International Society of the Learning Sciences

Leveraging Students to Generate Skill Tags that Inform Learning Analytics

Steven Moore, Huy A. Nguyen, John Stamper
StevenJamesMoore@gmail.com, hn1@cs.cmu.edu, john@stamper.org
Carnegie Mellon University, Pittsburgh, Pennsylvania, USA

Abstract: Generating skill taggings for learning activities is a task typically performed by instructors and domain experts, which can be challenging to scale due to the required expertise. However, these skill tags are used by many learning analytic platforms. In this work, we investigate the feasibility of leveraging students to generate skill tags in an online learning setting. Across four instances of two online courses in chemistry and programming, we embedded an optional and low-stake activity, where students were asked to propose three skill tags to a given problem in the course domain. We found that 12% of the student-generated skill tags matched expert-generated ones used in the courses while 54% of them were unusable even with revisions. Our results suggest that students are willing to engage with an optional skill tagging activity; however such interventions should be accompanied by scaffolding the process for students to yield higher quality responses.

Introduction

Educational technologies, such as intelligent tutoring systems and digital learning platforms, often employ a mapping of skills to assessment items in order to measure student performance and guide them in the learning process (Bier et al., 2014). These skills treat student knowledge as "an acquired unit of cognitive function or structure that can be inferred from performance on a set of related tasks" (Koedinger et al., 2012). Typically the skills mapped to assessments in these technologies are at a fine granularity intended to measure the low level concepts assessed by a given activity, compared to the higher level skills often found in summative assessments that are often at the standard or learning objective level (Guilbert, 2002). This skill mapping is intended to capture the student's underlying cognitive process and is vital to many core functionalities of educational software, enabling features such as adaptive feedback and hints (Moore & Stamper, 2019).

While having these skills allows for student modeling and can assist in analytical pipelines like open learner models or learning dashboards, developing the mapping of skills to assessments poses several challenges. Traditionally an expert is employed to create the mapping of skills to an assessment, potentially with assistance from an instructional designer or learning engineer (Koedinger & McLaughlin, 2010). This process can be arduous and often requires methods such as think-alouds to elicit the skills from the expert while avoiding pitfalls such as expert blindspot (Nathan et al., 2001). As a result, scaling this process is challenging due to the time constraints, resources for eliciting the skills, and recruiting the domain experts to assist with skill tagging (Matsuda et al., 2015).

Attempts have been made to scale this process, as several studies have turned to crowdsourcing to recruit a pool of humans to skill tag assessments (Moore et al., 2020a, Moore et al., 2020b). While there was surface level success with this pool of human judgement, prior knowledge of the domain is often required to properly generate and map skills to assessments (Koedinger & McLaughlin, 2010). Enlisting students, who have varying domain knowledge and have been successful in related tasks, may offer a viable solution to this problem (Williams et al., 2016).

To explore how we could potentially engage and scale a group of humans in the skill tagging process, we sought to test a novel solution that utilized a more knowledgeable base. To utilize students in generating skill tags, we deployed a completely optional activity where students could input three skills, via free-text boxes, needed to solve a problem in the context of four undergraduate online courses. This study contributes new knowledge about student participation with optional tasks in an online course, particularly one involving skill tagging. It also supports how student performance correlates with the quality of their skill tags. The results provide insights into the feasibility of leveraging students to skill tag problems and challenge us to further investigate how we can deploy interventions that leverage student knowledge to develop stronger learning analytics.

Theoretical framework

A skill refers to a granular piece of knowledge that is required to solve a particular problem in a digital learning environment (Koedinger et al., 2012). These skills power a plethora of learning analytic resources: learning curves, open learner models, adaptivity, and many more (Stamper & Koedinger, 2011). Findings have demonstrated that

digital learning environments that leverage this skill-powered adaptivity, such as cognitive tutors, can help students reach mastery in 26% less time compared to traditional classroom instruction (Koedinger et al., 2012). These systems and the advances they bring all rely on having accurate skill tags, meaning they are written at the correct level of specificity and model part of the knowledge required to solve the problem.

Skills used for these learning analytics systems are more fine-grained than learning objectives or common core standards that typically get mapped to sections or pages of a given course that encompass multiple skills (Badawy et al., 2016). For instance, a learning objective for an algebra course might be "Graph linear and quadratic functions and show intercepts, maxima, and minima". Learning objectives in contrast are fairly broad and encompass several larger concepts that could further be broken down. On the other hand, skills refer to a specific semi-granular subject area associated with a learning resource (Razzaq et al., 2007). An example of a skill tagged to a problem in an algebra course might be "Identify the slope from an equation in the form of y=mx+b". While the skill involves knowing other concepts such as slope-intercept form and arithmetic operations, it is specific enough to be assessed by a single problem and can be encompassed by the larger learning objectives for the course. Note, skills should be relevant to both the problem and the context of the course. For example, while knowing what "+" and "=" mean in the problem is required, a problem in an algebra course would not necessarily be assessing students for that knowledge.

Traditionally, generating and mapping skills to assessments is done by experts that are familiar with the process of skill tagging, their required granularity, and with knowledge of the given domain (Koedinger & McLaughlin, 2010). Frequently when an expert instructor is utilized to create skill mappings, a cognitive task analysis (CTA) or think aloud is conducted (Chipman et al., 2000). In CTA, an expert is interviewed and performs a series of tasks while articulating their thought process, which are intended to specify the cognitive processes associated with task performance. During this process, the interviewer records the skills being articulated by the experts. Studies have shown this leads to effective instructional designs and an accurate skill mapping, but requires substantial human efforts (Lee, 2004). These methods are done so the domain expert involved can potentially overcome expert blind spot, where the explicit steps taken to solve a problem can get grouped together when it becomes second-nature (Nathan et al., 2001). This can impede the skill tagging process, as the expert might otherwise overlook a skill that is innate to them, but one that is critically assessed by the given problem.

Compared to experts, students' knowledge of a domain can be classified towards novice, which may help them avoid some of the pitfalls of expert blind spot, as they can leverage the knowledge of their own previous misconceptions in the learning process (Edens & Potter, 2008). The same CTA methods have been leveraged on novices such as workers on a crowdsourcing platform to provide knowledge components for problems in the domain of geometry and college-level English writing (Moore et al., 2020a). This study had crowd workers complete several assessments and then prompted them to provide the skills, via free-text boxes in a survey, they needed to solve the last problem they completed. They found that roughly half of the skills generated by experts and previously used for the problems in both domains were able to be matched by the skills generated by the participants. While crowdsourcing skills demonstrated baseline success by matching some of the expert skills, the specificity of the skills was not at the appropriate level of detail, perhaps in part due to the lack of domain knowledge of the crowdworkers.

In the present work, we propose utilizing students to help skill tag problems, to create and scale high quality skill mappings that are at the correct level of detail to be utilized by learning analytic systems. As a first step towards this, we must first investigate if students can successfully provide accurate skill tags. To test this, we present students with an optional skill tagging activity embedded in their online coursework. We then utilize a 4-item rubric to assess the quality of students' skill tags and analyze how they relate to both expert-generated ones and the students' performance in the course. The two specific research questions we address are: (1) How accurately can students generate skill tags that domain experts have identified? and (2) How does the accuracy of students' skill tags relate to their participation and performance in the course?

Methods

Learning platform

The present study takes place in a digital courseware platform known as the Open Learning Initiative (OLI). OLI is an open-ended learning environment that offers courses from a variety of domains (such as chemistry, biology, statistics, economics, etc.) and consists of interactive activities and diverse multimedia content (Bier et al., 2019). OLI activities are presented in two distinct categories: low-stakes/formative, providing students with feedback, or high-stakes/summative, used to evaluate student learning at the end of a structured unit. The low-stakes formative assessments in the system are all optional, allowing the students to scroll by them and focus purely on the instructional content. These assessments consist of a variety of question types such as multiple-choice questions, short answer, and dropdown style questions. Each question in OLI is broken down into one or more problem steps,

where each step corresponds to an opportunity for student input. For instance, if a question asks a student to set the value of three dropdown boxes, then it consists of three unique steps that each have their own set of feedback and correct or incorrect responses. This distinction between a problem and its steps is important, since students may work on a problem, but not complete all of the steps due to reasons such as not knowing the answer to some of them, wanting to skip them to save time, or getting distracted (Chen et al., 2018).

Study context & students

For this study, we used data collected from four instances of two different introductory courses, one in chemistry and the other in Python programming. The two instances of the introductory chemistry course were taught at a community college in the western United States. This course provides students with fundamental knowledge of chemistry concepts, preparing them for future biology and chemistry courses. The course is generally geared towards freshman and sophomore undergraduates from varying degree backgrounds, with a majority of the students pursuing a chemistry-related degree. Our data comes from the fall semester of 2020, when the introductory chemistry course was offered in the OLI system.

The OLI content the students used for these two instances of the chemistry course in this study covers the topic of elements and compounds and consists of thirteen separate modules. Each module consists of several topic headers, containing paragraphs of instructional text and low-stakes activities embedded throughout. There are a total of 33 low-stakes and completely optional activities embedded throughout the thirteen modules of the course, not including the skill tagging activity used for this study. These activities include multiple-choice questions, selecting the correct option from a dropdown, drag-and-drop exercises, and submitting a short answer to compare against an expert response. Each of these activities is broken down into steps, depending on the components of the activity, for a total of 178 unique steps. For instance, if a problem has three fill-in-the-blank boxes, then that problem would consist of three unique steps. Every activity and their steps provide students with feedback after they have been answered. Additionally, students have unlimited attempts to answer these questions, so they can continue until they are correct or choose to advance, regardless of a correct or incorrect response.

The two instances of the introductory Python programming course were taught at an R1 university in the northeastern United States. This course provides students with knowledge on introductory concepts in the Python programming language, preparing them for their future coursework in computer science. This course was taught to incoming master's students in a human-computer interaction program who indicated by self-report they did have a sufficient understanding of programming concepts. The collected data we used comes from the summer semesters of 2020 and 2021, when the course was offered in the OLI system. The OLI content in the two instances of the programming course covers the topic of iteration and consists of twelve separate modules. There are a total of 41 low-stakes and completely optional activities embedded throughout the twelve modules of the course. Each of these activities is broken down into steps, depending on the activity, for a total of 113 unique steps.

While the four course instances were taught in different semesters by different instructors, students were provided with the same set of instructions regarding the use of the OLI materials. They were not required to answer the questions found throughout the OLI modules or even access them. Students were provided with an "Introduction to OLI" module, which is an overview of how to effectively make use of the system and the concepts that will be covered in the course. All the instructional materials in OLI were optional to the students; there was no requirement for them to access or complete the materials. However, students were assessed on the concepts covered by the OLI materials, so it was beneficial for the students to utilize them. A further breakdown of the course offerings, including the anonymized instructor, semester, and number of students that accessed the course materials can be found in Table 1.

Table 1, The four course instances used in this study

Course	Semester	Instructor	Student Count	Number of Activities
chem 1a	fall 2020	t1	23	33
chem 1b	fall 2020	t2	26	33
programming 1a	summer 2020	t3	33	41
programming 1b	summer 2021	t4	30	41

Data collection

We focus on an activity we added to this course that involves the students generating three skills that they believe are required to solve a particular problem in the course[1]. In the chemistry course, this activity is found in the ninth module of the OLI content for this section of the course. This module provides several paragraphs of instructional text, two worked examples, and several low-stakes multiple-choice questions on the topic of nomenclature for

ionic compounds with polyatomic ions, along with this single activity. In the programming course, this activity is found in the fourth module of the OLI course. This module covers the concept of python for-loops, consisting of several paragraphs of instructional text, multiple-choice questions, and two small programming exercises. In both courses, the activity is presented in the same low-stakes and optional format as the other activities found prior in the course. This task that prompts students to generate the three skills can be found in Figure 1. In the activity, students are prompted to generate three skills that are needed to solve the problem stated above this task. The students input the text in three different free-text boxes. The instructions for the task are minimal to encourage student participation, as lengthier text might deter students (Schmidt-Weigand et al., 2010). We also provide them with a domain contextual example of what a skill might look like for a different, non-related, problem in[1] the course. Aside from that, no training or scaffolding was provided to the students to help them generate three skills for the question. We intentionally wanted to keep this low-stakes and optional, to examine the students' participation with the task and the quality of their contribution.

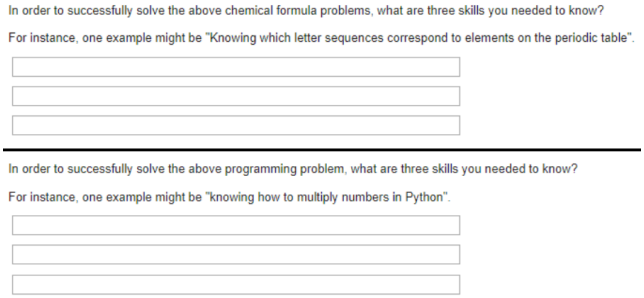

Figure 1. The self-explanation activity and accompanying question presented to the students.

Dataset

Student data was collected from their interactions with the 33 activities found in the chemistry course and the 41 activities in the programming course, in addition to the skill generation task. Since the skill generation task is our outcome, we focus our analysis on the other activities that the students completed in the courses, which consisted of a total of 178 unique steps in the chemistry courses and 113 unique steps in the programming ones. On average, an activity in the course consists of 3 unique steps, such as a single activity having the student select from three different dropdown menus. All of the activities found in the OLI course were completely optional; students could do as much or as little as they desired. For instance, sometimes a student would begin working on an activity, but did not complete all of the parts in it. As a result, the system logs them having worked on that activity and also provides the exact number of steps that they completed. For this data set in particular, it is more common for students to fully complete an activity if they start it, i.e., they will attempt all of the steps of a problem.

Our data also consists of three metrics related to student performance on the activities. When a student works on a step for a given activity, OLI records if their first attempt at that step was correct or not. A first attempt at a problem can be a strong indicator of a student's current understanding of the concepts being assessed (Corbett & Anderson, 1994). Relatedly, the total number of incorrect attempts made at a given step and the total number of correct attempts are recorded. These numbers can potentially exceed the total step count, as a student could correctly answer a question, then select an incorrect answer to see the feedback, then select the correct response once again, registering two correct and one incorrect attempt for that step.

Analysis

In order to assess the quality of the three student-generated skills, we had two item-raters evaluate each skill to determine how much it fit the problem, if it was at the appropriate granularity, and its match to the three skills previously assigned to it by other domain experts. Both raters had content-area expertise, ample experience in skill tagging, and experience applying coding schemes to student data. After familiarizing themselves with the OLI course module once again, they were instructed to go through each skill one at a time and place it into one of four categories. The inter-rater reliability was calculated, and the Cronbach's alpha value was .94, as the raters only disagreed on the categorization of a few skills. These discordant skills were discussed among the two raters until they reached a consensus on the categorization of them using the coding categories. Each of these categories was assigned to a numerical ranking (1-4) that also represented the goodness of the student-contributed skill. In this instance, a ranking of 1 was the best and indicated that the student-generated skill matched the expert-

[1] The full content of this activity is included in https://github.com/StevenJamesMoore/ICLS2022/blob/main/learnersourcing_activities.pdf

generated skill. A ranking of 4 was the worst and indicated that the student-generated skill was not relevant to the problem and therefore it did not match. A full description of the four categories can be seen in Table 2. These four categorizations are based evaluation rubrics from previous studies for assessing math problem solving skills and evaluating knowledge concept maps from novices and experts (Egodawatte, 2010; Kim, 2015).

Table 2. The four categories the student-generated skills were placed into along with an example student-generated skill from the chemistry course.

Category	Rank	Description	Examples
Expert Match	1	The skill matches one of the three skills originally tagged to the problem by an expert.	"Combine ions in the smallest ratio"
Match, Not Granular	2	The skill is very similar to one of the three expert-generated ones, however it could be more specific.	"Know charges on polyatomic ions"
Problem Relevant	3	The skill is technically utilized in the problem, but it is not necessarily what is being assessed given the context.	"Write the formula that reflects this ratio"
No Match	4	The skill is not relevant to the problem, it is not being utilized or assessed by it at all.	"Knowing nomenclature"

The two problems used in each of the courses were selected due to previous student data collected during their use indicating that students tend to struggle on both of them, with 40% of students getting an incorrect answer on their first attempt. Additionally, they each had three skills previously assigned to them by domain experts upon the creation of the courses. Our two item-raters verified that these three skills were appropriate for the problems and at the correct granularity for what was being assessed. While having experts evaluate student-generated skills is not scalable, in the scope of this study, we want to first investigate the feasibility of having students generate such skills. To assess the quality of these student-generated skills, we need to compare their outputs to that of experts that typically do this task. A list of the three skills tagged to each problem can be found in Table 3.

Table 3. Three expert-generated skills for chemistry (left) and programming (right)

Expert Skills - Chemistry	Expert Skills - Programming
1. Identify the cation and anion and their charges	1. Iteration over a value using the range() function
2. Write chemical formulas for ionic compounds that contain polyatomic ions	2. Utilizing a print statement in conjunction with numerical statements
3. Combine the ions in the smallest whole number ratio	3. Modify an existing program to fit a new set of instructions

Results

The student-generated skills were evaluated by experts to determine their quality. Then we analyzed how the student interactions in the course correlated with both student participation on the task and the quality of their contribution. We leveraged measures of central tendency to report the varying categories of the student-generated skills. Then we investigated the different patterns of student participation and performance in the course by looking at their interactions with the varying low-stakes activities and their steps embedded throughout the course. A Bonferroni correction was applied to post hoc analyses that follow (Armstrong, 2014).

Students generating accurate skill tags

Across all four course instances used in this study, a total of 112 students accessed the OLI course. Among those 112 students, 64 of them completed the optional skill generation task where they generated three skills for the presented problem. To assess the quality of these student-generated skills, we had two expert item-raters evaluate all 192 of their contributions. The raters checked if the student-generated skills matched an expert-generated one, indicating that they were accurate, meaning they are written at the correct level of specificity and model part of the knowledge required to solve the problem. This evaluation revealed that 23 (11.98%) of the contributed skills matched an expert-provided item and 65 (33.85%) matched a skill, but needed to be refined for further granularity. Table 4 shows the further breakdown of the item-rater categorizations of the 192 student-generated skills into the four different categories. Over half (54.17%) of the student-generated skills fell into the bottom two categories, which indicates that even with revision they are not usable or particularly useful. All of the skills labeled as the fourth ranking, "No Match", were one word or nonsensical responses by the students for the activity.

Table 4. The count of the four ranking categories assigned to skills in each of the courses.

Course	Student-generated Skill Count	Expert Match	Match, Not Granular	Problem Relevant	No Match
chem 1a	36	7	15	6	9
chem 1b	54	13	24	14	3
programming 1a	54	1	14	29	10
programming 1b	48	2	12	24	10
Totals (%)	192	23 (11.98%)	65 (33.85%)	72 (37.50%)	32 (16.67%)

Each of the activities in the two course domains had three skills tagged to it by domain experts, previously shown in Table 3. Student-generated skills from the chemistry courses were able to match all three of the expert skills for the problem, while only two of three expert skills were matched by students in the programming course. Table 5 shows an example of a student-generated skill from both domains that was categorized as an expert match by the two item-raters.

Table 5. An example of a student-generated skill that matches one of the chemistry expert skills (left) and one of the programming expert skills (right)

Expert Skill - Chemistry	Expert Skill - Programming
Identifying the cation and anion and their charges	Knowing how to use a print statement in conjunction with numerical statements
Student Skill - Chemistry	**Student Skill - Programming**
Identifying the cation, anion, and the charges	Knowing how to manipulate equation to have odd numbers printed instead of even numbers

Student participation and performance

In total, of the 112 students that were part of four instances of the courses, 19 of them did not access any of the course materials in the given modules. A total of 49 students were enrolled in the two chemistry courses, where 38 students accessed the materials, and 30 students did the skill generation activity. The chemistry course consists of a total of 33 optional low-stakes activities, not including the skill generation one, and on average the students completed 26.38 of the 33 (79.93%) activities. The two programming courses had 63 students enrolled in the courses, where 59 students accessed the materials, and 34 students did the generation activity. The programming course consists of 41 optional activities, not including the skill generation one, and on average students completed 30.48 of the 41 (74.34%) activities.

To determine which features of student interaction in the course were indicative of their participation in the skill generation activity, we performed a series of unpaired t-tests on their behavior with the other low-stakes activities found in the courses. This revealed a significant difference between the student participation with the other activities in both the chemistry and programming course and their participation in the skill generation task. An unpaired t-test showed there was a significant difference in the number of activity steps completed by students in the chemistry course who did the skill generation tasks $(M = 164.57, SD = 20.11)$ and those that did not do the task $(M = 39.47, SD = 52.45)$, $t(47) = 11.819$, $p < .0001$. A similar result was found for the programming course, with students doing the skill generation task $(M = 111.03, SD = 18.41)$ and those that did not do the task $(M = 54.00, SD = 40.27)$, $t(61) = 7.699$, $p < .0001$. Students who did the skill generation task were also more likely to complete all of the steps present in the activities embedded throughout the course. Similar significant results were observed for the number of activities done by a student and their participation for the skill generation task in the chemistry course, $t(47) = 11.483$, $p < .0001$ and the programming course $t(61) = 6.299$, $p < .0001$. This result supports the previous one, as the activities found throughout the course are composed of multiple steps and a subset of students completed all the 33 or 41 low-stakes activities respectively.

We then investigated whether student interactions with the other low-stakes activities in the course correlated with the quality of their generated skill contribution, to see how we might predict or promote better skills from the students. While there was a significant difference found between student participation in the skill generation task and participation in the other low-stakes activities throughout the course, it was not found to significantly correlate with the quality of the student contribution in the chemistry course $(\rho = 0.18, p = 0.333)$ or programming course $(\rho = 0.18, p = 0.303)$. Additionally, student performance on the activities as measured by their first attempt correct on the problems was not significantly correlated with the quality, measured by item-raters categorization, of their skill contributions for the chemistry course $(\rho = 0.29, p = 0.119)$ or programming course $(\rho = 0.09, p = 0.615)$.

Discussion

In this research, we investigated how we might enlist students working through online courses to assist in the skill tagging process. We found that even with the task being optional and only providing brief instructions with no scaffolding, some students were able to generate skills that matched experts, ones that could be utilized without any modifications. However, a majority of the generated skills were not sufficiently detailed, and even with revision, they would not be suited for use. The students who chose to participate in the task typically completed all of the other optional activities found in the course. In exploring what features of student interaction in the course correlated the quality of the skills they generated, we did not find any significant correlations with their performance on the other low-stakes activities. These findings suggest that students can generate and tag expert-level skills to problems from an optional and low-stakes activity within an online learning environment, but they might need more scaffolding to consistently do it.

Evaluation of the 192 student-generated skills indicated that, while not the majority, students across both course domains generated skills that matched all three expert ones in chemistry and two out of the three for programming. Students in all four course instances generated many skills that were categorized as a match but needed a bit more granularity to directly match an expert-generated skill. While these levels of skills cannot be as readily used as the direct match ones, they could still serve as an initial baseline to feed into a learning analytics system. We are hopeful that through more instruction or guidance, we can increase the quality of the student contributions by having them think deeply about the granular details of the skills they use to answer the activities.

With all the low-stakes activities embedded throughout the course being completely optional, including the skill generation one, there was still a high amount of overall participation from the students. This was particularly surprising for an optional activity, which generally has lower participation rates due to the lesser perceived value students see in completion of the activity (Farasat et al., 2017). While past skill generation methods have relied on experts (e.g., Koedinger & McLaughlin, 2010), our study presented the task as a low-stakes activity, seemingly fitting in among the fill-in-the-blank and drag-and-drop activities found on the other modules of course content. Leveraging just the native features of the system, in this case free-text boxes for short answer questions, we were able to provide students with the skill generation task seamlessly and without requiring them to utilize yet another platform. It is likely participation would be even greater if the task was required by students or embedded into a high-stakes assessment, such as a quiz question. However, this approach would introduce another series of potential complications, such as requiring it to be graded and potentially introducing an abundance of unacceptable questions contributed by students that do not wish to do the activity but are forced to in the context.

Our contributions should be interpreted against the following limitations. First, our study was conducted across two domains, chemistry, and programming. As skill tagging is directly influenced by the domain of the problems, this may impact the generalizability of the results. Second, students from each domain skill tagged a single problem in each instance of the course. Similar to the domain, the content of the problem is directly related to skills it assesses, so our results might differ depending on the problem or domain. Finally, we did not heavily investigate the background of the students doing the task. It is possible that some students were more familiar with the concepts of skill tagging or articulating this process more than others.

Conclusion

This research demonstrates a first attempt at directly engaging students in the processes of skill tagging problems across the domains of chemistry and programming. Our results highlight how student behaviors regarding their completion of activities in the course are indicative of their participation on the optional skill tagging task. Requiring the student to do the activity, such as putting it in a high-stakes assessment or having students use another system, is not necessarily required for contributions, some of which are evaluated as being on par with expert ones. Although rare, students can provide expert-level skills, without detailed instructions, prior training, or scaffolding. However, most of the skills students generated would not be usable, even with revision. As a result, we do not recommend enlisting students in the skill tagging process through this mechanism, as they might require training or better scaffolding to submit usable skills. This research demonstrates an attempt to help scale online learning analytics and improve educational resources, by leveraging the students in a course. This work opens further opportunities for both engaging students in the process of skill tagging and promoting their behavior that leads to a higher quality contribution for future tasks. Future research should consider expanding domains and experimenting with different amounts of training or scaffolding for the task.

References

Armstrong, R. A. (2014). When to use the Bonferroni correction. *Ophthalmic and Physiological Optics*, *34*(5),

502–508.
Badawy, M., Abd El-Aziz, A., & Hefny, H. A. (2016). Analysis of learning objectives for higher education textbooks using text mining. *2016 12th International Computer Engineering Conference (ICENCO)*, 202–207.
Bier, N., Lip, S., Strader, R., Thille, C., & Zimmaro, D. (2014). An approach to knowledge component/skill modeling in online courses. *Open Learning*, 1–14.
Bier, N., Moore, S., & Van Velsen, M. (2019). Instrumenting Courseware and Leveraging Data with the Open Learning Initiative. *Companion Proceedings 9th International Conference on Learning Analytics & Knowledge*, 990–1001.
Chen, X., Breslow, L., & DeBoer, J. (2018). Analyzing productive learning behaviors for students using immediate corrective feedback in a blended learning environment. *Computers & Education*, 59–74.
Chipman, S. F., Schraagen, J. M., & Shalin, V. L. (2000). Introduction to cognitive task analysis. In *Cognitive task analysis* (pp. 17–38). Psychology Press.
Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, *4*(4), 253–278.
Edens, K., & Potter, E. (2008). How students "unpack" the structure of a word problem: Graphic representations and problem solving. *School Science and Mathematics*, *108*(5), 184–196.
Egodawatte, G. (2010). A Rubric to Self-Assess and Peer-Assess Mathematical Problem Solving Tasks of College Students. *Acta Didactica Napocensia*, *3*(1), 75–88.
Farasat, A., Nikolaev, A., Miller, S., & Gopalsamy, R. (2017). Crowdlearning: Towards collaborative problem-posing at scale. *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, 221–224.
Guilbert, J. (2002). The ambiguous and bewitching power of knowledge, skills and attitudes leads to confusing statements of learning objectives. *Education for Health*, *15*(3), 362–369.
Kim, M. K. (2015). Models of learning progress in solving complex problems: Expertise development in teaching and learning. *Contemporary Educational Psychology*, *42*, 1–16.
Koedinger, K., & McLaughlin, E. (2010). Seeing language learning inside the math: Cognitive analysis yields transfer. *Proceedings of the Annual Meeting of the Cognitive Science Society*, *32*(32).
Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, *36*(5), 757–798.
Koedinger, K. R., McLaughlin, E. A., & Stamper, J. C. (2012). Automated Student Model Improvement. *International Educational Data Mining Society*.
Lee, R. L. (2004). *The impact of cognitive task analysis on performance: A meta-analysis of comparative studies*. University of Southern California.
Matsuda, N., Furukawa, T., Bier, N., & Faloutsos, C. (2015). Machine Beats Experts: Automatic Discovery of Skill Models for Data-Driven Online Course Refinement. *International Educational Data Mining Society*.
Moore, S., Nguyen, H. A., & Stamper, J. (2020a). Evaluating Crowdsourcing and Topic Modeling in Generating Knowledge Components from Explanations. *International Conference on Artificial Intelligence in Education*, 398–410.
Moore, S., Nguyen, H.A., & Stamper, J. (2020b). Utilizing Crowdsourcing and Topic Modeling to Generate Knowledge Components for Math and Writing Problems. *Proceedings of the 28th International Conference on Computers in Education*, 31–40.
Moore, S., & Stamper, J. (2019). Decision support for an adversarial game environment using automatic hint generation. *International Conference on Intelligent Tutoring Systems*, 82–88.
Nathan, M. J., Koedinger, K. R., & Alibali, M. W. (2001). Expert blind spot: When content knowledge eclipses pedagogical content knowledge. *Proceedings of the Third International Conference on Cognitive Science*, *644648*.
Razzaq, L., Heffernan, N. T., Feng, M., & Pardos, Z. A. (2007). Developing Fine-Grained Transfer Models in the ASSISTment System. *Technology, Instruction, Cognition & Learning*, *5*(3), Article 3.
Schmidt-Weigand, F., Kohnert, A., & Glowalla, U. (2010). A closer look at split visual attention in system-and self-paced instruction in multimedia learning. *Learning and Instruction*, *20*(2), 100–110.
Stamper, J. C., & Koedinger, K. R. (2011). Human-machine student model discovery and improvement using DataShop. *International Conference on Artificial Intelligence in Education*, 353–360.
Williams, J. J., Kim, J., Rafferty, A., Maldonado, S., Gajos, K. Z., Lasecki, W. S., & Heffernan, N. (2016). Axis: Generating explanations at scale with learnersourcing and machine learning. *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, 379–388.