Generalizability of Methods for Imputing Mathematical Skills Needed to Solve Problems from Texts

Thanaporn Patikorn[0000−0002−9879−2709], David Deisadze, Leo Grande, Ziyang Yu, and Neil Heffernan[0000−0002−3280−288X]

Worcester Polytechnic Institute, Worcester, MA, 01609, USA
{tpatikorn, dodeisadze, lgrande@wpi.edu, zyu, nth}@wpi.edu

Abstract. Identifying the mathematical skills or knowledge components needed to solve a math problem is a laborious task. In our preliminary work, we had two expert teachers identified knowledge components of a state-wide math test and they only agreed only on 35% of the items. Previous research showed that machine learning could be used to correctly tag math problems with knowledge components at about 90% accuracy over more than 100 different skills with five-fold cross-validation. In this work, we first attempted to replicate that result with a similar dataset and were able to achieve a similar cross-validation classification accuracy. We applied the learned model to our test set, which contains problems in the same set of knowledge component definitions, but are from different sources. To our surprise, the classification accuracy dropped drastically from near-perfect to near-chance. We identified two major issues that cause of the original model to overfit to the training set. After addressing the issues, we were able to significantly improve the test accuracy. However, the classification accuracy is still far from being usable in a real-world application.

Keywords: Natural Language Processing· Knowledge Component· Multiclass Classification· Generalizability

1 Introduction

One of the most important skills teachers need to have is the ability to recognize which sets of skills are needed to solve specific problems. While teaching, many teachers solve practice problems as an example. Those problems are also often used as homework practices to help students learn and to allow teachers to be able to measure student knowledge. For students who are unable to reach satisfactory level of knowledge, it is also common for teachers to give a student a few more problems as a chance to show their improvements [9]. Thus, it is important for teachers, educators, content providers, publishers, and researchers to use the same categorization of skills, also known as knowledge components (KCs), as vocabularies for their communication and interaction. Common Core State Standard (CCSS, www.corestandards.org) is one of the most common categorizations

of knowledge components skills in English language arts and mathematics from kindergarten to high school in United States. CCSS provides both broad and in-depth specific descriptions of skills, which is often accompanied by an example problem belonging to the skill. Figure 1 shows Common Core definitions of two 8th grade skills in expressions and equations.

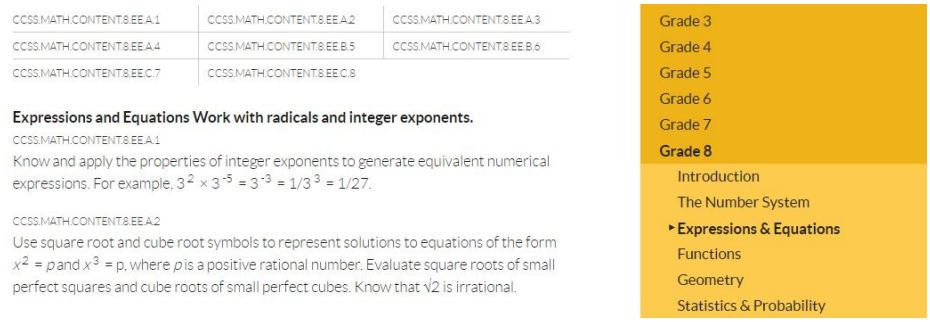

Fig. 1. An example of Common Core definitions from 8th grade skills in expressions and equations.

In recent years, there has been a significant increase of digital devices in the classroom. Such devices enable teachers and students to use learning management systems (LMSs), such as Google Classroom (classroom.google.com) and Schoology (www.schoology.com). These LMS tools are designed to help teachers organize their classrooms and classwork, improve communication between teachers and students, and provide students with help such as instant feedback for their homework. In addition, LMSs allow teachers to easily access course materials provided by other teachers, content creators, or publishers through Learning Tools Interoperability (LTI). As such, skill standards such as Common Core are now more important than ever, as they reduce miscommunication and ensures that teachers can navigate to the right materials, especially through content sharing such as LTI.

Tagging problems with their associated skills or knowledge components are usually done manually, often by experienced teachers and experts in the fields of learning. Identifying knowledge components is also hard, even for experts. In our preliminary study, we had two expert teachers identify knowledge components of a 37-item state-wide math test called TerraNova and they only agreed only on 13 items (35%). In addition, skill tagging is a very tedious and time consuming task [5]. With the growing pools of content created and shared through online systems, there is a need for a method that can identify knowledge components quickly and accurately, using the only information that is consistent across problems from different systems: the content of problems. In the field of machine learning, techniques used to extract information from text are called text mining and natural language processing (NLP). With the interactions of

many systems through LTI, generalizability of models is the top priority since the input problems in real applications may be from different sources, authored for different intents, and come in different formats. Generalizability will be the main focus of this work.

In the field of education, there have been multiple usages of text mining and NLP to help automate laborious tasks. Siyuan et al. applied several types of neural network-based models to automatically grade English essays [10]. Their best model, with Kappa of 0.78, was a variant of a neural network called a memory-augmented network, which can outperform state-of-the-art models like long short-term memory network (LSTM). Decision rules and Bayesian classifiers were shown to be able to automatically assign topics to news stories correctly [1] [6].

Another example of text mining in education is a work by Pardos and Dadu in 2017. The model they presented could accurately assign problems to their associated knowledge components [7]. With five-fold cross validation and 198 different Common Core knowledge components, their best model was able to identify knowledge components with an impressive accuracy of 90%. Their best model was a combination of using skip-gram on the sequence of problem IDs as they are encountered by students, and a neural network on bag of words of the problem text.

In this work, we replicated the methodology presented in [7] on a similar set of problems from the same source they used. In addition, as our work is driven by the need for the models to be accurate both problems within the same system and problems from other sources, i.e. problems created by different teachers, textbooks and publishers. Thus, the main focus of this work is applying the trained models to different sets of problems, and find the best hyper-parameters and preprocessing that allow the model to generalize effectively.

2 Replicability

Our first step to replicate the results from [7] was to obtain a dataset. We chose to use problems from a web-based LMS called ASSISTments (www.assistments.org). We decided to use only certified Skill Builder problems for K-12 mathematics as our training sets because these problems are officially curated and maintained by experts from the ASSISTments system. In addition, each ASSISTments Skill Builder is problem set of a large number of similar problems specifically created to help students learn a specific Common Core State Standard. Thus, these problems match ground truth labels of KCs. The dataset used in [7] also came from ASSISTments, which led us to expect results similar to theirs. Our final dataset includes 65,120 problems from 336 problem sets belonging to 173 different skill standards. The minimum number of problems belonging to a single skill standard is 14 problems and the maximum is 6,480 problems. All problems are formatted using HTML, which is a standard markup for text display in web pages. An example of a problem formatted using html is shown in Figure 2.

2.1 Text Representation and Preprocessing

For text representation, we used the bag-of-word technique similarly to [7]. Bag-of-word is a technique that converts text into a vector representation of the size of the vocabulary. Each element of the vector represents how many times each word in appears in the text. After we get the bag-of-word representation of each problem text, we divide each element of the vector by the sum of all elements. This process is also called L1 normalization.

We decided to keep all html tags and entities in our bag of words. However, unlike in [7] where html elements are used directly as input to the models, we transformed html tags and many symbols that are often used in mathematical problems such as less than (<), greater than (>), and equal (=) into special "words" that corresponds to each html tag and entity. For instance, <img> and π are encoded as htmltag_img and htmlentity_pi. All html attributes (such as links, text colors, and text sizes) and html syntax elements (such as closing tags) are removed. The main idea of this text replacement is to keep relevant special formatting and symbols, such as superscript and subscript, with discarding other information that are not directly related to math knowledge such as text colors and font sizes. In addition, some content sources may use different schemes to format their contents other than html. This transformation would allow models to be used on problems with equivalent formatting.

2.2 Replicated Model

After preprocessing the dataset, we applied various machine learning models. We evaluated each model by calculating its prediction accuracy with five-fold cross-validation. We explored using three different common machine learning models: artificial neural networks, decision trees, and random forests. Artificial neural networks, or ANNs, are models that are inspired by how human nerve cells (neurons) connect and communicate. ANNs have been shown to work very well on text processing, including in [7].

Decision trees are another type of model that have been shown in [1] and [6] to be able to do well in text classification tasks. The main benefit of decision trees are simplicity and interpretability of the models. Since in our dataset, certain skill contains only one sample, we chose to train all decision trees with minimum leaf size of 1.

Random forests are tree-based, ensemble machine learning models [2]. Random forests utilize feature bagging and bootstrapping to achieve both flexibility and prediction powers, while being resistant to overfitting. Random forests have been widely used in many fields from biology [4], text mining [8], and UMAP [3]. We chose the minimum leaf size of 1, similarly to decision trees, with 10 random trees in a random forest.

2.3 Replicability Result

We were able to successfully replicate the results from [7] using the methodology they described. For each method, only the model with the best 5-fold cross-

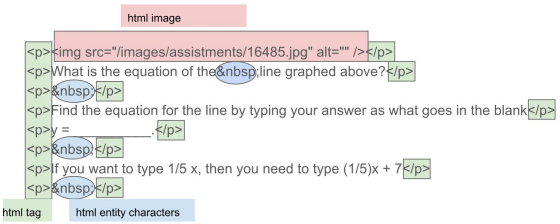

Fig. 2. An example of a problem formatted using html

validation accuracy is included. Our best model, which is shown in Table 1, which uses all the html information of all the problems in the dataset, was able to achieve 92.47% accuracy with five-fold cross-validation in our dataset. Removing html markups reduces the 5-fold cross-validation. While Random Forest is the best model, the 5-fold cross-validation accuracy for the other two models are only 1%-2% lower than that of the Random Forest.

preprocessing	Model	5-Fold Cross-Validation Accuracy	Illustrative Math. Problem Accuracy
keep all html markups	Decision Tree	92.47%	12.14%
keep all html markups	Neural Network	92.07%	13.67%
keep all html markups	Random Forest	92.74%	6.98%

Table 1. Results from using bag-of-word approach with different models using all problems and tranformed html markups.

2.4 Does It Generalize?

Since our goal is to develop a model that can identify knowledge components of problems from different sources and authors, we chose to obtain a second dataset for our generalizability test. We chose problems from Illustrative Mathematics as our test set. Illustrative Mathematics (www.illustrativemathematics.org) is an open and free mathematics curriculum. We chose problems from Illustrative Mathematics because 1) the problems are created with Common Core State Standard in mind, meaning we have the ground truth KCs, 2) it is an open and free educational resource widely used by teachers across the United States, and 3) all problems from Illustrative Mathematics we compiled are in html format similar to the training set. We compiled together 1,581 problems from grade 7 and 8, belonging to 114 different skill standards. The number of problem per skill standard ranged from 4 to 71. We removed all problems belonging to skills outside of our training set. Our final test set contains 392 problems from 23 skills. Afterward, we retrained the model using all training data, and uses on the problems from Illustrative Mathematics. To our surprise, the accuracy dropped

drastically as shown in Table 1. While the performance of all models is still better than chance, it is far from usable in a real-world application.

2.5 Causes of Overfitting

We investigated what could have caused such a massive overfitting by looking through our dataset and models. We found two potential causes of the overfitting. The first cause stems from a large number of near-identical problems. In order to create a large number of math problems, it is common for content creators to create a few templates and substitute different numbers and keywords in the problems and answers. For instance, a teacher might create "Train A from New Mexico to Nevada leaves at TIME_A at SPEED_A mph. Train B from Nevada to New Mexico leaves at TIME_B at SPEED_B mph. The two stations are 900 miles apart. What time will the two trains meet each other?" With this template, the teacher can substitute TIME_A, SPEED_A, TIME_B, SPEED_B with different numbers to create a massive number of practice problems. Specifically, Out of 65,120 problems in our dataset, only 2,523 problems are created without using templates. For the other 62,597 problems, there are at least 1,193 different templates, each of which has been used for more than 10 problems.

Using cross-validation without regard of templates could potentially mislead models to remember the specific words in templates rather than to learn the terminologies of skills, causing their cross-validation accuracy to inflate. In our train example, the model may choose to remember the word "New Mexico" and "Nevada", instead of "mph", "miles", and "time". Pardos and Dadu were aware of this issue of their best model and attempted to solve it by doing cross-validation in such a way that all problems from the same problem set were in the same fold, which significantly reduced their accuracy to around 70%. This approach did not resolve all the issue with templates, since problems created using the same template also exist outside of the problem sets.

The second cause of overfitting stems from the html elements and formatting included in the models. While the html elements are shown to improve cross-validation accuracy in [7], we found that it causes the decision trees to over-prioritize the formatting in the decision. In addition, templates also contribute to overfitting here since the formatting is also copied over to each problem from the same template. This causes the models to be unable to identify the knowledge components once the formatting "styles" have been changed.

3 Towards Generalizability

After we have identified potential causes of overfitting, we came up with multiple ways to address the two issues.

3.1 Near-identical problems

In order to address the issue with templates, we removed all but one problems that are created using the same templates. Luckily, the problems inside our

dataset also contain the information on creation, specifically if it is a (modified) copy of another problem. We used that information to remove all but one problem derived from each template. As a result, the size of our dataset is significantly reduced to 2,474 problems. The number of problem sets and skills remain the same (336 problem sets, 173 different skills). The number of problems per skill standard is also reduced to a minimum of 1 and a maximum of 198 problems per skill.

3.2 html element and formatting

In order to address the html formatting issue, we introduced two approaches to process the html elements. The first approach is to remove all html tags and entities. The goal of this approach is that some formatting schemes may not be equivalent or convertible to html, rendering our first approach unusable. In fact, for some math topics, problems can be written entirely in plain text (i.e. no formatting). This approach is advantageous because the model will be usable on problems of any formats (or no format), albeit often with some loss of information.

The second approach is keep only important html elements. The goal of this approach is to have certain html elements act as keywords which, when combined with other words, can be indicative of the knowledge components. For instance, the words "read" and "graph" together with an image could indicate that in this problem, the student needs to be able to extract information from a visual representation of a graph. In this work, we only include tables and graphs as important keywords.

preprocessing	Model	5-Fold Cross-Validation Accuracy	Illustrative Math. Problem Accuracy
keep all html markups	Decision Tree	62.53%	10.85%
keep all html markups	Neural Network	68.23%	16.26%
keep all html markups	Random Forest	65.24%	12.91%
keep only image and table	Decision Tree	59.71%	12.67%
keep only image and table	Neural Network	63.62%	22.19%
keep only image and table	Random Forest	60.97%	9.56%
remove all html markups	Decision Tree	58.73%	10.33%
remove all html markups	Neural Network	63.80%	22.47%
remove all html markups	Random Forest	61.22%	4.92%

Table 2. Results from using bag-of-word approach with different preprocessing and models using only non-template problems.

4 Results

In order to compare with the models we have in the replication section, we followed the same methodology we used there on each of the different preprocessing

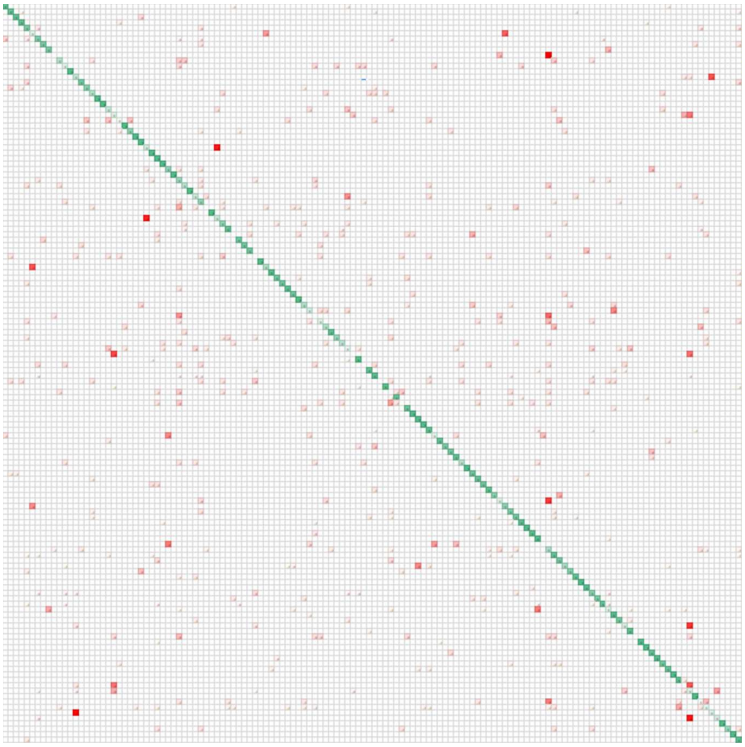

Fig. 3. Confusion Matrix from 5-fold cross validation of models using only non-template problems. Green denotes true positive. Red denotes false positive/negative. The intensity of color is the magnitude of correct/incorrect classification.

approaches to the dataset. The results are shown in Table 2. In general, the 5-fold cross-validation accuracy is much lower than that of the replication, which is to be expected since a large number of near-identical problems are removed. The best model for both training set and test set is a neural network model trained on no html markup at all. The best test accuracy is 22.47%, which almost that of the best replication model. This confirms our suspicion that the near-identical problems and the excessive/irrelevant formatting markups cause the model to overfit.

We also investigated what caused our model to fail. Figure 3 shows the confusion matrix of the our best model (no html markup, neural network) on the training set during 5-fold cross-validation prediction. The green dots indicate where the model correctly classified the problems, the red dots indicate where the model is wrong, and the intensity of the colors is proportional to the percent of correct/incorrect classification. A large number of strong green dots indicates that our models are able recognize the large number different KCs. There are also a large number of red dots, indicating that the model is unable to recognize some KCs, many of which are because there are not enough samples

(e.g. only 1 non-template sample from that KC). Interestingly, not many red dots fall on the same columns. This implies that the model is not quite biased toward any specific KCs. The codes we used in this work can be found here: https://drive.google.com/open?id=1yyJgJavBdAyPbKsFSuhI6TRN_cDRcRM7

5 Conclusion

While cross-validation has been regarded as a gold-standard technique to investigate generalizability of models, in this work we showed that it is also important to investigate generalizability using a separated test dataset that will emulate input data from real application. There are three main contributions of this paper.

The first contribution is the replication of the result from [7]. We were able to successfully replicate their work using the methodology presented. However, after further investigation, we found that the model, while being internally valid and consistent, was unable to generalize well to problems of different sources. We highly recommend future models to not rely solely on cross-validation accuracy for this problem. Instead, researchers should use problem texts obtained through different sources to ensure generalizability. This insight is also applicable to other domains as well.

Our second contribution is the investigation on potential causes of overfitting of models for imputing KCs using only the problem text. We found that near-identical data in the training set as well as html formatting can cause the model to over fit to the "styles" of the training set, reducing its generalizability. We also found that including just the indicators for images and tables do no increase generalizability as we have hypothesized.

The third contribution is an improvement of models for imputing knowledge components using only problem texts. Our best model is a neural network model trained using non-template problems without formatting markups. Our best model was able to correctly identify KCs about 1/4 of the problems in our test set, almost doubled the test accuracy of the model from our replication of [7]. It is important to note that this model is still far from being usable in a real world application.

6 Future Work

There are several areas that we chose not to investigate in this project. For instance, math problems can require multiple skills. In this work, we only choose one of the skills. With conjunctive skill representation, the models can be significantly improved. In this work, we did not individually tune each model. So, it is possible that the models, one finely tuned, may be able to generalize even better than the result presented in this work.

In this work, we did not use any information that Common Core State Standards provided with the skills. For instance, in Figure 1, we treated 8.EE.A.1 and 8.EE.A.2 are two totally different KCs. However, we believe that a model

that can utilize the information on the skill domains and their hierarchies, will perform much better. For instance, a human would be able to recognize that both of those skills are 8th grade Expressions and Equations, and they're also a part of "Expression and Equations Work with radicals and integer exponents".

In addition to improvements to models, another area of future work is to apply the trained models to real applications. One such application that we are currently is a module for ASSISTments that automatically detects the skill standard of problems inside a problem set teachers assigned to students. Then, the system suggests a list of problems belonging to the same set of skill standards as next-day review problems.

7 Acknowledgements

We thank multiple current NSF grants (IIS-1636782, ACI-1440753, DRL-1252297, DRL-1109483, DRL-1316736, DGE-1535428 & DRL-1031398) , the US Dept. of Ed (IES R305A120125 & R305C100024 and GAANN), and the ONR.

References

1. Apté, C., Damerau, F., Weiss, S.M.: Automated learning of decision rules for text categorization. ACM Transactions on Information Systems (TOIS) **12**(3), 233–251 (1994)
2. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
3. Cheng, H., Rokicki, M., Herder, E.: The influence of city size on dietary choices. In: Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization. pp. 231–236. ACM (2017)
4. Immitzer, M., Atzberger, C., Koukal, T.: Tree species classification with random forest using very high spatial resolution 8-band worldview-2 satellite data. Remote Sensing **4**(9), 2661–2693 (2012)
5. Karlovčec, M., Córdova-Sánchez, M., Pardos, Z.A.: Knowledge component suggestion for untagged content in an intelligent tutoring system. In: International Conference on Intelligent Tutoring Systems. pp. 195–200. Springer (2012)
6. Lewis, D.D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Third annual symposium on document analysis and information retrieval. vol. 33, pp. 81–93 (1994)
7. 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. ACM (2017)
8. Svetnik, V., Liaw, A., Tong, C., Culberson, J.C., Sheridan, R.P., Feuston, B.P.: Random forest: a classification and regression tool for compound classification and qsar modeling. Journal of chemical information and computer sciences **43**(6), 1947–1958 (2003)
9. Wang, Y., Heffernan, N.T.: The effect of automatic reassessment and relearning on assessing student long-term knowledge in mathematics. In: International Conference on Intelligent Tutoring Systems. pp. 490–495. Springer (2014)
10. Zhao, S., Zhang, Y., Xiong, X., Botelho, A., Heffernan, N.: A memory-augmented neural model for automated grading. In: Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale. pp. 189–192. ACM (2017)