Automated Matching of ITS Problems with Textbook Content

Jeffrey Matayoshi[1] and Christopher Lechuga[1,2]

[1] McGraw Hill ALEKS, Irvine, CA, USA
[2] University of California, Irvine, Irvine, CA, USA
{jeffrey.matayoshi,christopher.lechuga}@aleks.com

Abstract. As intelligent tutoring systems (ITSs) continue to grow in popularity, many classrooms employ a blended learning model that combines these systems with more traditional resources, such as static (i.e., non-intelligent) textbooks. Typically in such cases, a large amount of manual work is required to match the textbook content with the content covered by the ITS. While resource intensive, this matching is important as it allows the ITS and textbook to work together in a coherent and effective manner. Furthermore, matching the content in such a way lets the textbook take advantage of the adaptivity and sophistication of the ITS; in effect, this infuses the textbook with some measure of intelligence and adaptivity of its own. Given the importance of this matching process, in this work we investigate ways in which this workflow can be automated. To that end, we leverage natural language processing and machine learning techniques to build classification models for matching the textbook and ITS content. We then evaluate the potential performance of these models both as a fully automated system, as well as being used in combination with a human expert.

Keywords: Intelligent tutoring system · Adaptive content · Textbook · Natural language processing · Semi-supervised learning.

1 Introduction and Related Works

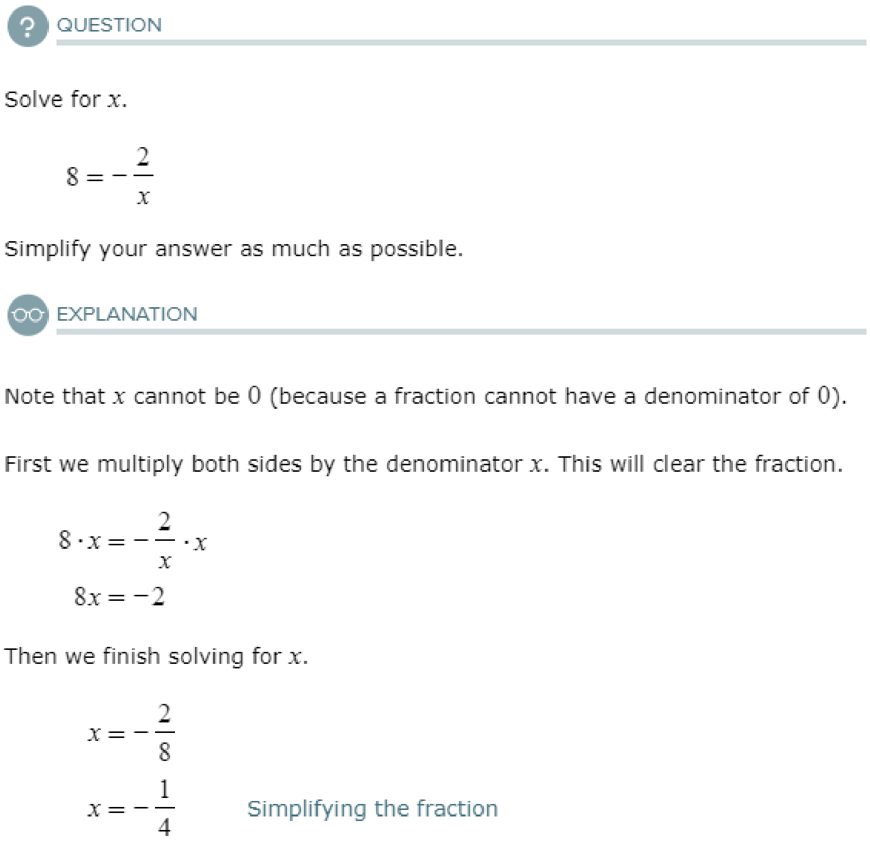
With the growth of intelligent tutoring systems (ITSs) and other digital learning materials, many classrooms operate under a blended learning model. In these classrooms, adaptive online resources can be used in combination with more traditional materials, such as static textbooks. While finding ways in which these various resources can work together is important, in many instances this work is labor and resource intensive. One such example occurs in the ALEKS adaptive learning system (www.aleks.com), in which instructors are given the option of integrating a traditional textbook into the online ALEKS course. The goal of this integration is to give a matching of the ALEKS topics with the content contained in the textbook; this matching allows the instructor to tailor the course to coincide with the content in the book, hopefully leading to a more seamless and consistent learning experience for the student. Additionally, another important benefit of this approach is that it allows a traditional, static textbook to take advantage of the adaptive and intelligent components of the ITS. For example, the ALEKS system keeps track of the topics that a student knows or doesn't know, as well as the topics that the student is most ready to learn. Using the matchings of these topics to the textbook content, an instructor can then see what parts of the textbook the student has already mastered, along with the parts in which there are gaps in knowledge. In essence, by matching the textbook content with the ITS topics, the course sequencing and adaptivity of the system can be used to guide the student through a more personalized path in the textbook, thereby endowing a basic textbook with several of the advantages of a more sophisticated intelligent textbook.

While there are clear benefits to this procedure, the matching of the ITS topics with the textbook content is a laborious task performed by experts who, in addition to needing to be well-acquainted with the subject matter of the textbook, must also be familiar with the specific characteristics and peculiarities of the topics. Furthermore, since the matching must be repeated for each unique textbook and course pairing, this process requires a large amount of manual work. Thus, the goal of this study is to investigate techniques by which this matching of textbook content and ITS topics can be automated.

The problem discussed in this manuscript has many similarities to works such as [9,10,13], which are concerned with the automated extraction of knowledge components; in the case of [9,10], this extraction is performed on problems from ASSISTments, while in [13] the knowledge components are derived from the content of an online textbook. Other previous studies handling similar problems include work such as [2], where an approach is outlined for identifying the core concepts in a textbook section (as opposed to prerequisite concepts which are not the main focus of the section); [5], which attempts to find similar pairs of exercises from within a large set of problems; and [6], which links similar content across textbooks. Our specific approach to solving the problem at hand has perhaps the most overlap with the techniques used in [9,10]. That is, we leverage and apply several techniques from machine learning and natural language processing to build classification models. Additionally, similar to the approach in [2], we also take advantage of the available unlabeled data by experimenting with semi-supervised versions of these models. Once we have developed our models, we then apply them to our specific problem of matching the ITS topics with the appropriate textbook content. While the ultimate goal is for the trained classifier to operate in a completely automated fashion without any human expert oversight, we also evaluate the possible benefits of using a hybrid approach; in the latter case, a human expert is still a necessary component of the process, but the hope is that the classifier can assist the expert and reduce the amount of manual work required.

2 Background and Experimental Setup

A course in ALEKS consists of a set of topics that covers a domain of knowledge. A topic is a problem type that covers a discrete unit of knowledge within this course. Each topic is composed of many example problems that are carefully chosen to be equal in difficulty and cover the same content. An example problem from an ALEKS topic titled "Solving a rational equation that simplifies to linear: Denominator x" is shown in Figure 1. This is one of the first topics covering rational equations that a student encounters in an ALEKS course. For comparison, in Figure 2 an example problem from a more advanced rational equation topic is shown. While this latter topic is more difficult, both topics are typically found in the same textbook chapter and section.



QUESTION

Solve for x.

$$8 = -\frac{2}{x}$$

Simplify your answer as much as possible.

EXPLANATION

Note that x cannot be 0 (because a fraction cannot have a denominator of 0).

First we multiply both sides by the denominator x. This will clear the fraction.

$$\begin{aligned} 8 \cdot x &= -\frac{2}{x} \cdot x \\ 8x &= -2 \end{aligned}$$

Then we finish solving for x.

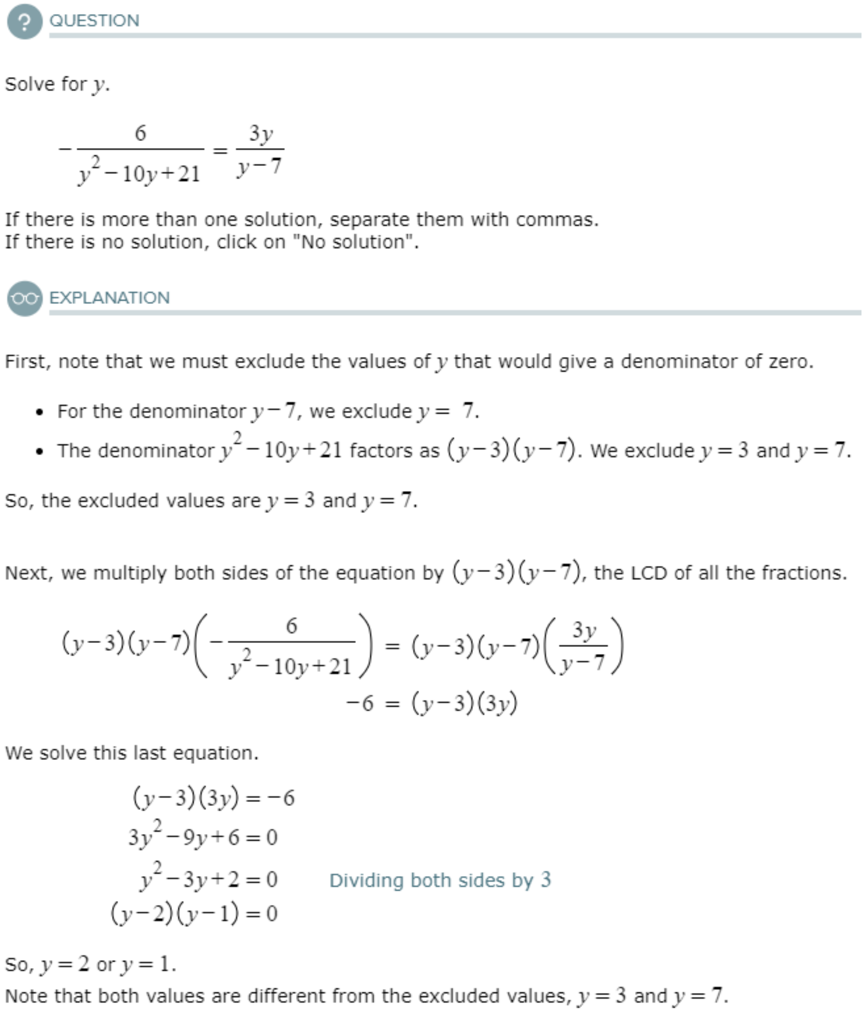
$$\begin{aligned} x &= -\frac{2}{8} \\ x &= -\frac{1}{4} \qquad \text{Simplifying the fraction} \end{aligned}$$

Fig. 1. Screen capture of the question and explanation for an ALEKS topic titled "Solving a rational equation that simplifies to linear: Denominator x." The text from the question, explanation, and title are extracted and used to generate the features for our machine learning models.

The most commonly used courses cover content from math and chemistry, with additional courses available for other subject areas such as statistics and accounting. A course might contain over a thousand topics in total, from which an instructor is free to choose a specific subset of topics; while in rare cases the instructor chooses to use all the available topics, a more typical subset consists of a few hundred. The instructor also has the option of integrating the content of a textbook with this set of topics. When this is done the topics are matched to chapters (and possibly sections) within the textbook. In order to obtain this matching, subject matter experts familiarize themselves with the textbook, and they then use this knowledge to determine the area of the textbook that best matches each topic. This procedure is completed for every available topic in the course, which allows a textbook to be matched with any possible subset of topics that an instructor chooses to use.

The goal of our study is to evaluate an attempt to automate this matching of ITS topics with textbook content. In doing so, we treat this as a classification problem, in which the chapters or sections of the textbook are the class labels. As this automated procedure is designed to emulate the process used by a human expert, our training data are extracted from the textbooks in the following way. Each section in the textbook generates one data point in our training data set; the features (as discussed in more detail shortly) are extracted from the textbook content in that section, while the ground truth label is either the chapter number or the section number. The unlabeled data set then consists of the ALEKS topics. After the model is trained, it is used to assign labels to these topics, and we can then compare these labels to those assigned manually by human experts.

For our experiments we use textbooks from three different college level math courses that are common in the United States: beginning algebra, intermediate algebra, and college algebra. For each of these textbooks, we rely on the matching of textbook content and topics that are currently used in the ALEKS system. Each of these matchings has been created by a single human expert using the following procedure. The expert first familiarizes themselves with each individual topic by reading the question and explanation, and also by looking at several of the example problems contained in the topic. The expert then browses through the book and matches the topic to a specific chapter (and, in some cases, a specific section). From each course we choose one or two textbooks to use as our validation set; the books in this validation set are then used to perform all of our feature engineering and model selection. Once we are satisfied with the features, models, and hyperparameters, we then apply these to one additional book from each math course as our test evaluation. Note that this evaluation on the test textbooks is different from the conventional machine learning workflow, in that we are not applying previously trained models to the books in our test set. Rather, we build new models on each of the books used for our test evaluation, using the best choice of features and hyperparameters from the validation books, and we then apply these models to classify the topics in the course. The reason we use this procedure is that it exactly follows the workflow that would be used in an implementation of the model within the ALEKS system. That is, in such



QUESTION

Solve for y.

$$-\frac{6}{y^2-10y+21}=\frac{3y}{y-7}$$

If there is more than one solution, separate them with commas.
If there is no solution, click on "No solution".

EXPLANATION

First, note that we must exclude the values of y that would give a denominator of zero.

- For the denominator $y-7$, we exclude $y=\ 7$.
- The denominator $y^2-10y+21$ factors as $(y-3)(y-7)$. We exclude $y=3$ and $y=7$.

So, the excluded values are $y=3$ and $y=7$.

Next, we multiply both sides of the equation by $(y-3)(y-7)$, the LCD of all the fractions.

$$\begin{aligned}(y-3)(y-7)\left(-\frac{6}{y^2-10y+21}\right) &= (y-3)(y-7)\left(\frac{3y}{y-7}\right)\\ -6 &= (y-3)(3y)\end{aligned}$$

We solve this last equation.

$$\begin{aligned}(y-3)(3y)&=-6\\ 3y^2-9y+6&=0\\ y^2-3y+2&=0 &&\text{Dividing both sides by 3}\\ (y-2)(y-1)&=0\end{aligned}$$

So, $y=2$ or $y=1$.
Note that both values are different from the excluded values, $y=3$ and $y=7$.

Fig. 2. Screen capture of the question and explanation for an ALEKS topic titled “Solving a rational equation that simplifies to quadratic: Proportional form, advanced.” In comparison to the topic shown in Figure 1, this is an advanced problem that requires the student to solve a complicated rational equation. However, both topics are typically found in the same textbook chapter and section.

an implementation there is nothing that prevents the training of a new model for each textbook; as such, using this testing procedure gives a realistic evaluation of the performance of the model. Additionally, a key characteristic of this procedure is that the features of the topics are available to the model during training.

Each of the textbooks is in a digital format, with each textbook section contained in a separate HTML file. Similarly, each ALEKS topic also has an associated HTML file, which contains the text of the question, the title of the topic, and an explanation which contains a worked out solution to the problem. As we rely heavily on techniques from natural language processing (NLP) to build our machine learning models, we process the HTML files using the following procedure. We first remove the HTML tags and extract the content from each of the files using the Beautiful Soup [12] Python library. Next, we preprocess the raw text by using the the Gensim [11] Python library to perform several standard NLP operations, such as removing stop words, punctuation, and numerals, and stemming each word to its root form.

Once we have processed the text, we build our set of features using an n-gram model. When building the n-gram model, we restrict the vocabulary to a predetermined set of words that have specific mathematical meanings. For example, words such as *add*, *degree*, and *triangle* are kept, while other less informative words (for our purposes) are removed. This procedure reduces the size of our set of features, and on our validation set it also gave a slight boost in performance. From this vocabulary we then extract our n-grams, experimenting with various sequence sizes on our validation set of books. In all all cases we apply term frequency-inverse document frequency (tf-idf) transformations before feeding the features to the machine learning models. When building the vocabulary and features for each textbook model, we also include the text from the questions, titles, and explanations of the topics; as we mentioned previously, in an actual application of this model, such information would be available during the model building process.

The textbooks we consider in our study typically have somewhere between 50 and 80 different sections, and each of these sections generates one (and only one) entry in our labeled set of training data. Thus, because of the limited amount of labeled data, we also experiment with semi-supervised learning models, in which the topics are used to generate the unlabeled portion of the data set. Semi-supervised learning models lie between supervised and unsupervised learning, and such models are unique in that they are able to take advantage of both labeled and unlabeled data [1,16]. This can be very useful in situations, such as ours, where assigning accurate labels to data takes a considerable amount of manual (human) effort. In such a case, adding the extra unlabeled data can possibly give a large increase in the accuracy of the model [7,8].

There are two main goals of semi-supervised learning. The goal of *inductive* semi-supervised learning is to learn a predictor function that can be applied to unseen data that is not contained in the training set; this is closely aligned with the motivations of standard supervised learning algorithms. On the other hand, *transductive* semi-supervised learning aims to learn a predictor function that can be applied to the unlabeled data *in the training set*; by this description, transductive learning has overlap with clustering and other unsupervised learning techniques that are not necessarily concerned with generalizing to future data. When training our models, we treat the problem of classifying the topics as one of transductive learning, where we use our small set of labeled data (i.e., the textbook content) to build a classifier that can help us assign labels to the topics.

Finally, we should also mention that, in the spirit of previous works such as [6,14], we experimented with using similarity metrics to find the textbook content that best matches each ITS topic. For example, in one attempt we used the doc2vec [4] model to generate vector space embeddings for each section of a textbook, as well as for each topic. We then used the cosine similarity measure to find the best match between the topics and textbook content. However, the results were less accurate in comparison to the classification models.

3 Experimental Results

Most likely due to the small amount of training data available for each textbook, the best performance on our validation set came from the simplest models we tried, namely, naïve Bayes and logistic regression classifiers. In comparison, more advanced model architectures, such as neural networks and random forests, were not as effective. Thus, on our test textbooks we restrict our evaluations to the naïve Bayes and logistic regression models; additionally, we also include the semi-supervised version of naïve Bayes described in Section 5.3.1 of [8],[1] as it also performed well on our validation set. For our n-gram models we use both unigrams ($n = 1$) and bigrams ($n = 2$), the combination of which had the strongest performance on our validation set. Adding larger values of n gave slightly worse performance; as before, overfitting on the small amount of training data seems to be the likely reason for this drop in effectiveness.

To evaluate the performance of our classifiers, we use the probability estimates from the models to extract the most likely chapter match for each topic, and we then compare these chapter matchings to those from a human expert. To find the most likely chapter, as determined by the model, we use the following procedure. Each textbook section is assigned a unique label in the training set. We apply the models trained on these labels to the ITS topics and find the textbook section with the highest probability estimate; this section is then mapped to its corresponding chapter to get the most likely chapter match for the topic. Somewhat counterintuitively, training the classifiers with the section labels, as opposed to using the chapter labels, gave stronger performance on our validation textbooks; thus, for this reason we use this (indirect) approach to find the best chapter match for each topic.

As a first evaluation, we look at how well the model classifications agree with the expert classifications. In Table 1 we report Cohen's kappa and accu-

[1] A Python module implementing this model is available at https://github.com/jmatayoshi/semi-supervised-naive-bayes.

racy statistics for each of the books in our test set. As shown there, the naïve Bayes classifiers outperform the logistic regression models on all three books. Additionally, while the difference varies slightly across the textbooks, the semi-supervised naïve Bayes model consistently outperforms the fully supervised version; the stronger performance of the semi-supervised model is consistent with what we observed on our validation textbooks.

Overall, the results in Table 1 are encouraging, as the Cohen's kappa scores for the semi-supervised naïve Bayes models range from 0.722 to 0.791. To get a relative measure of the performance of the classifiers, one of the authors, who wasn't involved in the original matching of the topics, performed a new matching on a random sample of 50 topics from the intermediate algebra textbook (unfortunately, resources were not available to do a more comprehensive comparison). For these 50 topics, 47 labels from this new matching are equal to the original chapter labels, resulting in a Cohen's kappa score of 0.930. Thus, while this comparison is only on a sample of 50 topics (out of the complete set of 802 in the intermediate algebra course), it is evidence that the classifier models are currently not quite as accurate as human experts.

Table 1. Results for the three test textbooks. The Cohen's kappa and accuracy values are computed by comparing the chapter classifications from each model with the chapter labels from the human expert.

		Beginning algebra	Intermediate algebra	College algebra
Number of:	Topics	818	802	962
	Chapters	10	13	9
	Sections	61	69	54
Logistic regression:	Cohen's kappa	0.685	0.738	0.767
	Accuracy	0.720	0.762	0.806
Naïve Bayes:	Cohen's kappa	0.701	0.768	0.788
	Accuracy	0.735	0.789	0.823
Semi-supervised naïve Bayes:	Cohen's kappa	0.722	0.775	0.791
	Accuracy	0.753	0.796	0.826

Our next two evaluations analyze the hypothetical impact of the models if they were to be used in combination with a human expert. That is, rather than fully relying on the models to make all of the classifications, in these evaluations we assume the models are assisting a human expert. We envision a couple of different scenarios in which this could be done. Our first approach assumes that, for a given textbook, the classifier would be used to determine the matchings for which it is most confident; the human expert could then focus their energy on the classifications that are more difficult for the model to determine. To evaluate this procedure, we want a measure of how often the most confident classifications from the model agree with the actual chapter classifications. To that end, we use the following procedure. For a given textbook, we first find the probability of the most likely class for each topic, using the estimates from the semi-supervised naïve Bayes model; based on these probabilities, the topics are then put in descending order. Next, for each positive integer n, where n ranges from 1 to the number of topics in the course, we compute the Cohen's kappa score for the first n topics in the ordered sequence (i.e., the n topics for which the classifier is most confident). For example, employing this procedure with a value of $n = 100$, we are computing Cohen's kappa for the 100 topics with the highest probability estimates.

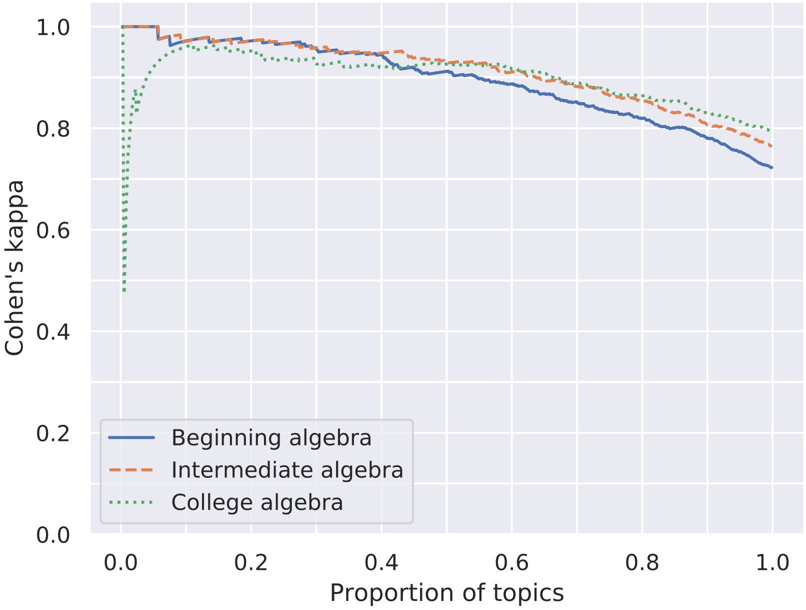


Fig. 3. Plots of the Cohen's kappa scores for the three test textbooks. For each textbook, the topics are put in descending order based on the probability estimates from the semi-supervised naïve Bayes model, and Cohen's kappa is then computed at each point in the sequence. For all of the test textbooks, the plot shows that using the highest 50% of the probabilities returns a Cohen's kappa value of at least 0.9.

The results from this procedure are shown in Figure 3, where we can see the evolution of the Cohen's kappa scores. For example, using the 20% of the topics in each course for which the classifier is most confident (i.e., 0.2 on the x-axis), the Cohen's kappa values are all at 0.95 or above; then, using the 50% of the topics in each course with the highest probability estimates, the Cohen's kappa values are all at 0.9 or above. Based on these results, one seemingly reasonable scenario would be to let the classifier perform the matchings for, say, the 50%

of the topics for which it is most confident, and the human expert then handles the rest.

Our next evaluation assumes that the human expert is involved throughout the entire process, but that they are being guided by the predictions of the model. In this scenario, we envision the expert using the recommendations made by the model to make the matching process more efficient. Specifically, for each topic we assume that the expert is given a list of the chapters in descending order based on the model probability estimates. The job of the expert is to then start with the chapter with the highest probability and check if it indeed matches the topic; if it doesn't, the expert moves on and checks the chapter with the next highest probability, and so on. Thus, in this scenario the model would be useful if, in most cases, the expert would only need to check a small number of chapters.

To that end, in Table 2 we show how often the human expert classification is contained within the n most likely chapters, according to the class probabilities from the semi-supervised naïve Bayes models. That is, for $n = 1$ we simply show how often the chapter from the human expert agrees with the chapter that the classifier gives the highest probability estimate. Next, for $n = 2$, we find the section with the highest probability that is from a different chapter, and we then report how often the human expert chapter label agrees with either of these two chapters. The process then continues for the larger n values. From the results in Table 2 we can see that, the vast majority of the time, the human expert label is contained within the three or four most likely chapters.

Table 2. Statistics showing the proportion of times the human expert label appears within the n most likely chapters, as determined by the probability estimates from the semi-supervised naïve Bayes models.

	Beginning algebra	Intermediate algebra	College algebra
$n = 1$	0.753	0.796	0.826
$n = 2$	0.885	0.895	0.927
$n = 3$	0.925	0.940	0.962
$n = 4$	0.950	0.954	0.974

4 Discussion

In this work we built and evaluated models for automatically associating topics in an ITS system to the most appropriate content from a textbook. We did this by leveraging NLP techniques and machine learning classifiers, and we analyzed the results from both supervised and semi-supervised models. When attempting to match the topics with the appropriate textbook chapters, the results from applying the models showed fairly strong agreement with the human expert decisions, as the Cohen's kappa values ranged from roughly 0.72 to 0.79.

While the overall performance of the machine learning models is encouraging, the argument could be made that the current versions are not quite accurate enough for a fully automated implementation. However, the results from our analyses seemed to indicate that they could be useful in a hybrid application. For example, in all three of our test textbooks, using the 50% of topics from each course with the most confident predictions (i.e., the highest probability estimates) returns Cohen's kappa scores of 0.9 or above. Thus, based on this strong agreement with the human expert labels, a possible procedure would be to use the matchings for which the classifier is most confident, while the human expert then concentrates on matching the remaining topics.

Although this current work has focused on the matching of the topics with the textbook chapters, being able to identify a specific section in the book, while more challenging, would also be useful. One complication with matching topics to specific sections is that the problem becomes slightly more ambiguous; that is, while it is more-or-less straightforward for the human expert to identify a chapter that is a good fit for the topic, this is not necessarily the case with the sections. There are many examples where a topic might seem to fit equally well in two or three different sections; at the other extreme, it may also be possible that no section appears to match the topic.

Another challenging application would be to match the ALEKS topics with the various mathematics education standards that are used for K-12 education in the United States. As with the matching of textbook content, matching the ITS topics to these education standards would enable instructors to leverage the information in the ITS. For example, based on the topics the student knows, the instructor could see what education standards they have mastered and what they still need to learn. However, similar to the matching of topics to textbook sections, this problem also introduces extra complexities, as these education standards can be very specific and narrow in scope.

Fortunately, there are additional modifications to the models that could lead to large performance gains and make the aforementioned problems more tractable. First, the current models are entirely text-based; among other things, mathematical expressions are completely ignored by the models. Thus, a natural next step would be to incorporate equations and expressions into the features of the model, which in many cases could allow for very specific information into the type of material that is being covered. Recent approaches for extracting information from mathematical formulae, such as those used in [3,15], are highly relevant and worth investigating in the context of our current problem.

Second, the ALEKS system contains detailed data on the relationships between the topics in a course. For example, the system has its own clustering of topics into related groups, and it seems reasonable that topics within the same group are more likely to appear in the same area of a textbook (or in related education standards). Additionally, the system has information on the prerequisite relationships between topics. Since it is unlikely that a particular topic would appear after another topic that it is a prerequisite for, this information could be useful in refining the matchings and improving their accuracy.

References

1. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press (2006)
2. Labutov, I., Huang, Y., Brusilovsky, P., He, D.: Semi-supervised techniques for mining learning outcomes and prerequisites. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 907–915 (2017)
3. Lan, A.S., Vats, D., Waters, A.E., Baraniuk, R.G.: Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In: Proceedings of the Second (2015) ACM Conference on Learning @ Scale. pp. 167–176 (2015)
4. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning. pp. 1188–1196 (2014)
5. Liu, Q., Huang, Z., Huang, Z., Liu, C., Chen, E., Su, Y., Hu, G.: Finding similar exercises in online education systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1821–1830 (2018)
6. Meng, R., Han, S., Huang, Y., He, D., Brusilovsky, P.: Knowledge-based content linking for online textbooks. In: 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI). pp. 18–25. IEEE (2016)
7. Mitchell, T.: The role of unlabeled data in supervised learning. In: Proceedings of the Sixth International Colloquium on Cognitive Science. pp. 103–111 (1999)
8. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Machine Learning **39**(2-3), 103–134 (2000)
9. Pardos, Z.A., Dadu, A.: Imputing KCs with representations of problem content and context. In: Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization. pp. 148–155 (2017)
10. Patikorn, T., Deisadze, D., Grande, L., Yu, Z., Heffernan, N.: Generalizability of methods for imputing mathematical skills needed to solve problems from texts. In: International Conference on Artificial Intelligence in Education. pp. 396–405. Springer (2019)
11. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010), http://is.muni.cz/publication/884893/en
12. Richardson, L.: Beautiful soup documentation. April (2007)
13. Thaker, K., Brusilovsky, P., He, D.: Student modeling with automatic knowledge component extraction for adaptive textbooks. In: Proceedings of the First Workshop on Intelligent Textbooks, International Conference on Artificial Intelligence in Education. pp. 95–102 (2019)
14. Thaker, K.M., Brusilovsky, P., He, D.: Concept enhanced content representation for linking educational resources. In: 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI). pp. 413–420. IEEE (2018)
15. Youssef, A., Miller, B.R.: Deep learning for math knowledge processing. In: International Conference on Intelligent Computer Mathematics. pp. 271–286. Springer (2018)
16. Zhu, X., Goldberg, A.: Introduction to Semi-Supervised Learning. Morgan & Claypool (2009)