in IR domain and two Master students who completed a graduate IR course with high final class scores. After eleven weeks, we replaced one Master student with a new Master student who also completed the IR course with high scores to see how the code book could help to achieve a good agreement rate with a new annotator. The PhD student was paid by the project and the three Master students were paid a stipend of $12 per hour. The annotators were given task descriptions and the initial code book for annotating concepts (discussed in the next sections). Before staring the process, the annotators had to pass an annotation test and make themselves familiar with the task and the annotation interface (see Figure 2).

3.4 Task Description

Annotators were expected to work on one chapter per week for the first 16 chapters of IIR textbook (i.e., we only process these chapters because they are used in a real class room that students need to read them through an intelligent textbook interface). Each chapter includes multiple sections, which were considered as units or annotations. The sections were identified according to the headings in the table of content of the book (unless a section is too short and can be combined with the consecutive sections). The annotators were required to annotate all possible concepts which appear in the text of each section. Within a week, after completing annotating concepts, experts need to sit down together to discuss cases that they do not agree with one another, and come up with possible rules that help to increase the agreement.

3.5 Initial Code Book

The annotators initially started performing the tasks by following a concept annotation instructions. The instructions shown to the annotators are depicted in Figure 3. The instructions were developed by a group of experts in the field for the tagging tasks, and we consider it as the initial code book of our coding procedure. Throughout the coding process, the code book had to be updated and eventually become an outcome of the annotation procedure.

3.6 Annotating Process for the First Two Chapters

The annotators started the annotation process following the procedures described above. They completed one chapter every week (called "round") via the annotation interface. At the beginning of each round the annotators tagged concepts section by section, which took about 2-3 hours in total. The results (3 independent sets of annotations) were processed to identify agreement cases (i.e., the concepts tagged by all three experts) and disagreement cases (concepts that were tagged by two or only one expert). The annotators set up meetings to discuss disagreement cases they do not agree with one another and modify the results, which took another 2-3 hours. Based on the discussion and the analysis of disagreement cases, the code book was updated by adding or modifying the rules and the new agreement score was re-calculated after discussion. In the next round, annotators performed the annotation task based on the updated code book from the previous rounds.

[1] https://nlp.stanford.edu/IR-book/

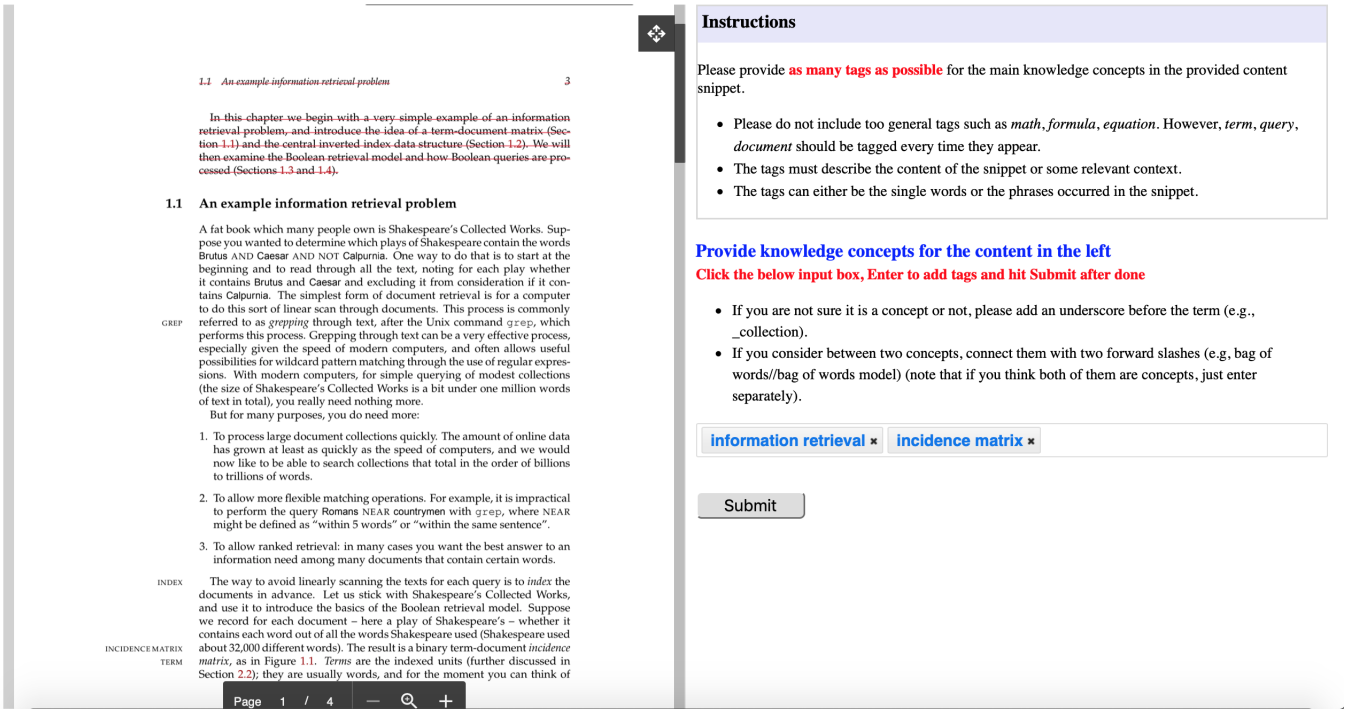

Figure 2: The main interface for annotating concepts.

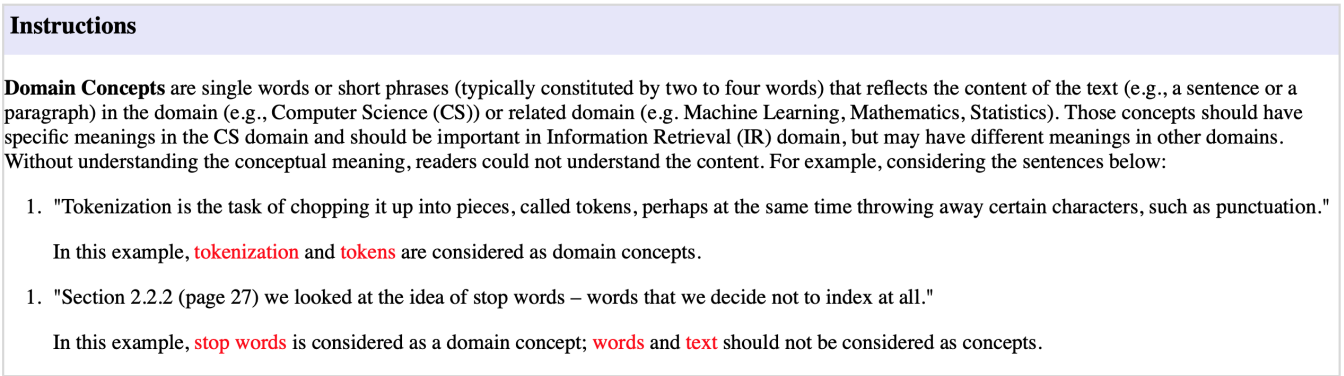

Figure 3: The initial code book for textbook concept annotation task.

3.7 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 the concepts although they agree that these concepts should be tagged. To resolve this problem, we refined our annotation process by adding one more step: after completing their own annotation part, the experts were required to check missed concepts (see Figure 3.7). It was done by reviewing a file where the experts could see each other's annotation results and decide whether they want to change their own annotations. The experts were asked to locate the missing concepts in the original context to make the decision. After checking the missing terms, the new agreement was calculated and the annotators discussed and updated the code book as described in the previous section.

3.8 Improvements from the Modified Process and Code Book

To see the improvements after refining the coding procedure and to demonstrate the benefit of the incrementally improving code book, we report the inter-annotator agreement among the three annotators and also the average agreement of the pairs in Figure 5.

As can be seen in Figure 5, the agreement after the discussions was very high from the start, above *0.9*. However, for the first two

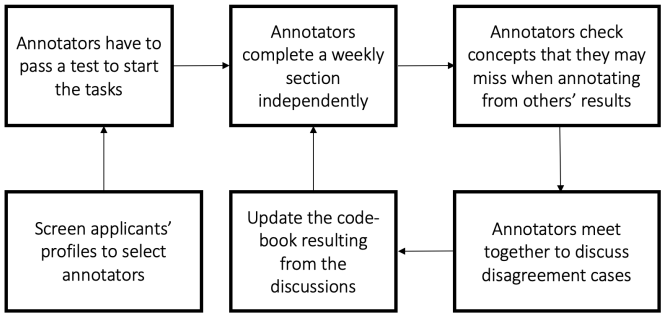

Figure 4: Modified Coding procedure diagram.

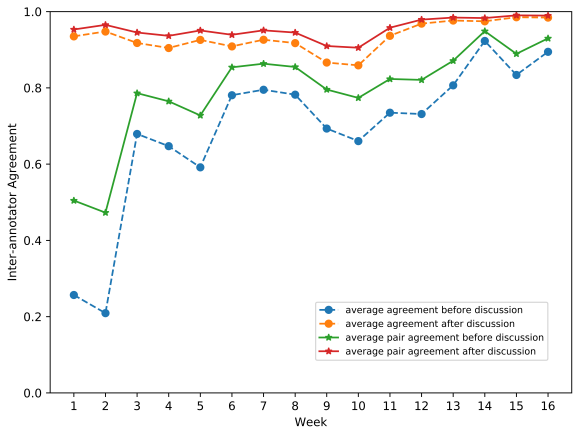

Figure 5: Inter-annotator Agreement Results (week by week).

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 mentioned above, investigating the reason of this low agreement, we found out that though 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 missing concepts (explained in Section 3.7). This refinement resulted in much higher inter-annotator agreements of above *0.6* before the discussions. Moreover, strictly following the code-book helped the experts to become more consistent in annotating concepts. The inter-annotator agreements before discussing had gradually been increasing from *0.68* at week 3 up to *0.9* at the end. The after-discussion agreements also increased at the last few rounds, in which the annotators almost agree with each others for all 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 code book and the Information Retrieval corpus including the text of the first 86 sections from the selected *IIR* book and the list of concepts 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 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 Figure 4 includes the following steps:

- **Step 1**: The project lead screen profiles of candidate annotators to choose annotators who satisfy specific criteria; for example: background knowledge.
- **Step 2**: The annotators make themselves familiar with the interface that is used to annotate knowledge components. The annotators also study the instructions that they need to follow in the annotation process. To ensure that they understood what they are asked to do and how to do it, the annotators had to pass a test related to the main tasks.
- **Step 3**: The annotators complete one round of annotations processing independently an assigned portion of text (in our case, one chapter every week) following the code book.
- **Step 4**: The annotators check potentially missed concepts by reviewing the annotation results produced by other annotators. They are required to locate the missing concepts in the original text to make decisions.
- **Step 5**: The annotators meet 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 code book (if necessary).
- **Step 7**: Switch to the next portion of text to be annotated and repeat the process starting from Step 3 until completing all text is annotated.

4.2 Code book

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 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 *Introduction to Information Retrieval* textbook. We make this data available on Github folder[2], called SKA (i.e., Systematic Knowledge Annotation) corpus. Some process and outcome statistics for this corpus is shown in Table 2. To stress the importance of the systematic annotation process, along with the data about final concepts (agreed by all the three experts after their discussions, see column 4&5 in Table 2), we also report the statistics for concepts

[2]https://github.com/PAWSLabUniversityOfPittsburgh/Concept-Extraction/IIR-dataset

Rule	Description	Examples & Explaination
1. (Round 1)	Only noun/noun phrases are considered.	**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.	-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.)	**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.	**Concept:** 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).	- "boolean and proximity queries" In the example above, you need to annotate the two concepts *boolean queries* and *proximity queries*
6. (Round 5)	All variation of the concepts should be annotated.	-Multi-term query -Bi-term query -Three-term query The examples above are variation of the concept *query*, therefore they should be annotated.
7. (Round 6)	Annotate all special / not general phrases in computer science related domain e.g., Statistics, mathematics	**Concepts:** quadratic function, binomial distribution *Quadratic function* and *binomial distribution* are concepts, because they are important phrases in Statistics domain.
8. (Round 6)	Ignore the Abbreviation in brackets.	-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.	- (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.	- A well-known example is the Unified Medical Language System... *Unified Medical Language System* should be annotated.

Table 1: Coding schema for concept annotation

that are annotated by all the experts before discussions (see column 2&3 in Table 2). Note that the number of concepts and unique concepts after discussions are larger than those before discussions.

As also can be seen in Table 2, the distribution of n-grams is very similar before and after discussion. 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 frequent it in the corpus. Unique 1-grams account for *18.02%* of all the unique concepts while 1-grams alone account for *35.31%* of all the concepts. On the other hand, unique 3-grams account for *21.39%* of all the unique concepts while 3-grams 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 one to four grams which contribute to about *99.5%* to improve the model performance.

5 EVALUATION

In this section, we evaluate our SKA corpus and compare it against several baselines produced by alternative annotation approaches. Since the main criteria of annotation quality for us is better support for intelligent textbooks, the comparison is performed on two tasks where intelligent textbooks rely on concept annotation: document linking and student modeling.

5.1 Baselines

To demonstrate the effectiveness of our annotation procedure, we compare our SKA courpus against the baseline corpora obtained with alternative annotation procedures. To understand the importance of the discussion phrase, 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):** concepts annotations produced by non-expert crowdworkers. To produce this corpus, we recruited three crowdworkers from Amazon Mechanical Turk[3]. The annotators were assigned to chapter 6 and 8 of IIR textbook (we chose these two chapters based on the reasonable amount of text for the annotation assignments), annotating in total 13 sections. We used the same interface (shown in Figure 2) to collect the data. The workers perform their assignments independently.
- **Expert:** concepts annotated by one expert. To model traditional ad-hoc annotation process, one PhD student working in IR domain (who is treated as the expert) was asked to annotate the concepts using our interface but without any explicit guidelines or code book.
- **IBM Natural Language Understanding API (IBM):** we use the client library *watson_developer_cloud* provided for Python[4]. IBM Watson was selected as one of the most advanced examples of automatic annotation. Given a text document, the API will return 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 concepts datasets annotated by three expert annotators following the code book but before the discussion stage.

Table 3 shows the basic statistics of the baselines and SKA corpus. We observe that 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 less than a half of the concepts produced by SKA procedure. Table 4 shows the baseline corpora comparison with SKA corpus in terms of precision, recall and F1, where we treat SKA as "ground-turth". The high number means the high similarity between the baseline and the SKA corpus. Not suprisingly, the three datasets annotated by three annotators are most close to the SKA corpus. Annotation by expert alone without code book is more similar to the SKA corpus than MTurk and IBM corpus.

5.2 Document Linking

In this section, we evaluate 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 subsections from different textbooks. We believe that textbooks are carefully designed by their authors to organize knowledge or concepts for a given field, as each book section contains certain knowledge hidden behind the concepts. Therefore, concept annotations of better quality could help to better link the textbook sections.

5.2.1 Document Linking Problem. We follow the content linking problem defined in [31], which is to match the subsections in BOOK1 and the corresponding subsections in BOOK2. As it is 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 the 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).

5.2.2 Ground-truth. We used the ground-truth data on subsection mapping in the information retrieval textbooks prepared by Guerra et al. [16]. The data includes mapping of subsections from the textbook which 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.

5.2.3 Evaluation Metrics. 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; the rest are one-to-N ($N > 2$) mapping relationships. The well-known ranking-based evaluation metrics NDCG@N was adopted for evaluation. As more than half of the mappings are either one-to-one relationships and

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

Characteristic	Number of concepts (before discussion)	Number of unique concepts (before discussion)	Number of concepts (after discussion)	Number of unique concepts (after discussion)
1-grams	958 (36.19%)	236 (18.60%)	1121 (35.31%)	278 (18.02%)
2-grams	1291 (48.77%)	8719 (56.66%)	1565 (49.29%)	871 (56.45%)
3-grams	351 (13.26%)	270 (21.27%)	422 (13.29%)	330 (21.39%)
4-grams	41 (1.55%)	38 (2.99%)	58 (1.83%)	55 (3.56%)
5+6-grams	6 (0.23%)	6 (0.47%)	9 (0.28%)	9 (0.58%)
all grams	2647	1269	3175	1543

Table 2: Data statistics of IR corpus. The concepts included in the final result are agreed by all the three experts before the discussions (i.e., column 1 & 2) and after the discussions (i.e., column 3 & 4).

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 3: Data Statistics of different concepts corpora for IIR textbook

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

Table 4: Corpora Comparison with SKA

91.5% of them are one-to-N ($1 \leq N \leq 3$) relationships, N was set to be 1 and 3.

5.2.4 Experiment Design. 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 vector as the knowledge representation of each book subsection. The similarity between two book subsections is measured by cosine similarity.

5.2.5 Evaluation Results and Discussion. In this section, we provide our results with baseline corpora (refer section 5.1). As can be shown in Table 5 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: MTurk, Expert and IBM, Expert performs best at point 1. This may because the Expert dataset is more similar to SKA corpus (see Table 4). To see how the discussion phrase help improve the quality of the corpus, we also compared SKA corpus with individual annotation before discussion. The results in Table 5 show that the SKA corpus produce better results than all three

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**

Table 5: Document Linking Results under SKA Corpus and the Baselines

datasets produced by expert annotators before the discussion. This provides the evidence in favour of discussion phase, which tries to bring together knowledge from different experts and combine their views to annotate text with concepts. We also observe that each of the three datasets produced by annotators before the discussion perform better than the single Expert performing ad-hoc annotation without code book and discussion. This demonstrates that code book can guide the expert to extract better knowledge hidden in the textbooks and thus improve the 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 [10, 34]. SMs are maintained by observing student work with learning materials and used to adapt system behavior to individual students, i.e., recommend 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 on which students' knowledge is measured. For example, a student practicing elementary mathematics problem might have to understand knowledge units like "Addition", "Subtraction", "Mulitplication" and "Division". Traditionally experts annotate practice activity or learning resource 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 sub sections we will discuss the system used, data collection procedure and experiment details.

5.3.1 System and Dataset. The dataset used for this experiment is collected from online reading platform, Reading Circle [15]. This system was used in a graduate Information Retrieval course. The system provides an active reading environment to the student where

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

Table 6: SM Dataset Statistics

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 the questions. Reading Circle logs all attempt made by the student. The dataset contains students' time spent on reading sections and quiz performance. The dataset includes interactions from 22 students collected for Spring 2016 semester. Details of the dataset are listed in Table 6.

5.3.2 Student Modeling Method. To assess the quality of each corpus, we used its concepts as KUs is to model students reading and quiz behaviour and predict their future performance. To perform this we used Comprehensive Factor Analysis Model (CFM) [36]. CFM is a logistic regression based model which takes students previous performance and students reading behaviour to predict students success rate for a given question. We selected CFM to model student performance as it performs better on Intelligent Textbooks than other state-of-the-art student modeling approaches and also incorporates student reading behaviour which has proved to be beneficial in case of online textbook based learning systems. [18, 37].

5.3.3 Evaluation method. To evaluate performance of CFM on students performance we performed 5 fold cross validation with student stratified folds. Firstly, we randomly selected 80% of students and put all their reading and quiz activity data into training set. Then for the remaining 20% of students all their reading and quiz activity data into training set. The prediction are reported on quiz performance. 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.

5.3.4 Evaluation Results and Discussion.

- **Comparison with other annotation methods :** In this section, we report our results with baseline corpora (refer section 5.1). We ignored the MTurk baseline as crowdsourced annotations were collected for only two chapters, which is not sufficient to model students performance. As shown in Table 7 SKA performs better than both Expert annotation and IBM concepts. This shows that code book based annotation method is better in extracting KUs than simple expert annotation (Expert) and automatic concept extraction method (IBM).
- **Comparison with individual expert annotations before discussion :** To understand the importance of discussion phase of the annotation process we also tried to compare our final annotations with individual annotation before discussion. The results of this phase are listed in Table 7. As the results show, SKA annotations perform better than each of the individual annotators with increase in AUC, which also shows the effectiveness of the discussion phase. One more observation regarding simple annotation scheme is that *Anno w/o Discussion* in SKA approach were able to come up with better concepts than Expert. This is another evidence of the effectiveness of using code book.

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**

Table 7: Results of Student Performance Prediction with SKA Corpus and the baselines

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 inter-agreement among annotators gradually increased by following our procedure. Besides this approach itself, the outcomes of our work include a code book, 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 their models. We compared our SKA corpus against alternatively produced annotation corpora in terms of their performance on two target tasks performed by intelligent textbooks : document linking and student modeling. The results demonstrate the effectiveness of our approach.

While the present work provides the first approach to annotate knowledge for intelligent textbooks, our work left a number of questions open. First, in this work, MTurk crowdworkers were asked to annotate without codebook. It remains to be seen whether the annotation produced by crowdworkers *with* the codebook could reach the quality of the experts. Second, the concepts extracted by IBM automatic approach produce good results in both tasks, which encourages us to explore a hybrid approach which combines the automatic extraction method and the systematic procedure. The automatic extraction method may have potential power of improving the quality of the annotation as well as reducing the annotation load.

REFERENCES

[1] Rakesh Agrawal, Sreenivas Gollapudi, Anitha Kannan, and Krishnaram Kenthapadi. 2014. Study navigator: An algorithmically generated aid for learning from electronic textbooks. *Journal of Educational Data Mining* 6, 1 (2014), 53–75.

[2] Ray Bareiss and R. Osgood. 1993. Applying AI models to the design of exploratory hypermedia systems. In *Fifth ACM Conference on Hypertext*. ACM Press, Seattle, WA, USA, 94–105.

[3] Petra S Bayerl, Harald Lüngen, Ulrike Gut, and Karsten I Paul. 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, Sanibel Island, Florida, USA.

[4] Rafael Berlanga, Victoria Nebot, and Maria Pérez. 2015. Tailored semantic annotation for semantic search. *Journal of Web Semantics* 30 (2015), 69–81.

[5] Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, Nagoya, Japan, 543–551.

[6] Peter Brusilovsky. 2003. Developing Adaptive Educational Hypermedia Systems: From Design Models to Authoring Tools. In *Authoring Tools for Advanced Technology Learning Environments: Toward cost-effective adaptive, interactive, and intelligent educational software*, Thomas Murray, Stephen Blessing, and Shaaron Ainsworth (Eds.). Dordrecht, Kluwer, 377–409.

[7] Peter Brusilovsky, Girish Chavan, and Rosta Farzan. 2004. Social adaptive navigation support for open corpus electronic textbooks. In *Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2004) (Lecture Notes in Computer Science)*, Paul De Bra and Wolfgang Nejdl (Eds.), Vol. 3137. Springer-Verlag, Berlin, Heidelberg, 24–33.

[8] Peter Brusilovsky and Leonid Pesin. 1998. Adaptive navigation support in educational hypermedia: An evaluation of the ISIS-Tutor. *Journal of Computing and Information Technology* 6, 1 (1998), 27–38.

[9] Vinay K. Chaudhri, Britte Cheng, Adam Overtholtzer, Jeremy Roschelle, Aaron Spaulding, Peter Clark, Mark Greaves, and Dave Gunning. 2013. Inquire Biology: A Textbook that Answers Questions. *AI Magazine* 34, 3 (2013), 55–72.

[10] Albert T. Corbett and John R. Anderson. 1995. Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4 (1995), 253–278.

[11] Gülşen Eryiğit, Fatih Samet Cetin, Meltem Yanık, Tanel Temel, and Ilyas Çiçekli. 2013. Turksent: A sentiment annotation tool for social media. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. Association for Computational Linguistics, Sofia, Bulgaria, 131–134.

[12] Corina Florescu and Cornelia Caragea. 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)*. Association for Computational Linguistics, Vancouver, Canada, 1105–1115.

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

[14] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. 2009. Extracting Key Terms from Noisy and Multitheme Documents. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 661–670.

[15] Julio Guerra, Denis Parra, and Peter Brusilovsky. 2013. Encouraging Online Student Reading with Social Visualization. In *The 2nd Workshop on Intelligent Support for Learning in Groups at the 16th Conf. on Artificial Intelligence in Education*. Citeseer, Tennessee, USA, 47–50.

[16] Julio Guerra, Sergey Sosnovsky, and Peter Brusilovsky. 2013. When one textbook is not enough: Linking multiple textbooks using probabilistic topic models. In *European Conference on Technology Enhanced Learning*. Springer, Berlin, Heidelberg, 125–138.

[17] Nicola Henze, Kabil Naceur, Wolfgang Nejdl, and Martin Wolpers. 1999. Adaptive hyperbooks for constructivist teaching. *Künstliche Intelligenz* 13, 4 (1999), 26–31.

[18] Yun Huang, Michael Yudelson, Shuguang Han, Daqing He, and Peter Brusilovsky. 2016. A Framework for Dynamic Knowledge Modeling in Textbook-Based Learning. In *Proc. 24th Conf. on User Modeling, Adaptation and Personalization*. ACM, Halifax, Canada, 141–150.

[19] Anette Hulth. 2003. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP '03)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 216–223.

[20] Xin Jiang, Yunhua Hu, and Hang Li. 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)*. ACM, New York, NY, USA, 756–757.

[21] Alenka Kavcic. 2004. Fuzzy User Modeling for Adaptation in Educational Hypermedia. *IEEE Transactions on Systems, Man, and Cybernetics* 34, 4 (2004), 439–449.

[22] Igor Labutov, Yun Huang, Peter Brusilovsky, and Daqing He. 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)*. ACM, New York, NY, USA, 907–915.

[23] Praveen Lakkaraju, Susan Gauch, and Mirco Speretta. 2008. Document Similarity Based on Concept Tree Distance. In *The 19th ACM Conference on Hypertext & Hypermedia*. ACM, Pittsburgh, PA, USA, 127–131.

[24] Andrew S. Lan and Richard G. Baraniuk. 2016. A Contextual Bandits Framework for Personalized Learning Action Selection. In *the 9th International Conference on Educational Data Mining (EDM 2016)*, Tiffany Barnes, Min Chi, and Mingyu Feng (Eds.). EDM, Raleigh, NC, USA, 424–429.

[25] Tho Thi Ngoc Le, Minh Le Nguyen, and Akira Shimazu. 2016. Unsupervised Keyphrase Extraction: Introducing New Kinds of Words to Keyphrases. In *AI 2016: Advances in Artificial Intelligence*, Byeong Ho Kang and Quan Bai (Eds.). Springer International Publishing, Cham, 665–671.

[26] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. 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)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 620–628.

[27] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic Keyphrase Extraction via Topic Decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 366–376.

[28] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 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)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 257–266.

[29] Olena Medelyan, Eibe Frank, and Ian H. Witten. 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)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 1318–1327.

[30] Erica Melis, Eric Andrès, Jochen Büdenbender, Adrian Frishauf, Georgi Goguadse, Paul Libbrecht, Martin Pollet, and Carsten Ullrich. 2001. ActiveMath: A web-based learning environment. *International Journal of Artificial Intelligence in Education* 12, 4 (2001), 385–407.

[31] Rui Meng, Shuguang Han, Yun Huang, Daqing He, and Peter Brusilovsky. 2016. Knowledge-based content linking for online textbooks. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, Omaha, Nebraska, USA,, 18–25.

[32] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep Keyphrase Generation. In *ACL2017, Annual Meeting of the Association for Computational Linguistics*. ACL, Vancouver, Canada, 836–845. http://arxiv.org/abs/1704.06879

[33] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of EMNLP 2004*. Association for Computational Linguistics, Barcelona, Spain, 404–411. https://www.aclweb.org/anthology/W04-3252

[34] Philip Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance Factors Analysis–A New Alternative to Knowledge Tracing. In *Proc. the 2009 Conf. on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*. IOS Press, Brighton, UK, 531–538.

[35] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic Keyword Extraction from Individual Documents. In *Text Mining: Applications and Theory*, Michael W. Berry and Jacob Kogan (Eds.). John Wiley and Sons, Ltd, United States, 1–20.

[36] Khushboo Thaker, Paulo Carvalho, and Kenneth Koedinger. 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*. ACM, Tempe, Arizona, 111–115.

[37] Khushboo Thaker, Yun Huang, Peter Brusilovsky, and He Daqing. 2018. Dynamic Knowledge Modeling with Heterogeneous Activities for Adaptive Textbooks. In *The 11th Int. Conf. on Educational Data Mining*. ACM, Buffalo NY, 592–595.

[38] Salvatore Trani, Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. 2014. Manual annotation of semi-structured documents for entity-linking. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*. ACM, Shanghai, China, 2075–2077.

[39] Shuting Wang, Chen Liang, Zhaohui Wu, Kyle Williams, Bart Pursel, Benjamin Brautigam, Sherwyn Saul, Hannah Williams, Kyle Bowen, and C. Lee Giles. 2015. Concept Hierarchy Extraction from Textbooks. In *Proceedings of the 2015 ACM Symposium on Document Engineering (DocEng '15)*. ACM, New York, NY, USA, 147–156.

[40] Adam Winchell, Michael Mozer, Andrew Lan, Philip Grimaldi, and Harold Pashler. 2018. Can Textbook Annotations Serve as an Early Predictor of Student Learning?. In *the 11th International Conference on Educational Data Mining*. ERIC, MontrÃĺal, Canada, 431–437.

[41] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 2005. Kea: Practical automated keyphrase extraction. In *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*. IGI Global, Berkeley, California, USA, 129–152.