NITELITE: Studying Student Language and Teacher Feedback Through Explanation and Asynchronous Collaboration

by

John A. Erickson

A Dissertation

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Data Science

April 2021

APPROVED:

Neil T. Heffernan, Ph.D., Major Advisor

Jacob Whitehill, Ph.D.

Adam Sales, Ph.D.

Adam Kalai, Ph.D., Microsoft Research

Abstract

Online education technologies have provided support for teachers and students in a plethora of ways. Teachers have, and continue, to utilize the automated support; mainly, their automated scoring, feedback messages and student reports. Students benefit from the support of automated scoring, immediate feedback, common wrong answer messages, hints and scaffolding. However, most of the support has been limited to questions with structured answers. Mainly, the support for these systems are widely limited to multiple choice or fill in the blank questions. Questions which have well defined answers. While these provide insights into the students learning and performance; understanding what the mistake was, or why, is based on cumulative common wrong choices. Not every student will process information the same way. While there are many common mistakes, why the student is making this mistake is unique to them. For teachers, the language a student uses provides deep insight into the students' process of thinking, where they may be struggling in the material, or what steps specifically may be causing the confusion. Requiring students to elaborate their work, and write through their steps taken, assists the teacher in identifying each student's unique understanding of the materials and requires a larger range of cognition from the student. For the student, the direct communication through teacher feedback in open response questions provides them with a more personalized explanation to why they scored what they did. However,

for a teacher to score open responses and reply to students, it's inevitably a tedious task. My research aims to utilize student language and teacher feedback to:

1. Develop open response support for teachers with a set of tools which utilize natural language processing to automatically grade and suggest feedback for student answers to open response questions.
2. Evaluate the potential unfairness of these predictive models.
3. Diversify this open response feedback by evaluating the sentiment teachers use in their feedback.
4. Develop open response support for students and evaluate its effectiveness with a randomized controlled trial on NITELITE (Nonsynchronous Integrated Technology Environment - Learning from Interdependent Terminologies and Explanations), a tool for utilizing open response rationale for asynchronous collaborations within intelligent tutoring systems.

Acknowledgements

First off, I would like to acknowledge the person who has not only stood by me through the highs and lows of a Ph.D, but never wavered in her excessive support, motivation, positivity or love for me; the love of my life, Jessica. Additionally, I wanted to thank my family - Fritz, Jan, Jenna, Alex, Abbey, Joe and Sam - for their continuous support throughout my schooling. The continued love and support from all of you helped keep this dream alive.

In the end, I could spend pages and pages detailing all the support, collaborations, discussions, motivations I have received on my path to achieving my dream of obtaining a Ph.D. A dream I have had since I was a little kid. To that end, I wish to send a special thanks to my friends and fellow lab mates:

Anthony	Chris	Neil	Allishah
Avery	March	Jake	Max
Cole	Emma	Cristina	Adam S.
Connor	Aaron	Adam K.	Jack
Hannah	Ashish	Jenny	
Joe	Priyanka	Taylyn	
Jessica	Sami	Erin	
Bob	Ashvini	Nicole	
Cindy	Paige	Mary	

Without you all, I never could of made it to this milestone in my life. I will forever be grateful.

Funding

The work presented in this dissertation has been supported by numerous funding avenues. Specifically, my work has been funded by a Graduate Assistance in Areas of National Need (GAANN) Fellowship. More broadly, my work was funded by NSF (e.g., 1917808, 1931523, 1940236, 1917713, 1903304, 1822830, 1759229, 1724889, 1636782, & 1535428), IES (e.g., R305A170137, R305A170243, R305A180401, & R305A120125), GAANN (e.g., P200A180088 & P200A150306), EIR (U411B190024), ONR (N00014-18-1-2768) and Schmidt Futures.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Online educational technologies have provided automated support for both teachers and students, alike. Teachers have spent years utilizing automated grading, reports and feedback to students. Likewise, students have benefited from automated feedback from teachers, hints, common wrong answer messages, scaffolding, just to name a few. While these have been shown to be beneficial, over the years the content is limited to questions with structured answers. Mainly, support from online educational technologies have been limited to multiple choice or fill in the blank questions; questions where there are easily programmable/identifiable correct answers.

While questions with easily defined answers, such as multiple choice or fill in the blank, have their undeniable advantages (mainly efficiency) [SK05], teachers and students both benefit from a diverse set of question types[Mar99][Ku09]. Teachers are provided a deeper insight into the student's process and their steps taken. Instead of making correlational inferences from previous students, a teacher gains individual insights into the students approach. Within mathematics, for instance, this is beneficial because there are multiple correct approaches (albeit some are more efficient than others) to solves an answers. This is, however, the beauty of mathematics. As long as you follow the correct foundational rules, you can take any series of steps to arrive at the final answer. There are multiple steps and multiple variations within those steps a student can take. By utilizing open response questions, teachers can utilize the student's language to infer their confusion or knowledge.

However, even though there are a plethora of insights to be made within open response answers, it is undoubtedly a laborious task to score and provide feedback to students. While teachers clearly benefit from the deeper insights from open response answers from students, the feedback students receive from teachers are equally as beneficial. In the following chapters, it is clear that teachers aim to utilize open response questions when available. As time continues, the numbers dwindle quickly. What is more apparent is that while the teachers assign open response questions, with the hopes of utilizing that information from student answers, a minuscule percentage of those answers are ever graded or given feedback to.

There clearly is a dialogue missing between students and teachers within open response questions assigned. As mentioned earlier, teachers have had vast support systems when assigning questions with structured answers. Whether it be automated scoring, automated feedback to students or even automatically generated student reports, there is a plethora of support. From this, teachers are drawn to those multiple choice and fill in the blank questions. This, however, is not the case for open response questions. This dissertation looks to utilize modern natural language processing to study student language and teacher feedback within online educational technologies to help open up that dialogue between students and teachers. From these studies, this work successfully developed an automated scoring algorithm and a method for automatically suggesting feedback to students. This can help to open back up that dialogue, by providing teachers with a support system for open response questions similar to that of questions with easily defined answers.

Automatically suggesting feedback can clearly help teachers efficiently score and respond to students, but the question remains on what type of feedback are teachers giving out. Recent advancements in natural language processing has brought powerful pre-trained sentiment analysis tools to provide deeper insight into tonality of language. Not only does this sentiment analysis provide this insight, it can be used to help diversify the feedback suggested to teachers. As a supplementary filtering step, sentiment can be used to suggest differing levels of sentiment in the automated feedback.

Developing support for the teachers is imperative to helping teachers diversify the content they provide students. With that being said, intelligent tutoring systems current support for students, as mentioned earlier, are limited to questions with easily defined answers. When a student works through an open response question, they are on their own. This work looks to utilize collaborations as a means for support for students. Collaborations have been researched for many years and have shown to benefit learners [Gok95] [Gok95]. The question then becomes, how does one utilize collaboration in an increasingly asynchronous learning environment? Within collaborations, it not the live interactions that benefits the students, alone. It's being presented other student's perspectives, answers and rationales. This causes the students to think deeper about their work and the other student's work. Requiring themselves to justify why their work is best or why the other approach may be a stronger method. This processes is what drives the main benefits of collaboration. With that in mind, this work sets out to develop an asynchronous collaboration tool deemed NITELITE. This tool utilizes modern natural language processing, which was discussed earlier, to power NITELITE.

Finally, while all these predictive and exploratory models are all very powerful, attention needs to be paid to any potential unfairness that may lie within these approaches. In this work, all natural language processing, and models built utilizing NLP, use only the text as input. Demographics on the students themselves are left out. However, research has shown that many common approaches, such as working with pre-trained embeddings, can have inherent bias within themselves [BCZ$^+$16]. This doesn't mean models trained with those embeddings elicit bias predictions, however. Within this work, emphasis is put on fairness.

Overall, this dissertation focuses on utilizing student language and teacher feedback to develop support for both teachers and students alike. Each chapter consists of publications and submitted works which all have the overarching goal of utilizing teacher and student language to build such support. All the works then culminate in the final chapter of the dissertation which deploys a pilot study with my tool NITELITE to provide support for students via asynchronous collaborations. This tool embodies aspects of many of the chapter presented. In acknowledgement of all the support I have gotten along the way, each chapter has a citation with a list of authors and the abstract.

Chapter 2

The Automated Grading of Student Open Responses in Mathematics

Erickson, J. A., Botelho, A. F., McAteer, S., Varatharaj, A., & Heffernan, N. T. (2020). The Automated Grading of Student Open Responses in Mathematics. *In Proceedings of the 10th International Conference on Learning Analytics and Knowledge (LAK '20)*, March 23–27, 2020, Frankfurt, Germany. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3375462.3375523

Abstract

The use of computer-based systems in classrooms has provided teachers with new opportunities in delivering content to students, supplementing instruction, and assessing student knowledge and comprehension. Among the largest benefits of these systems is their ability to provide students with feedback on their work and also report student performance and progress to their teacher. While computer-based systems can automatically assess student answers to a range of question types, a limitation faced by many systems is in regard to open-ended problems. Many systems are either unable to provide support for open-ended problems, relying on the teacher to grade them manually, or avoid such question types entirely. Due to recent advancements in natural language processing methods, the automation of essay grading has made notable strides. However, much of this research has pertained to domains outside of mathematics, where the use of open-ended problems can be used by teachers to assess students' understanding of mathematical concepts beyond what is possible on other types of problems. This research explores the viability and challenges of developing automated graders of open-ended student responses in mathematics. We further explore how the scale of available data impacts model performance. Focusing on content delivered through the ASSISTments online learning platform, we present a set of analyses pertaining to the development and evaluation of models to predict teacher-assigned grades for student open responses.

2.1 Introduction

With classrooms progressively adopting free online educational resources (OER's) and curricula, such as Engage New York (EngageNY), Illustrative Mathematics, or Utah Math, a large number of teachers and students are gaining access to expert-authored content. The benefit of using such resources extends to give teachers the ability to assign content aligned with developed standards, supplying them with a range of problems which can be used to provide students opportunities to practice each skill and also can help to assess students' knowledge and understanding of such skills. While the resources themselves provide promise to help teachers gain these benefits, OER's are merely content-based and are not a technology aimed at helping teachers beyond providing the problems and suggested structure of the curriculum.

Conversely, one of the goals of computer-based learning platforms is to help teachers deliver content to students in order to supplement instruction, provide aid to students, and report student learning progress and assessment to the teacher. In doing so, these systems often record large amounts of fine-grained student data to help the teacher make more data-driven decisions in the classroom (e.g. helping to identify which homework problems on which to focus a class discussion). In many cases, this is accomplished through the system's ability to automatically grade student content in order to then report that information back to the teacher.

As open educational resources such as EngageNY, Illustrative Mathematics and Utah Math are more content-focused, and computer-based learning platforms are more instruction-, assessment-, and feedback-focused, the incorporation of OER content into these systems can wed the benefits of each to support both teachers and students. ASSISTments, the learning platform from which we have acquired the data used in this work, is one such system that has incorporated such content. While these learning platforms have many strengths, a current limitation exists in many systems regarding the support for open response questions which comprises a large percentage of the content within these OERs, but of course open-ended problems are not limited to these content sources alone.

The task of automating the grading of student responses to problems in computer-based learning platforms has largely been limited to well-defined or well-structured types of problems. These types of questions include, for example, those problems which have a standard correct response, such as solving a simple mathematical expression (i.e. $6 * 6 = ?$). Likewise, there are those problems which could be represented by a mathematical expression (i.e. solve for x: $8x + 3 = 7$) where the correct answer could take the form of a fraction or a decimal value (i.e. 0.5 or 1/2), where either would be considered correct. While these problem types aim to evaluate students' knowledge of a given topic, questions that require students to explain their reasoning further provide the opportunity to assess students' understanding of the assigned concepts. In order to do so, however, the grader needs to be able to parse and, to some degree, understand the semantics of each response to measure the student's comprehension of the material.

Many of the widely used Intelligent Tutoring Systems, such as McGraw Hill's ALEKS$^{\text{TM}}$ and Carnegie Learning's Cognitive Tutor$^{\text{TM}}$, have no concept of open response questions, likely due to their inability to automatically assess students' responses. Others, such as ASSISTments, do provide a tool for teachers to grade student responses but make no attempt to automatically grade them. While a wide range of automatic short answer grading systems have been developed and documented [BGS15], grading responses to open-ended questions in mathematics remains a task that teachers predominantly do manually.

In ASSISTments, the manual grading of open responses is not common, likely due to the arduous nature of reading and assessing student work. Figure 2.1 illustrates the percentage of open response problems in ASSISTments that are ultimately graded by teachers as well as the percentage of such problems where the teacher has provided feedback (i.e. in the form of a comment or message) for the work. In that figure, it is apparent that less than 15% of assigned open response problems in the system are given a grade by the teacher and even fewer (less than 4%) receive feedback. Furthermore, this percentage decreases over the course of the school year, presumably as teachers realize how much time it takes to properly attend to student responses. This figure illustrates the need to provide teachers with better support in assessing student work.

In this paper, we study the viability and challenges of developing models for the

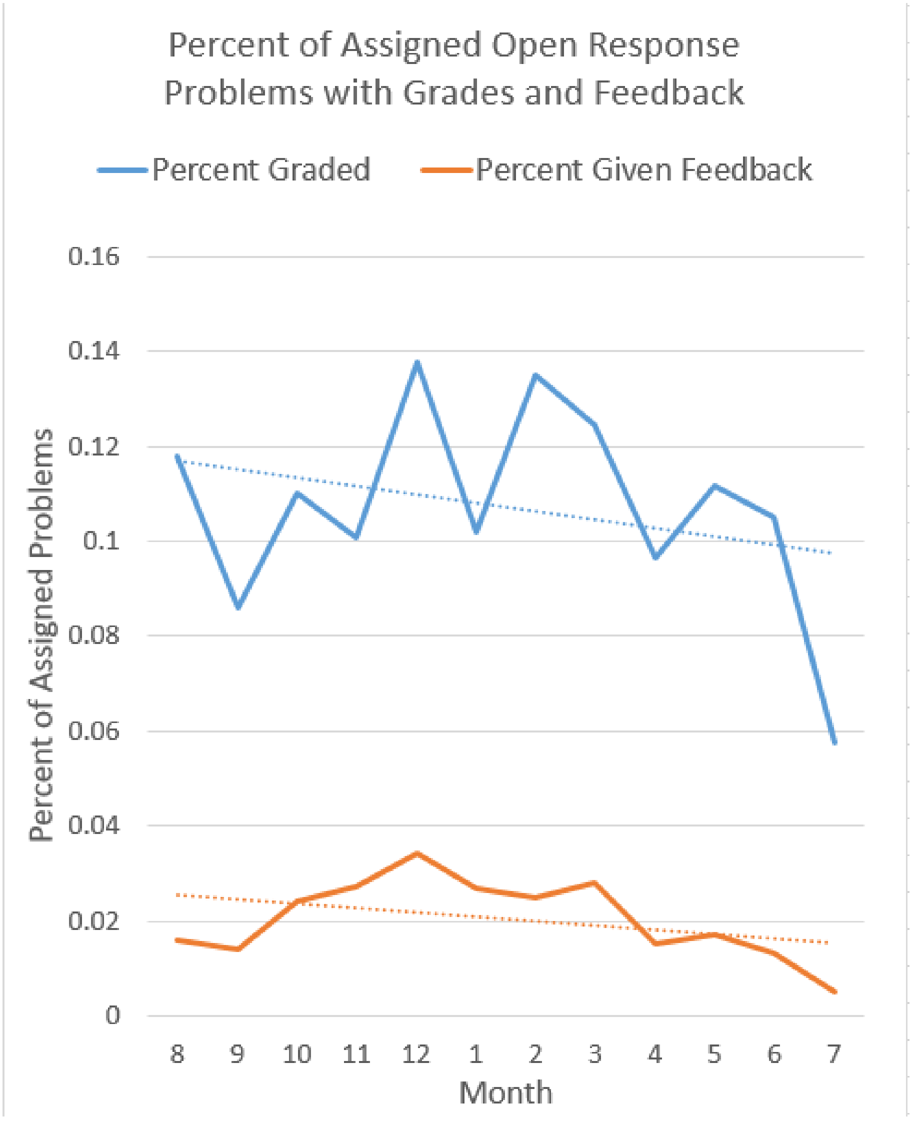


Figure 2.1: Percent of Assigned Open Response Problems with Grades and Feedback

automatic grading of mathematics open response questions using data collected from real teachers assigning content within the ASSISTments online learning platform. Toward this goal, we seek to:

1. Examine variations in teacher grading policies of open responses within ASSISTments
2. Evaluate how well models are able to predict teacher-assigned grades of student open responses given the currently available data from within ASSISTments.
3. Investigate how the performance of our models are affected by the scale of available training data.

The goal of this research is to serve as tool and framework for future studies and experiments involving the automated grading of and generation of feedback for open response questions in computer-based learning platforms

2.2 Background

There have been many previous works utilizing natural language processing (NLP) to provide feedback on responses to open-ended short answer essay questions; the specific NLP techniques used in these works, however, have ranged in complexity in an attempt to extract information from the language. Studies such as [SPR03] have developed systems which use hand-crafted pattern matching to grade one-to-two sentence student responses to open-ended questions. Others, like c-rater [SB09], make use of grading rubrics breaking down scores into multiple knowledge components for evaluation; student responses are parsed to detect the presence of either a paraphrasing of a concept or statements that infer a concept pertaining to such knowledge components. Recent studies like [RHC^{+}17] have also shown promising results using neural network models with no need for feature engineering. In many recent works, several deep learning methods, such as Word2Vec [MCCD13] and GloVe [PSM14], have been used for their ability to capture the semantic and contextual information of words, while another approach has attempted to use memory-augmented networks to better incorporate labeled examples of essays [ZZX+17].

While these deep learning methods have gained popularity for use in NLP tasks, the methods often require large amounts of language data to train and pre-trained models may be limited to words that were in the original corpus (which often excludes the specific math words and symbols that may be found in student responses to open-ended questions). It is for this reason that another, albeit much simpler technique, known as "bag of words" has been applied with some success in certain NLP tasks; this method observes the frequency of each word within and across the given samples, generating a weight measure representing the prominence of that word. While bag of words is a simplistic approach, it is one that has been around for a long time with studies such as [Joa96]. Today, bag of words is the foundation of many studies and strategies. Studies such as Alessandro Sordoni's dynamic context generative models utilize bag of words as an input to their RLMT generating responses from text [SGA+15]. In addition, one of the more common approaches, latent semantic analysis, is based on the bag of words approach, essentially allowing for the comparison of the K-dimensional vectors of two bag of words representations and evaluating the match between these vectors [GWHWH+00].

While most of the discussed non-deep learning approaches utilize bag of words, a known flaw is that the structure of the sentence is not understood. A simple approach is a n_gram model. This will allow the model to save and understand spatial information of the sentences. Studies such as [RYR+16] utilized this approach to create the variables within their logistic regression to predict course completion.

Although the majority of research pertaining to the automatic grading of student open-ended responses has largely focused on non-mathematical content, there have been several works applied within the domain of mathematics. [LVWB15] have explored mathematical language processing for open responses by utilizing clustering methods and bag of words. However, in the case of that study, the focus was on limiting the model to analyze only the mathematical expressions while disregarding text; when the independent variables (i.e. the corresponding prominence of each word) were generated for the model, all non-algebraic text was omitted. This current work, which will be discussed further in the Methodology Section, uses multiple of these approaches including bag of words to include both mathematical expressions and non-algebraic text, and utilizing pre-trained word embedding within deep learning methods to help find the semantics within the open response text.

As it will become more apparent by the description of our data in the Dataset Section, there are several factors that differ between the task described in this work and that of previous related works. These factors can be summarized in terms of the domain of focus, the scale of available data, and the consistency of grading (outcome label). The work of [RHC$^+$17], for example, used datasets from state-level assessments spanning science, biology and ELA (English Language Arts), with an average of 2200 responses for each question; in addition to this, the consistency of labels within the data were arguably more consistent as they were scored by two human annotators. Other studies such as [GF12] and [SSS16], consist of equally large datasets of 80 questions and 2273 student responses (approximately 28 responses for each question) from ten assignments consisting of four-to-seven questions each, and two exams containing ten questions each. Similar to the Riordan study, two human judges were used to score the student responses. In the case of [LVWB15], the dataset consisted of 116 learners solving 4 open response mathematical questions (2 high school level math questions and 2 college level signal processing questions) in an edX course. Similarly, [RYR$^+$16] utilized a single education focused HarvardX course to collect data from 41,946 enrolled students.

This study aims to enable the ability to automatically grade student open responses within online tutoring systems. As discussed prior, support for open response questions on current platforms, such as ASSISTments, are limited and automatic grading is lacking. As shown earlier, the lack of automatic grading leads to a sharp decline in open response questions as the year gets busier and the efficiency of multiple choice questions becomes more enticing. Studies such as [SK05] support this, discussing how multiple choice are prioritized for the ease, accuracy and speed of grading. [Ku09] highlighted the advantages to a wider variety of question types; that providing evaluations of just one question type is insufficient in testing the students true critical thinking and understanding. Other studies [Mar99] discussed how constructed response questions (open response questions) elicit a larger range of cognition's than that of just multiple choice.

In the case of standard essay grading (e.g. pertaining to non-mathematics content and ranging from one sentence to multi-paragraph), there are often very large datasets on which to train NLP models as it is understood that large scale is often necessary; the ASAP Kaggle competition [Pri] is an example of such data that has been made publicly available. Open ended responses in the context of mathematics, however, differs greatly from those observed in other domains as the structure of the language is often secondary to the students' ability to demonstrate knowledge and understanding of the concepts in regard to what is considered when determining appropriate scores. The lack of publicly available data on which to build automated graders of student open-responses, within the domain of mathematics, further makes it difficult for the field to progress in this task. It is for this reason that we not only focus on teacher generated content, but also on OER content. As it is widely used by teachers, supporting an opportunity to make meaningful strides to support teachers on material already being used in real classrooms. While many previous works used a larger pool of data per question or better consistency across labels, our dataset is comprised of student responses to content assigned in true classroom settings by teachers, and is therefore representative of the type of information that would be available to models deployed in such settings.

2.3 Pilot Study: Variations Amongst Graders

Among the largest challenges in developing models to automate the assessment of student open responses is the subjective nature of grading labels. In systems that allow teachers to manually grade responses to open-ended problems, such as in ASSISTments, such teachers are not prescribed a rubric to follow or a set of criteria by which they must assess students; teachers grade their own students based on how well they feel the student has met their own requirements. In most cases, teachers presumably assess students based on how well they are able to articulate and demonstrate their knowledge of assigned content. Others, however, may also grade based on effort, perhaps based on grammar, or even based solely on completeness rather than the content of the response. While some of these cases can be detected (i.e. teachers who only grade based on completion of the problem), other causes of variation are likely more difficult to detect and normalize to help a model learn to assess students on a common scale.

In order to better understand the degree to which teachers' grading policies vary, we conducted a pilot study with 14 teachers[1] who use ASSISTments to regularly

[1]The teachers in this pilot study were recruited through a funded NSF grant

assign content from open educational resources. As it is normally difficult to measure variations in grading due to differences in both assigned content and the wording of student responses, we presented these teachers with a subset of a group of 125 student responses to a set of 3 problems that had been assigned in the previous month; each teacher was given a random subset of 25 responses from their own and other teachers' students plus an additional set of up to 10 anonymized responses from their own students (e.g. if the random set of 25 student responses contained 5 responses from a teacher's own students, an additional 5 responses from their class were selected for that teacher). This selection process allowed for multiple teachers to grade a same subset of student responses as well as re-grade a subset of their own student responses in an anonymized manner (as the pool of responses were selected from those that had been assigned and graded by the 14 teachers previously).

From this data, we were able to calculate inter-rater agreement on the set of teachers to understand how much variation existed in how teachers assess student open-ended work. We apply Fleiss' Kappa as we have more than two raters per response and found that there was just under 17% agreement above random chance on grades between the teachers (kappa=0.167) when assessing on the 5-point scale. When the grades are dichotomized into a binary value (where a grade less than 2 is treated as a 0 and grades equal to or greater than 2 are treated as 1), the agreement rises to 41% above random chance (kappa=0.417). These levels of agreement are surprisingly low, suggesting that there is very large variation in how teachers approach grading these open responses questions. It was also found, when looking at the internal consistency of teachers' grades of their own students, their Cohen's kappa ranged from 23% agreement (kappa=0.231) to over 67% agreement above chance (kappa=0.677); again, these later kappa values are calculated by observing the agreement between the given grades with those previously given for the responses of their own students (all presented anonymously). In following interviews with these teachers, it was suggested that this low internal consistency may be attributable to other contextual factors that are considered when grading students.

The large variation in grades and potential contextual factors that may exist external to the content of a given open response highlight potential challenges faced in developing models that seek to automate this process; such models need to be able to generalize across teachers and students, and the results of our pilot study suggest that this will be difficult. It is for this reason that we include a teacher-level factor in our analysis described in Section 5.4 and discussed further in the Results Section.

2.4 Dataset

For the goal of developing models to automatically assess student open responses in mathematics, we collected a dataset comprised of authentic student answers to open response questions within ASSISTments[HH14] [RFMM16]; while the source of content does vary, a large portion of the open response problems contained within the dataset are from open educational resources such as EngageNY, Illustrative Mathematics, and Utah Math. ASSISTments is used by real teachers and students for classwork and homework, and is developed around the idea of providing immediate feedback to students (on all but open-ended problems) and the reporting of student performance for teachers. As stated in the Introduction Section, ASSISTments has incorporated several OER curricula into its available content, providing the means to collect the student responses to EngageNY, Illustrative Mathematics and Utah Math open-ended questions (as well as others) as they were assigned and graded by teachers.

In the raw and unfiltered state, the dataset consisted of 27,199 unique students with 150,447 total student responses to 2,076 unique problems, and graded by 970 unique teachers. In the data, there were a number of empty responses provided by students caused by either a student submitting nothing (i.e. submitting an answer consisting of only a 'space' character) or by a student submitting an image as their response; images were not included in the data resulting in what appears to be an empty response. As such, any empty responses are omitted from the dataset for the analyses described in subsequent sections, as it is also the case that few would argue that a truly empty response should be given a grade of 0, and the omission of such cases will avoid inflating model performance. Once the filter was applied, the total number of graded student responses dropped to 141,612, the number of unique problems was decreased to 2,042, and left 25,069 unique students and 891 unique teachers. An example of the types of responses and their variations can be seen in Table 2.1. What is clear is that there are a wider variety of responses from students, with inconsistent spelling, mathematical functions written differently and random text. This is one of the main challenges of this study. Another challenge presented in the dataset, and of this study, is that each student response is graded by one teacher with the exception of a small number of samples where multiple teachers assessed the same student responses.

2.4.1 Response Feature Extraction

To support our model development, we take two steps to extract features from the text of the student responses: we first tokenize the student responses and then create a numeric representation of these parsed words using one of several methods. It is common in natural language processing approaches to tokenize, or identify individual words from provided text. For instance, a student may respond with "I

Table 2.1: Sample Responses from Example Problem Selected from Illustrative Math open educational resource

Grade	Example Responses
5	Because B is 2x biggest than A
4	I didn&rsquo;t understand ?
5	Because 2/12 time 2 equals 5
5	2.5 x 2=5
5	2.5 times 2 is 5 so the scale factor is 2 oops that is what I meant
5	Cause 2.5 divided by 5 is 0.5
3	Because the top one is 2.5 and 1.5 goes to 2.5
1	I guessed
3	Because the part on a is half the size of the one on part b.
5	2.5 times 2 is 5.
5	A has 2.5 on top and B has 5_2.5 x2is withc means that it was 2
2	I said that because two of them are equal

didn't know the answer, so I guessed 4" where the text would be divided into each component; a simple approach here would be to simply split the text using spaces, but other approaches may attempt to additionally separate punctuation or contractions into separate components. Within this analysis, the text is tokenized utilizing two different approaches: what is known as standard count vectorizer splitting, as well as the Stanford Tokenizer[MSB+14]. This later tokenizer was applied to better support our deep learning approach described later in this section.

To describe the standard count vectorizer, this approach will take our full corpus of responses, split the words and create a list of those words. Table 2.2 shows an example of the initial processing to extract words/features from student responses. From there, the text which is being trained on is passed through this list, creating a RxW matrix where R is the number of responses by students in the training set, and W is the number of unique words within the overall corpus. Then in each column of W, a count of the occurrence of the word in the student's response is tallied.

In the end, the final RxW matrix acts as the bag-of-words approach described

Table 2.2: Example Tokenization: Standard Count Vectorizer

Raw Student Text	Count Vectorizer Tokenizater	Stanford Tokenizer
“The answer couldn’t be 6x4 = 6” “skies are blue” “It’s something I dont understand” “I didn’t know, so is this right? x+4=8 6x1”	[“6x1”, “6x4”, “answer” “are” “be”, “blue”, “couldn”, “didn” “dont”, “is”, “it”, “know” “right”, “skies”, “so”, “something”, “the” “this”, “understand”]	[“The”, “answer”, “could”, “n’t” “be”, “6x4”, “=”, “6” “skies”, “are”, “blue”, “It” “’s”, “something”, “I”, “dont” “understand”, “I”, “did”, “n’t” “know”, “,”, “so”, “is” “this”, “right”, “?”, “x” “+4”, “=”, “8”, “6x1”]

in Section 2; by adding the frequency values, a numeric representation is given to each word describing its weight amongst other words in the corpus. While this representation approach could result in undesirable omissions, where, for example, the method may partition an equation contained in a student response (as shown in Table 2.2 with *6x4 = 6* recognized as simply *6x4*); it does, however, allow for more flexibility in capturing similar numeric occurrences. As is the case in a bag-of-words approach, the ordering of words within each response is not maintained by the representation and instead relies on a measure of word prominence. With just the count, it is apparent that certain 'stop' words such as 'i', 'me', 'my', 'it', 'this', 'that' would carry more weight given that the words are used often. To combat this, the term frequency-inverse document frequency (tf-idf) statistic is calculated across the matrix. These features will later be used in the non-deep learning models described in the next section.

The other approach to extracting the features from text utilized in this study is the Stanford Tokenizer [MSB+14] combined with Global Vectors for Word Representation (GloVe)[PSM14]; GloVe word embeddings, pre-trained on large datasets, have been made openly available to researchers conducting natural language processing research. In the pre-trained embeddings used in this work, the Stanford Tokenizer was used in the generation of such word representations, so the same tokenizer is applied to maximize the number of words recognized by that model. The Stanford Tokenizer was applied, which increases the amount of words which are able to be respresented by a GloVe vector. For example, the Stanford Tokenizer will represent "didn't" as "did" and "n't" which is necessary for the pre-trained GloVe model to recognize each component (i.e. there is no pre-trained GloVe representation for "didn't" but there are representations for "did" and "n't"). We used the 100-dimensional GloVe vectors pre-trained on a large Wikipedia dataset with the hypothesis that such a corpus is more likely to include mathematics terms than other pre-trained models using, for example, news sources.

2.5 Methodology

To develop our models, we use both traditional machine learning techniques and more complex deep learning algorithms combined with natural language processing approaches. The range of models observed in this work is intended to compare models of varying complexity and flexibility in regard to how such models represent the presented data. Specifically, we compare two decision tree-based models with a deep learning network within the context of a probabilistic baseline model.

With the tree-based machine learning approaches including random forest and XGBoost, each described within this section, there is a decrease in flexibility of the model (in comparison to deep learning algorithm's such as a neural network or LSTM), but greater likelihood in being able to interpret results and identify impactful words/equations within the student's response in order to justify the model's prediction (a potentially desirable quality of a model that will be suggesting grades to teachers). With the inclusion of deep learning, our analysis is taking advantage of the newest approaches and allows us utilize embedding's to help our models understand the semantics of the words/equations within the student's response. Additionally, the final models utilize the predictions from the traditional machine learning and deep learning models as covariates within a Rasch model. The following sections detail the methodologies applied to address the goals outlined at the end of the Introduction Section.

2.5.1 Random Forest

While there has been an expansion of deep learning models (and we attempt as well) within natural language processing, mathematics student open responses are not necessarily comparable to the corpus in most prior analyses. For example, the datasets made available through competitions (cf.[Pri]) have largely focused on non-mathematics content to which others have been able to explore a range of methods including that of deep learning[TN16]. However, the differences in data sources may be worth noting in comparing this to prior works. Namely, many deep learning methods, particularly those using pre-trained embedding models as we describe later in this work, are unable to effectively represent numbers and equations well in the context of other words; while such representations recognize some numbers, the corpus is limited. It is for this reason that a bag-of-words type of approach begins to make more sense as it is easier to train such a method to recognize all words within our specific context. Additionally, the tree-based methods likely require fewer training examples than a complex deep learning model but still offer a large degree of non-linearity in their representations of data.

In regard to the random forest model explored in this work, as discussed earlier, the input of this model is the term frequency inverse document frequency value. This assists in lowering the weight of less important stop words. We allowed the forest to contain 100 decision tree's. By having a more slightly more robust dataset, there is less of a chance of over fitting. Additionally, this allows the forest to identify as many important words within the student responses. For each of the 100 trees, pruning is not performed. This allows for each of the trees to expand out and identify as many words as impactful. Once again, this does bring in the risk of overfitting. Training and testing is performed with a 10-fold cross validation. The model then output a probability that the grade would belong to each of the 5 categories. These probabilities are then used as covariate within the final Rasch model described later in this section.

2.5.2 XGBoost

Continuing with the tree approachs, XGBoost was another flexible model applied. This method will apply gradient boosted decision trees. As [CG16] describe, there are three parts to the tree boosting. First, a regularized learning objective is calculated to prevent overfitting. It starts by calculating a prediction thru summing all the independent tree structures and leaf weights from ensembled decision trees. From there, the model attempts to understand what were the effective set of functions learned within the model by minimizing the loss function. This function is calculating the difference between the predictions and the targets, while attaching a complexity penalty. By attaching this penalty, as the authors discuss, the final weights of the ensembled trees are smoothed to avoid overfitting.

From there, the model aims to optimize the ensembled trees, but the authors [CG16] noted that with functions as the parameters, a traditional euclidean space is not able to be used for the optimization. This lends itself to an additive modeling approach by adding more functions and calculating the loss function. The model adds the functions which minimise the loss function and most improves the current model.

Additionally, the model aims to combat overfitting by utilizing shrinkage, also commonly referred to as regression to the mean. As the authors note[CG16], by utilizing the shrinkage, it can help to reduce how much influence each tree has on the overall ensembling. This then can help create more room for additional, potentially stronger trees, thus improving the model while reducing the chances of overfitting. Lastly, the XGBoost takes one aspect from the Random Forest model, and that is feature subsampling.

Similar to the Random Forest, the term frequency inverse document frequency matrix is used as input to the model. We set the model to perform multi-class classification with a softmax probability learning task. This then allows the model to produce a separate probability for each possible grade. Once again, all training and testing is performed using 10-fold cross validation.

2.5.3 LSTM

The final student grade prediction model for comparison is a deep learning long-short term memory (LSTM) network [HS97]. As mentioned previously, deep learning has been on the forefront of recent advancements in natural language processing. Such models differ from the more traditional methods described above in that such networks consider the ordering of words. Contrary to approaches using a bag-of-words approach, LSTMs recognize that the ordering of words may contribute to the interpretation of the responses and considers this within the network structure.

Before modeling, each sentence is first processed to remove unwanted characters such as line endings. Next, stop words are removed from each sentence to help reduce the sentence to only the most representative words; by shortening the sequence of words it is also believed that the model will be able to more efficiently learn from the data in that it will not need to learn to ignore such common words. From here, each sentence is tokenized using the Stanford Tokenizer and subsequently vectorized using the pre-trained 100-dimensional GloVe embeddings described in Section 4.1.

We apply a bi-directional LSTM model consisting of 3 layers: a 100-dimensional input layer that accepts the pre-trained GloVe vectors of each word, a 40-node hidden LSTM layer (20 nodes that observe the sequence in order and 20 nodes that observe the sequence reversed), and finally a 5 node output softmax layer corresponding to each of the 5 possible grade values with a cross-entropy loss applied. The application of a bi-directional network is believed to help the model learn order dependencies between words as well as help it learn more prominent long-term dependencies at the beginning and end of each response.

The model is trained using a 10-fold cross validation with an Adam optimizer with a step size of 0.03. The small step size combined with the comparably small network size (it is not uncommon for such networks to contain many more layers with hundreds of nodes per layer) is meant to help reduce model overfitting; while the overall dataset is arugably large enough to support deep learning models, a separate model is trained per problem which, in some cases, exhibit smaller sample sizes than would normally support a deep learning model. However, given the model size and the pre-trained nature of the GloVe embeddings (the model does not need to learn new word representations), we feel that the application of this model is justified for the given prediction task.

The model is trained using a variable stopping criterion based on the performance of a holdout validation set consisting of 1-fold's worth of data (approximately 1/9th of the available training data). The model is trained over many epochs, or cycles through the training data, until the performance on the validation set plateaus.

Similar to the other previously described models, the LSTM treats each grade as mutually exclusive classes for training; despite the ordinal nature, this aspect of the output is not explicitly included in the model

2.5.4 Rasch Model

While the previously described machine learning and deep learning models are the focus of comparison for this work, we utilize one final model as a baseline and a means of more fairly evaluating the performance of the previous models. For this,

Table 2.3: Rasch Model Performance

Model	AUC	RMSE	Kappa
Rasch Model with teacher component	0.696	1.09	0.162
Rasch Model without covariates	0.827	0.709	0.370
Rasch Model with number words covariates	0.829	0.696	0.382
Rasch Model number words and Random Forest covariates	**0.850**	**0.615**	**0.430**
Rasch Model number words and XGBoost covariates	0.832	0.679	0.390
Rasch Model number words and LSTM covariates	0.841	0.637	0.415

we use a two- and three-component Rasch model. A Rasch model, commonly applied in item response theory (IRT), is a probabilistic model (in this case, a variational bayes model) that uses fully connected data to learn components that describe the users and content independently of each other. In IRT, it is common that such a model may learn a student ability parameter for each student as well as an item difficulty parameter describing each problem. In our specific application, we use the Rasch model to learn a student ability parameter, item difficulty parameter, and, for an additional comparision, a teacher strictness parameter following the results of our pilot study described in Section 3.

The formulation of the Rasch model is as follows:

$$\begin{split} grade = ordered\_logistic(student\_ability - item\_difficulty \\ + \beta * X) \end{split} \tag{2.1}$$

$$\begin{split} grade = ordered\_logistic(student\_ability - item\_difficulty \\ - teacher\_strictness + \beta * X) \end{split} \tag{2.2}$$

In the first Rasch model in Equation 2.1, it is formulated as an ordinal logistic regression observing a learned value per student and a learned value per problem.

In addition to this, a set of covariates X will be used to evaluate the previously described machine learning models. Since the base Rasch model, where X is an empty matrix, observes no information of the problem text itself, a model that is able to effectively learn from student response text should lead to notable improvements in model performance. It is for this reason that we use this model as a means of comparison. The predictions of each of the previous models will be incorporated into the Rasch systematically to compare the added benefit (if any) beyond the attributes of student ability and item difficulty. For example, the 5 predicted probabilities produced by the LSTM model are presented to the Rasch model as X and the performance of such a model is compared to the Rasch model without such covariate data (as well as compared to the Rasch model containing the other machine learning predictions).

The Rasch model in Equation 2.2 incorporates an additional learned parameter of teacher grading strictness, in observance of the results of our pilot study. By including this term, we should gain an understanding of how well such a model is able to perform when observing that different teachers grade with different policies, particularly in regard to being more or less strict (i.e. a less-strict grader may apply higher grades on average than a more-strict grader).

In both of these cases, the Rasch model helps to observe the model performance independent of student "goodness" and item difficulty that may otherwise inflate or deflate model performance. In addition to this, the Rasch incorporates an ordinal regression which was not observed by any of the other machine learning models; as such, the combination of the two methods holds promise to produce better results by observing the ordered relationship between grades.

As one additional baseline model, we include the Rasch model with a single covariate representing the number of words in the student response. It seems plausible that longer responses may, on average, receive higher grades, so we include this term alongside the others as a more appropriate baseline of comparison.

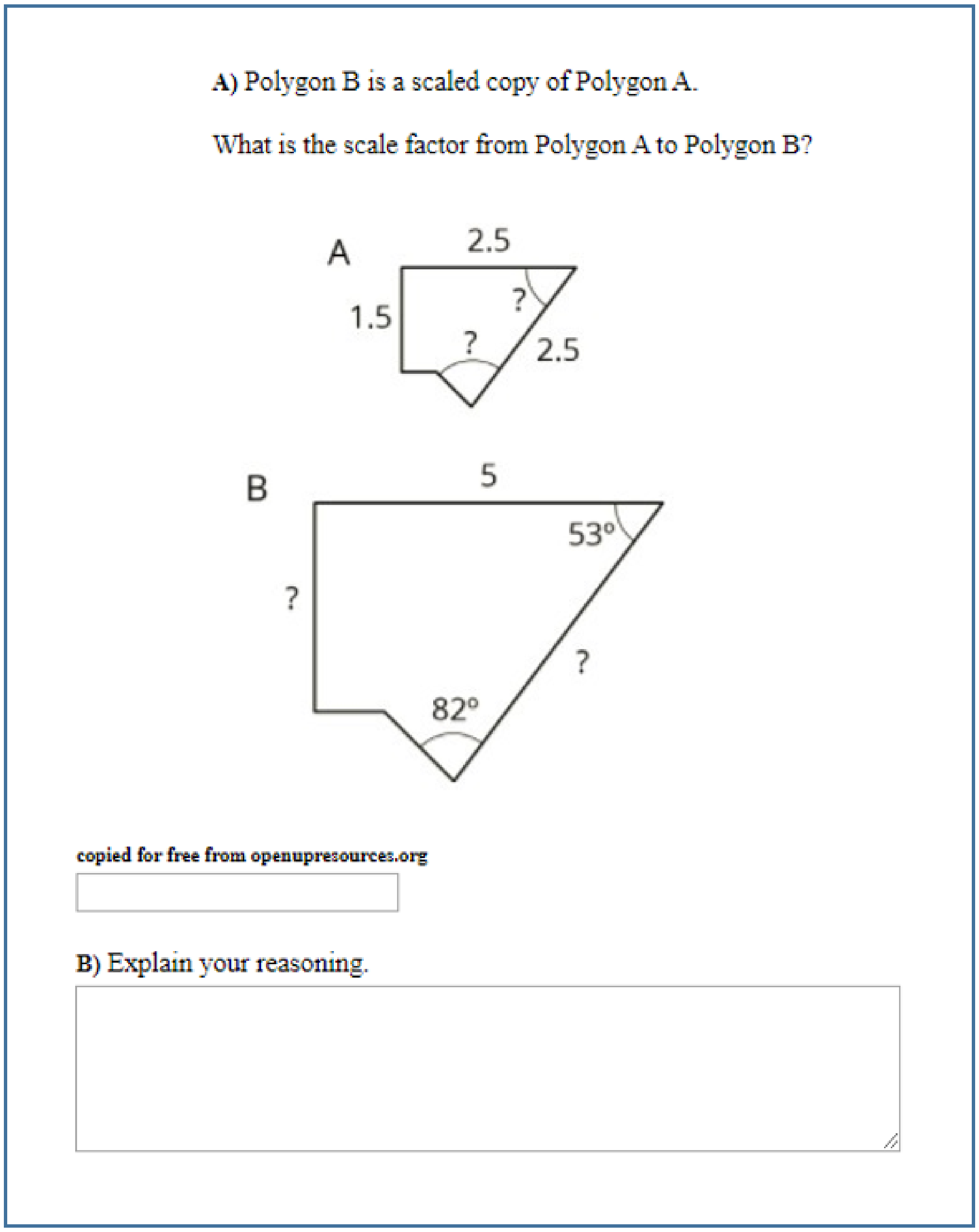
2.6 Results

We report three evaluation metrics with which to compare each model: AUC, RMSE and Cohen's Kappa. AUC is calculated using a simplified multi-class calculation of ROC AUC [HT01], where values close to 0.5 represent performance at chance and values close to 1 represent higher performance. RMSE is calculated as the root of average squared errors when observing the ordinal predictions and labels (i.e. observing that the difference between a prediction of 3 and an actual grade of 4 is a value of 1); this differs from the other metrics that observe the 5-point labeling scale as a multi-class classification problem. Finally, we observe multi-class Cohen's kappa as a measure of inter-rater agreement above random chance (observing that some labels such as 4 and 0 appear more frequently than others.

Overall, each of models managed to predict student open response better than the simple Rasch model baseline. The baseline model, of just a basic Rasch model without any additional covariates, managed to classify students' open response grades with an AUC of 0.827, as shown in Table 2.3. In fact, all models manage to classify student grades with an AUC greater than 0.820 aside from the Rasch model incorporating a teacher strictness component; it is possible that the model is either unable to learn three parameters from the given data or the influence of variations in teacher grading are not as impactful as the pilot study suggested. Likewise, aside from the identified Rasch model, the Kappa values are moderately high, all models are able to classify and predict the student's grade at least 37% above chance.

However, its is apparent that the incorporation of the machine and deep learned grade prediction covariates provide the Rasch model with insight previously not identified. While the LSTM and XGBoost manage to improve the models performance, Random Forest managed to provide the most additional insight to the Rasch model. What is also evident is our model's ability to become more confident in our predictions with more covariates. Our RMSE manages to drop in the Rasch model with any of our additional covariates and, once again, the Random Forest managed the lowest RMSE.

In the end, it is clear in Table 2.3 that the best overall model was the Rasch model with Random Forest covariates. This model was able to classify/predict student's open response grades with an increase in the AUC of 0.023, a drop in error rate (RMSE) of 0.094, and an increase in the kappa of 0.060 over the baseline Rasch model without additional covariates.



A) Polygon B is a scaled copy of Polygon A.

What is the scale factor from Polygon A to Polygon B?

copied for free from openupresources.org

B) Explain your reasoning.

Figure 2.2: Example Problem Selected from Illustrative Math open educational resource

2.7 Exploration of Sample Sizes

While the results in the previous analysis suggest that the models are performing moderately well in comparison to our baseline, this research aimed to explore the impact the amount of data has on our performance. It is unclear if, given more data, we would expect to see large increases in model performance. We selected a problem from our dataset to exemplify this process here, but it is intended that this analyses can be repeated on all problems to assess the impact of available data at a finer level of granularity. Of the 10 problems with the closest grade distribution to that of the overall population, we selected the problem with the largest sample size (shown in Figure 2.2).

This last analysis was performed with a leave one out cross validation and an increasing training set sample size. Starting at 5 training points, we train and predict the test point. We repeat this sampling 10 times to allow us to calculate our confidence intervals. Following the 10th iteration, the sample size is increased to 15, and the process repeats for the same test point. This is repeated, increasing the sample size by 10 until it can't sample anymore. Once this is finished, the model moves to the next test point of the leave one out cross validation and repeats. In this way, we create a bootstrapping example of how model performance changes at each sample size; where we see the model performance stabilizing and beginning to plateau, it is suggestive that additional data would not lead to substantial gains in model performance. For this analysis, we used the random forest model alone without the Rasch for exemplary purposes.

In terms of the sample size and its effect on our ability to predict a student's open response grade, it is clear from Figure 2.3 and the confidence that we see a statistically significant improvement in the performance from sampling 5 training

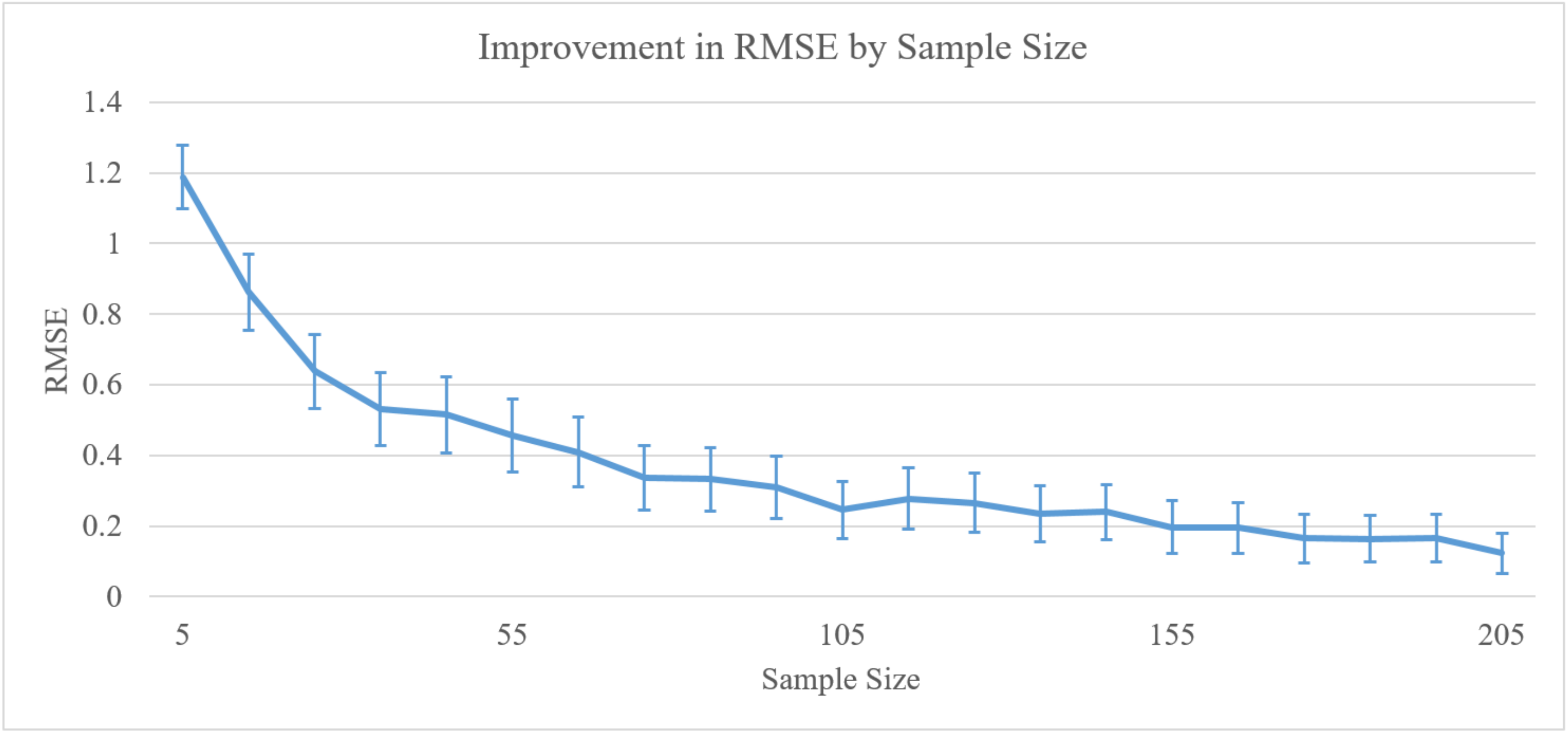


Figure 2.3: Selected for exemplary purposes, this plot illustrates the random forest model performance within a single problem over increasing sample sizes. Similar plots are generated for every problem to observe where our models may benefit from more data.

points to just under 55 training points. However, what is evident is that the model has maxed out its potential in its current form at just under 55 training points, and that additional data is not significantly improving the ability to predict the student's grade. With plots such as Figure 2.3 it suggests that any further improvement's would require updates to the model and data representation rather than simply collecting more data.

2.8 Discussion

Overall, the study aimed at utilizing modern machine and deep learning approaches to predict grades from authentic student open responses. With the ensembling of machine and deep learning with Rasch models, we have shown a strong ability to predict a students grade. Additionally, this study showed that in some cases more data wouldn't necessarily improve performance. Thus providing us the understanding that our ability to predict a grade, for a specific problem, may or may not improve with more data. However, given there are 2,042 unique problems, a limitation of this part of the study is that it's difficult to ascertain this information for each individual problem.

2.9 Future Work

With the overall strong model performance, there are a couple next steps we wish to address in future work. There is still a weakness in our model's ability to understand text. Currently, the best performing model, the Random Forest, is utilizing a bag of words approach, counting the words within student responses. There is no consideration of the structure of the student's response or what words relate to other words within the student's response. We attempt to combat this hindrance by utilizing the LSTM, but the lesser results of that model in Table 2.3 suggest that either the semantics and context of the words did not provide additional insight to the model, or, more likely, there is not enough data within each problem for such a model to effectively learn. The representation of data may also be an issue across these models, specifically in reference to the pre-trained GloVe embeddings. While a very powerful tool, as shown in previous research and discussed in this paper, models which utilize this are bound to the words in the pre-trained corpus. Even with the use of a Wikipedia trained GloVe embedding, our LSTM did not gain much in terms of additional information. Understandably, many functions and formulas students write aren't represented in the pre-trained embeddings. Currently, our team is developing an approach to expand these pre-trained embeddings to account for missing words, functions or math terms without requiring re-training of a GloVe embedding.

It is our goal to use these findings and the continued development of these grading models to deploy tools that can help teachers save time in assessing their student work so that they may direct their attention to the students who would most benefit from additional feedback.

Chapter 3

Is It Fair? Automated Open Response Grading

Erickson, J. A., Botelho, A. F., Peng, Z., Huang, R., Kasal, M., & Heffernan, N. T. (2021). Is It Fair? Automated Open Response Grading. In Proceedings of the Fourteenth International Conference on Educational Data Mining, June 29 - July 2, 2021. Paris, France. Poster.

Abstract

Online education technologies, such as intelligent tutoring systems, have garnered popularity for their automation. Whether it be automated support systems for teachers (grading, feedback, summary statistics, etc.) or support systems for students (hints, common wrong answer messages, scaffolding), these systems have built a well rounded support system for both students and teachers alike. The automation of these online educational technologies, such as intelligent tutoring systems, have often been limited to questions with well structured answers such as multiple choice or fill in the blank. Recently, these systems have begun adopting support for a more diverse set of question types. More specifically, open response questions. A common tool for developing automated open response tools, such as automated grading or automated feedback, are pre-trained word embeddings. Recent studies have shown that there is an underlying bias within the text these were trained on. This research aims to identify what level of unfairness may lie within machine learned algorithms which utilize pre-trained word embeddings. We attempt to identify if our ability to predict scores for open response questions vary for different groups of student answers. For instance, whether a student who uses fractions as opposed to decimals. By performing a simulated study, we are able to identify the potential unfairness within our machine learned models with pre-trained word embeddings.

3.1 Introduction

In recent years, natural language processing (NLP) has been at the forefront of machine learning in multiple fields. Whether it be within corporations or within the scientific community, NLP has provided deeper insights into consumer and user behaviors. Linguistics provides another source of information outside the standard data from user logs. Instead of relying on correlational assumptions from this data, inferences can be deduced directly from the users linguistics. While utilizing linguistics in education isn't genuine, modern machine learning and natural language processing has helped to automate the analysis and provide effective tools for learning. Especially within the online educational technology environment.

While online educational technologies has embraced linguistics, more specifically linguistics of teachers, students and chat systems; in recent years, the development of more advanced deep learning has brought a deeper semantic understanding of words within these linguistical models. More specifically, there has been a rise in sequential models which utilize word embeddings and vector spaces to develop algorithms which understand the semantic meaning of the words in sentences. To be able to infer more accurate predictions.

With the emergence of word embeddings and their vector spaces, many researchers have looked to utilize these approaches in their analysis in multiple fields. However, there is one shortcoming of word emnbeddings; to develop an accurate word embeddings which allows for accurate semantic understanding of words (based on their location within the embedding vector space), a researcher requires copious amounts of data. Without these large datasets, the vector spaces may provide very inaccurate semantic relationships of words. Thus, companies and universities sought out to utilize their own, or crawl the internet for their own, larger datasets to generate their own word embeddings. They would then publish them for public usage.

The emergence of word embeddings was an important development in machine learning and NLP, but the publishing of publicly available pre-trained word embeddings provided researchers with a powerful tool for optimizing algorithms with linguistics. While word embeddings were powerful for studies within areas such as MOOCS (i.e [KIK20][OT20]), smaller studies with less robust linguistic data were unable to utilize this modern approach for semantic relationship of words. Pre-trained word embeddings cut through that by providing researchers with more robustly trained word embeddings. Thus, researchers had a vector space which allowed for semantic relationships of words which their algorithm wouldn't have been able to generate on their own.

Its undeniable that having a word embedding trained on larger datasets, such as GloVe[PSM14] being trained on data from Wikipedia or Word2Vec[MCCD13] being trained on all of GoogleNews, provides deeper insights for the algorithm into the semantic meaning of words. Research has shown that the language which those embeddings were trained on provided underlying known biases [BCZ+16]. For instance, Word2Vec, as mentioned earlier, was trained on GoogleNews. The language utilized within the articles influenced the word embeddings to relate words in a bias way.

Since research has shown that some of the semantic meanings inferred from pre-trained word embeddings can elicit undesirable biases, the major question then becomes, does this underlying bias suggest the algorithm or predictive model will make unfair decisions? For instance, if an algorithm utilizes linguistics and NLP with pre-trained word embeddings will the predictions be unfairly made from those underlying biases (i.e. a scoring mechanism changing scores for certain groups of students). Our research attempts to explore:

1. Whether, through 3 simulated studies, the format a student writes an answer (i.e. fractions vs. decimals) effect the scoring model and potentially elicit unfair scoring?

2. What effect, through 3 simulated studies, if any, do *'distractor'* words have on the unfairness?

3. Whether or not underlying bias in pre-trained word embeddings can lead to unfairness in open response scoring models in middle school mathematics?

The simulated study and the analysis of the genuine middle school mathematics data utilize the recently published approach, termed ABROCA [GBB19], to evaluate what level of potential unfairness is present.

3.2 Background

3.2.1 Online Educational Technologies: Intelligent Tutoring Systems

In recent years, online educational technologies have been on the forefront of learning for students. While most learning with these systems have been supplemental to in-person learning, most recently, these systems have become more relied upon to deliver an effective education. A common online educational technology, intelligent tutoring systems (ITS) [CKA97], has been prevalent in education for many years. Some of the most common ITS are ASSISTments[HH14], McGraw Hill's ALEKS^{TM} and/or Carnegie Learning's Cognitive Tutor^{TM}. These systems aim to support both students and teachers through automated summaries of student performance, automated feedback and grading to students, hints, scaffolding, and common wrong answer messages. Through the use of both machine learning and software engineering, these systems have been shown to be effective at increasing the scores of students with end of the year standardized math exams[RFMM16] and the effects of their intelligent tutoring closely resembles the effect face to face tutoring has on students[Van11]. Other ITS, such as AutoTutor[GVR$^+$01],have attempted to resemble the face to face tutoring more directly by developing automated conversations and dialogues between students and ITS [GVR$^+$01]. However, most of the support and benefits of these ITS have been limited to questions with structured answers (i.e. multiple choice or fill in the blank questions).

3.2.2 Automation of Intelligent Tutoring Systems

While there are a plethora of ITS offering automated support for both students and teachers, this is mostly limited to questions with structured answers. Mainly, multiple choice or fill in the blank. It should be noted, that some of these systems, such as ASSISTments, support open response or short answer questions. However, the automation is limited to questions with structured answers. For a system such as ASSISTments, McGraw Hill's ALEKS^{TM} and/or Carnegie Learning's Cognitive Tutor^{TM}, it is straight forward to teach a system that A is the correct answers. Thus, if a student selects B, a system can easily score and suggest formulated feedback to that selection. The answers are finite.

Automated support of ITS is a draw for many teachers; one study noted that many utilize multiple choice questions for the efficiency and accuracy of grading [SK05]. However, since most of the automation is limited to questions with structured answers, the content which teachers provide is limited. To be able to expand the system's automation purview, natural language processing (NLP) has been brought to the forefront. Studies have looked to utilize NLP to automatically evaluate work or questions which require a student's unique linguistics (i.e. open response questions, or essays) including [SPR03][SB09][RHC+17][AB06][FLL99][TN16][LXZ19]. While most of this research has been primarily focused on content outside of mathematics, our previous research, [EBM+20], looked to help teachers diversify the content which they provide students in middle school mathematics by utilizing traditional and modern NLP to develop an automated scoring model for open response middle school mathematics questions. A more diverse set of question types can be beneficial to students and can elicit differing levels of cognition, as studies [Mar99][Ku09] note.

3.2.3 Natural Language Processing

Towards the goal of automating open response questions, or any linguistical/NLP prediction task, the major task is in how to numerically represent words thus a machine learned algorithm can generate an accurate prediction. One of the more simplistic approaches utilizes the frequencies of each unique word within the corpus, whats commonly known as a *Bag of Words* approach. While undoubtedly easy to interpret and not computationally intensive, this approach has been utilized in studies such as [Joa96] and is the foundation of more advanced approaches such as[SGA+15][GWHWH+00].

While frequency based approaches, like bag of words, are simplistic in nature and can provide insight, a major pitfall is that they begin to weight words more that occur more frequently. However, words occurring most often aren't always the most important or most informative. A common approach to combating this is to utilize term frequency inverse document frequency (TF-IDF). One study was able to use TF-IDF to accurately match words written in a query to the documents that are the most closely related[R+03].

Eventually, with the advancement of machine learning and deep learning, more modern approaches have gone to utilizing embedding vectors to represent words. Essentially, each word will have an attributed list of numbers which places that word in the embedding vector space. From this vector space, deep learning can utilize their locations within the vector space to understand the semantic relationship of words. As mentioned earlier, GloVe[PSM14] and Word2Vec[MCCD13] are two of the most common word embedding algorithms. However, for these approaches to be effective, there needs to be enough data present to generate the proper semantics of words. Without enough data, it is likely any semantics are incorrectly identified and the embedding space is ill defined. From this, it is difficult for an algorithm to utilize the generated embedding space to accurately comprehend what the text means or what it is inferring.

3.2.4 Pre-Trained Models

While word embeddings have been some of the most prevalent NLP techniques in recent years, there is a hindrance to this approach, data. To develop an accurate word embedding, with accurate informative semantics of words, there needs to be enough data with robust enough text. As mentioned earlier, if there isn't enough data, incorrect semantics of words can be inferred and the algorithm will incorrectly interpret text and linguistics. However, efforts have been made to combat this through pre-trained models. Instead of generating word embeddings from scratch, those with access to larger corpuses and datasets, such as Google and Stanford, trained their own Word2Vec and GloVe embeddings on GoogleNews and Wikipedia, respectively. This undoubtedly provides researchers with a very powerful asset to their NLP. Now, researchers can use pre-trained word embeddings with smaller corpuses and develop predictive models with embeddings generated from datasets that dwarf their own. This means a study which wouldn't be able to accurately utilize word embeddings in their predictive model, now can. As these pre-trained word embeddings have grown in popularity, word embeddings have expanded to utilize bidirectional encoder representations from transformers (referred to as BERT[DCLT18]) to create pre-trained word embeddings, as well.

With the success of word level embeddings, researchers looked to develop sentence level embeddings. Similar to the word embeddings, sentence level embeddings utilize deep learning to generate embedding vector spaces and embedding vectors which represent entire sentences. Two common approaches used are SBERT[RG19a] and the Universal Sentence Encoder[CYK+18] (often referred to as USE). Additionally, embeddings have expanded from the word and sentence level to document level embeddings. Approaches, such as Doc2Vec [LM14], are able to generate a single vector representation of entire documents. These more generalized embeddings allow for simpler direct comparisons of sentence and documents versus individual word embeddings. Similar to the word embeddings, these approaches are often pre-trained and released for public use.

3.2.5 Fairness

There are clear advantages to word embeddings and even more advantages to pre-trained word embeddings. This is also clear with sentence and document level embeddings as well. As discussed earlier, not everyone will have the resources to pull and analyze massive datasets to be able to accurately generate embeddings at the word, sentence or document level. With Google utilizing GoogleNews and Stanford utilizing Wikipedia, researchers have the opportunity to utilize semantics where they wouldn't have been able to previously. However, all of these pre-trained algorithms beg the question, what is being inferred from the data which is was trained on?

When it comes to linguistics, the way someone speaks, the way someone articulates, can be unique to themselves. Similarly, the way someone writes is personal to themselves and specific to their topic. So when algorithms are being pre-trained on data which isn't the researchers own data, there are questions to be asked. For instance, while there is more data, what are some of the semantic relationships these embeddings are identifying? From the word level, if the embeddings are developed from GoogleNews or Wikipedia, what is being identified? A recent study [BCZ$^+$16] looked to identify the semantic similarities.

Research[BCZ$^+$16], has been able to identify some potentially harmful semantic relationships present in common pre-trained word embeddings. For instance, [BCZ$^+$16] was able to identify that Google's pre-trained Word2Vec on GoogleNews elicited some harmful stereotypes. As the title of their research states, Google's pre-trained Word2Vec on GoogleNews closely associates ***Man*** with ***Computer Programmer*** and ***Woman*** with ***Homemaker***. Similarly this study looked to see what other potential gender stereotypes could be present within these pre-trained word embeddings. The authors managed to see that, for instance, occupations most closely related to the pronoun ***She*** were nurse, receptionist, socialite, housekeeper, nanny; and the occupations most closely related to the pronoun ***He*** were maestro, captain, skipper, boss and protege, just to name a few. There is clear evidence, that the language used within GoogleNews perpetuates certain stereotypes and undesirable biases.

While its clear that undesirable bias and harmful stereotypes are present in the pre-trained word embeddings, it doesn't guarantee that predictive models which utilize these are inherently biased. It may be the case that the algorithm could potentially be inferring dangerous semantic relationships, but the questions is will it effect the decision the algorithm makes. In education, this needs to be explored deeper. Automated scoring algorithms should be developed with the intention of scoring students without bias or harmful stereotypes being considered. That's why, for instance, in our past research [EBM$^+$20], all demographics were left out of the automated scoring model. It was our goal to score the student on their content; and content alone. There in lies the important question, while omitting variables which could cause unfairness in the automated scoring, are we continuing to avoid unfairness if we utilize pre-trained word embeddings.

Naturally, the next question becomes, how does one identify potential unfairness in their algorithms or predictive models? For instance, how can you identify if an open response answer automated scoring model is unfairly scoring? Recent work,[GBB19], developed an approach called Absolute Between-ROC Area (ABROCA). This approach utilizes the areas between two ROC curves to identify a model's ability to perform a classification task between two different groups of data. For instance, with open response answers, some students may write answers with mostly fractions and another group of students may use decimals and fractions. By generating the ROC curve of the prediction task for each group, you can utilize the area under the curves to identify the potential unfairness. So if there is a small area between the two ROC curves, one for the prediction task for each group, the less unfairness. However, if there is a large area, there is more unfairness in the prediction model. This research aims to develop a simulated study to examine whether the utilization of the pre-trained GloVe word embeddings within an automated open response scoring model can elicit unfair scoring, and whether or not there is evidence of unfairness with our previously developed open response scoring model in middle school mathematics by utilizing ABROCA as the evaluation metric.

3.3 Study 1: Simulated Study of Fairness in Automated Scoring

It is clear that embeddings have become an integral part of NLP and those hoping to develop predictive models utilizing linguistics are often drawn to the semantic properties which your model can utilize and learn. While researchers have noted that pre-trained embeddings are very powerful in providing an embedding vector space developed from robust datasets, and its clear there are undesirable biases built into those embedding vectors, its unclear as to whether or not those undesirable biases or stereotypes influence algorithms unfairly. Thus, this research developed a simulated study to attempt to identify if pre-trained word embeddings are utilized within an automated scoring model for open response answers, do they influence the model to make unfair predictions. As mentioned earlier, an example of this would be if a group of students states their answer with a fraction and surrounding text, does the predictive model generate scores similarly for those students that use decimals along with surrounding text? Through this simulated study, we are able to gain a deeper insight into what/if any unfair scoring occurs when utilizing the pre-trained GloVe word embeddings trained on Wikipedia.

There are 3 studies within this simulated study to help achieve this goal. First, we develop answers which contain differing distributions of answers which contain fractions and decimals and generate the ABROCA value at the differing distributions. Second, we attempt to see if decimals and fractions alone generate differing ABROCA values. Third, we attempt to see if additional 'distractor' words replace decimals in the text, do the ABROCA values differ at differing distributions? These studies will help provide deeper insight into the potential unfairness an automated scoring model can be producing when utilizing pre-trained word embeddings

3.3.1 Data Generation

At the foundation of this simulated study is the generation of the *student* dataset. The goal of this process was not to just generate 4 or 5 unique answers and randomly select 100 of those. This study set out to generate answers with more variability in their content and linguistics. To accomplish this, the generation was split into two facets, the training dataset student answers and the test set student answers. This was performed such that the model would not be able to have any identical answers between the training set and the test set. While this does create more noise, it helps to isolate what correlations our scoring model will eventually identify and predict from. Essentially, that the predictions aren't being made because the model has already seen that exact series of embeddings associated with a certain score.

Training Data: Corpus Generation

Table 3.1: Sample of Phrases and Their Associated Avg. Score

Generated Phrases	**Avg. Score**
my answer is	0.718750
i picked	0.622222
i guess the answer is	0.600000
i think it is	0.600000
i think the answer is	0.590909
i worked out	0.585366

Towards the goal of generating student answers with enough variability in their content, the generation of the corpus was founded on the goal of utilizing random selection. From this randomization, this study can mimic real open response student answers. This was based on the intuition of what makes open response answers unique is the variability within the answers. In our previous study, discussed further in chapter 4, we were able to infer that many answers were similar, but not fully identical. First, as Table 3.2 shows, there are 4 different length student answers in this corpus. There are answers which are 6, 5, 4 and 3 word length answers. The generation of the student answers can be surmised into 4 steps and visualized with Table 3.2:

1. Select whether it will be a student answer which uses decimals or fractions
2. Randomly select what length the answer is.
3. Once a length is randomly selected, another random selection is made between the two structures (i.e. 'Answer Structure' in Table 3.2)

4. Randomly select text from **Fill "1"** and **Fill "2" Fractions** or **Fill "2" Decimal** to fill the identifiers **'1'** and **'2'**

To summarize, the first step of the generation of the student answers is to decide whether the answer will contain a decimal or a fraction. This is followed by randomly selecting what length the answer is. Once a selection is made, there are two potential answer structures to choose from. In Table 3.2, this is the column 'Answer Structure'. For all length answers, there are two types of answers with different structures. Another random selection is made between the two structures. From there, *'1'* and *'2'* are filled with random selections made from the available text (i.e. *Fill "1"* and *Fill "2" Fractions* or *Fill "2" Decimal* in Table 3.2). Another way to describe the text and language used in *Fill "1"* are 'distractor' words.

Test Data: Corpus Generation

With a corpus generated to simulate training data, the next step includes generating a testing corpus of student answers to select from. The steps are the same as the generation of training dataset steps listed above. However, Table 3.3 shows there is one key distinction between the training and test generation, the text which can be filled (*Fill "1"*). More specifically, the answers which are generated for the test set will never occur in the training set. As mentioned earlier, this was performed for two reasons. One, this allowed for a more realistic distribution of student open response answers. Often times, answers are similar, but few are identical. This is what makes automatically scoring open responses questions difficult. There are a infinite set of answers. Therefore, this variability helps to simulate data genuine student answers. Secondly, by having different text and phrases to select from that are different than the training set corpus, guarantees that our automated scoring model will not be identifying sequences of words, or phrases, that are identical in the training and test set. This allows us to understand, more specifically, what our algorithm is making decision on and what correlations its finding. If it see's the exact same answer it was trained on and predicts a score, that doesn't provide insight to whether the word embeddings are impacting the fairness of the algorithm, rather that it has identified an identical answer. Without this step, it would be difficult to identify if any changes in predictability between two groups are from one group having the identical answers and scores, the 'distractor' words, or the math terms.

It should also be noted that the *Answer Structure* for the test corpus is also different than the training corpus. In the training dataset, words were being selected and placed in the answers for the 'distractor' words. Whereas in the test dataset, whole phrases are being selected for the 'distractor' words. Again, this allows for variability in the answers between the training and test datasets.

In the end, a separate corpus of student answers was generated which contain decimals and fractions, separately. Therefore, an individual corpus of generated student answers using fraction for both training and test and an individual corpus of generated student answers using decimals for training and test datasets were generated. These corpuses are what will be used to select the final training and test data.

As for the scoring of the simulated student answers, a general rule was set that any answer that contains 3/4 or 0.75 is considered correct. All other answer are considered wrong. Partial credit is not considered in this simulation study. Thus this is a binary classification task.

3.3.2 Methodology

Once the corpuses have been generated, the process of selecting data can begin. This can be surmised by the overall goals of this study. This study sets out to identify if a automated scoring model for open response questions, which utilizes pre-trained word embeddings, elicit unfair scoring. To accomplish this analysis, there needs to be an identifiable difference, outside of the 'distractor' words, between student answers. In this case, each student open response answer has 'distractor' words and either a decimal or a fraction (as discussed in the previous section). Inversely, our goal of this simulation study is to also extrapolate whether the 'distractor' words, not the fractions or decimals, influences any unfair tendencies in the scoring of the simulated student open response answers.

For the sampling (as shown in Figure 3.1) of the simulated student open response answers, we set out to simulate data which consists of a balance of student answers which utilize fractions and decimals. The training set is made up of simulated student answers from both the answers which contain fractions and contain decimals separately. The steps to the selection process is as follows, at an instance a simulated student answer is to be selected:

1. a student answer is always drawn from the training dataset of simulated answers which contain a fraction. This is considered *Group A* students.
2. a random integer is drawn
3. if the integer is below our specified threshold ($\boldsymbol{T}$ in Figure 3.1), another selection is made from the training dataset of simulated answers which contain a decimal. This is then considered an answer from *Group B* students.
4. if the integer is above our specified threshold, another selection is made from the training dataset of simulated answers which contain a fraction. This is also considered an answer from *Group B* students.

A threshold was set for selecting decimals and fractions to control the balance of answers. This lends itself to our goal of being able to identify whether or not the format a student writes an answer, i.e. using factions vs. decimals, effects our ability to score student open response answers. By having a threshold, we can increase the threshold incrementally and see what is the model's ability to score the simulated student answers. So as the threshold increases, more and more answers that contain decimals (Group B students have more and more answers containing decimals) are selected and trained upon. Thus, with ABROCA, fairness can be identified.

For the test set, a similar approach is taken. Since the training set contains answers of both Group A students, which are students who all answered with a fraction in their text, and Group B students, which some student used fractions and some used decimals, the test set will contain the same Group A distribution of answers containing fractions only, but with different content making up those answers, and the same Group B distribution of answers containing decimals.

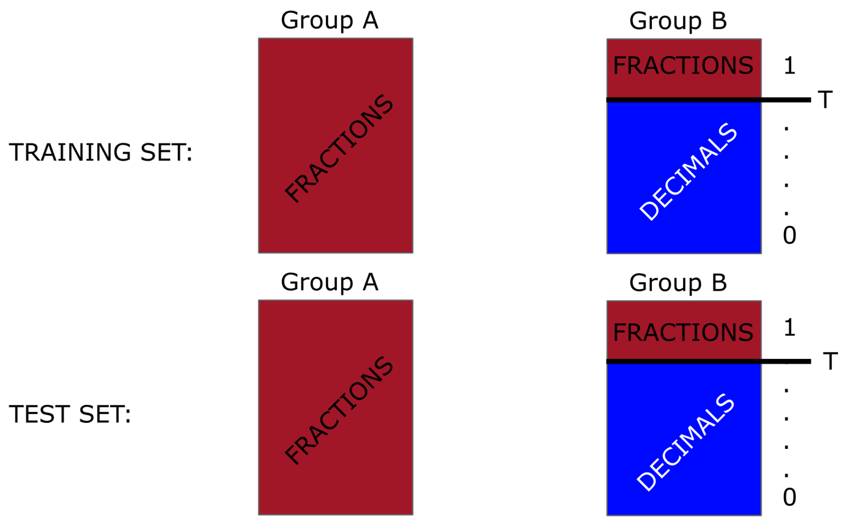


Figure 3.1: Train Test Data Sampling Process with Threshold

While it was emphasized that the training and test sets have similar distributions of answers containing decimals and fractions, the two datasets have identical distributions of grades. This was done to remove outside influence on the automated scoring model. If there is an unbalanced grade distribution, then the performance of the model could be driven by more scores of 0.0 or 1.0. By balancing the grades across both the training and test datasets, this uncertainty is removed.

If an automated scoring model is fair, as the distribution of student answers which fractions and decimals changes within the training and test dataset (as mentioned earlier, the distribution is the same for both training and test), the model's ability to score them should not change. Again, this is utilizing ABROCA. In simplest terms, the absolute difference between the area under the ROC curves should be minimal between two groups in a prediction task to be considered fair. This shouldn't change given a distribution or more answers which contain decimals or fractions.

To improve the reliability of the results, we re-sample/re-select the test dataset 10 times and evaluate the model's ability to score an open response answer. This form of cross validation allows us to see if the ability to predict the score was only for that unique set of words, or was the performance consistent across multiple iterations.

To summarize, the training dataset is a selection of both answers which contain fractions and decimals (Group A student answers and Group B student answers), using the specified sampling/selection method mentioned above. The test dataset contains the same distribution of data, Group A students, who always use fractions, and Group B students, students who use both decimals and fractions. Again, to reiterate, the balance of Group B in the training and test set are the same. Thus, this can narrow down, if there is a large ABROCA value at different thresholds (more and less decimals/fractions), there is evidence that the fractions and decimals are not impacting the algorithms ability to score the open response answers. If there is a large ABROCA value, there is evidence that there is unfairness in the model's predictions.

As mentioned earlier, the threshold was set to decided whether or not an answer which contains a decimal or fraction is sampled. To reiterate, this is performed by randomly selecting a value between 0 and 1, and if the value falls below the threshold, an answer which contains a decimal is sampled, otherwise, an answer with a fraction in the answer is sampled. We take an incremental approach to the threshold, starting off with a threshold of 0.0, and increasing by 0.10 until a threshold of 1.0 is reached. Again, this allows us to see if there is evidence of unfairness, in terms of ABROCA, at each of the levels. Additionally, if there is evidence, is it occurring with more decimals or fractions?

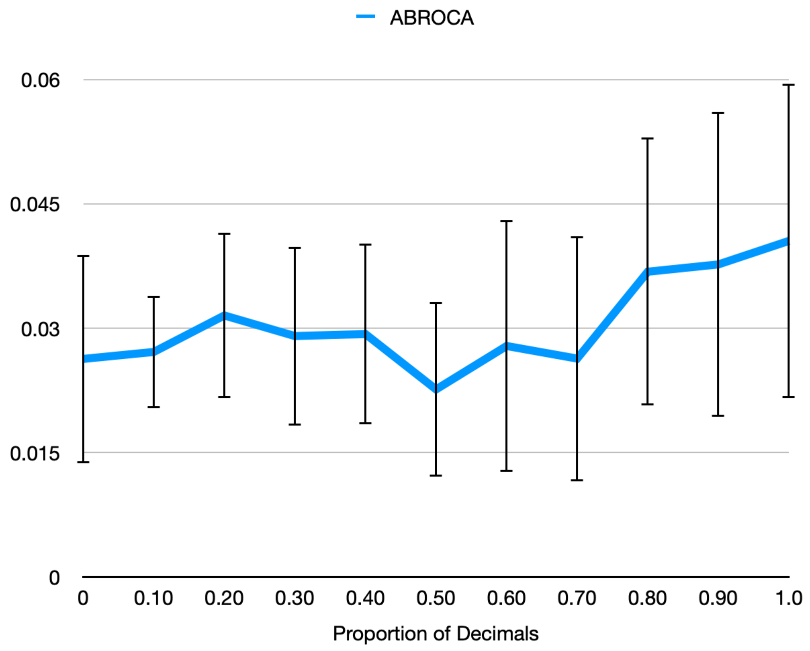


Figure 3.2: Study 1: ABROCA Values at Incremental Fraction/Decimal Thresholds

All of the studies will incorporate a Long Short Term Memory (LSTM) [HS97] model which utilizes the pre-trained word embeddings to automatically score open response answers. An LSTM model is appropriate here for its sequential attributes. We are able to feed in sequences of words which the LSTM can reference the pre-trained word embeddings to garner semantic meanings of words and the order which they are used. To identify whether or not unfairness is present in an automated open responses scoring model utilizing pre-train word embeddings, we constructed 3 simulated studies with the artificially generated student answers. First, a study which utilizes the similarly balanced simulated training and test datasets, and predicts Group A scores (the group of students who all used fractions within their answers) and Group B (a split of students utilizing fractions and decimals in their answers). Then, we incrementally increase the threshold controlling the split in Group B data (which is similarly controlling Group B threshold in the training set), and utilize ABROCA to directly compare our LSTM's ability to score the student's answers in Group A and Group B separately. We increase the threshold, and repeat. This continues while the threshold increases by 0.1 until a threshold of 1.0 is met. With this, we can gain insights into whether or not the automated scoring of answers of each group is unfair and if so, evidence could be that the use decimals or fractions could be to blame.

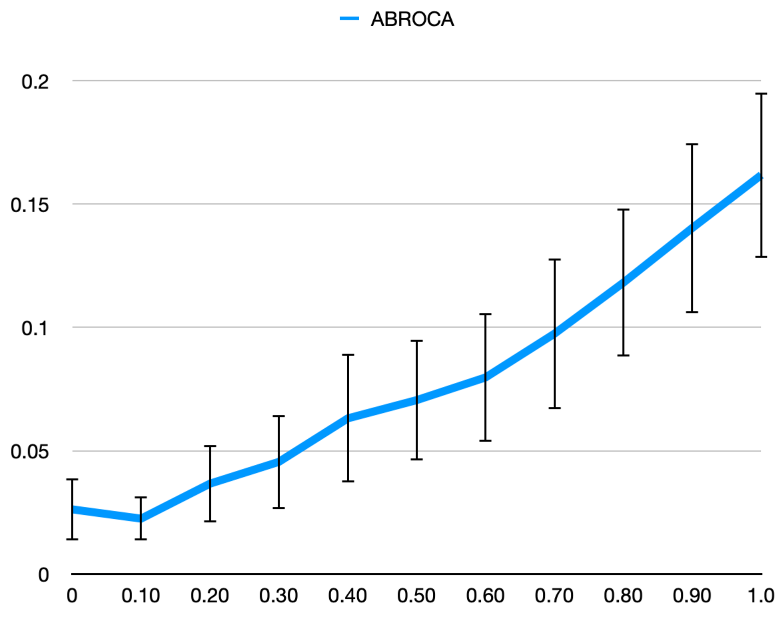


Figure 3.3: Study 3: ABROCA Values at Incremental Fraction/Decimal Thresholds

The second study looks to identify, if there is variation in our ability to predict scores for Group A and B, whether or not fractions or decimals are the culprit of the potential unfairness in the automated scoring model. In the simplest approach, we remove all non-fraction and non-decimal text from the student answers. Thus, leaving just a fraction or a decimal in the testing dataset. We then develop predictions in the same fashion in the first simulated study mentioned above. This allows us to identify whether or not the pre-trained word embeddings associated with the training data causes the LSTM show unfairness to those using decimals vs. fractions in their answers. Ideally, the ROC curves should be similar. If not, this would suggest that the surrounding 'distractor' words could be influencing the unfairness.

This then leads into the final simulated study. While holding the training and testing of the LSTM constant with the previous two simulated study, this final study replaces the decimals with 'gibberish', or words which are not recognized by GloVe as a pre-trained word embedding. These were chosen by randomly selecting a string of characters. This would increase the amount of 'distractor' words in the text. From this, we can identify whether or not there is unfairness in the LSTM in predicting Group A and Group B scores. If there is unfairness, large ABROCA values, this would suggest that the 'distractor' words are influencing the unfairness. Mainly, the 'gibberish' added does not provide additional information to the LSTM because there aren't pre-trained embeddings associated with those random strings of characters. Thus, a list of 0's is passed through the LSTM and no inferences can be made from those words. Also, since we are only replacing decimals, as the threshold increases, fewer fractions will be available for the LSTM to learn from. So as the fractions drop, so should the ABROCA score. If the ABROCA score increases, there's evidence supporting that unfairness is present from the differing coverage of answer-related tokens within applied methods utilizing pre-trained NLP embedding methods.

From all 3 of these simulated studies, a picture can be painted if the bias present in pre-trained word embeddings causes automated open response scoring models, such as our LSTM, to unfairly grade different groups of students. Similarly, if there is unfairness present, the combination of these 3 studies will allow us to identify what may be causing or influencing the unfairness within our model. Is it the model, is it the embeddings, is the word usage, is the use of decimals vs. fractions? These are the questions which these simulated studies can help to answer.

3.3.3 Results

First, the results from the standard prediction task of taking the artificially generated student open response answers, in their original form and utilizing pre-trained word embeddings, and utilizing a LSTM to predict what score a student will receive. From our simulated study, Figure 3.2 presents the ABROCA values at each incremental threshold. Reminder, as the threshold increases, more student answers contain decimals instead of fractions (and vice versa). What is apparent in Figure 3.2 is that the ABROCA values ever so slightly increase with the more answers which include decimals. However, the amount is almost negligible. Producing ABROCA values near 0.02 and just over 0.04. This is minimal, that means that the absolute difference between the area under the ROC curves is around 0.04. The model appears to be able to accurately predict the score a student will receive quite similarly for both groups when decimals, fractions and ‘distractor’ words are within the answers.

For the second study, which attempts to identify if any unfairness is present when training on answers which contain fractions, decimals and ‘distractor’ words, but the test dataset only consists of answers with just a fraction or a decimal. This could help to again identify whether or not our ability to score Group A or Group B of students, which use differing levels of fractions and decimals, changes with differing levels of decimals and fractions in the training and testing data. In the end, the ABROCA score was consistently 0 across all thresholds. This meant that no mater the distribution of fractions, decimals and 'distractor' words in the training set, and the distribution of fractions and decimals in the testing dataset, the LSTM with pre-trained word embeddings predicts the score equally for both groups. The LSTM appears to pick up on the rules that answers with 3/4 or 0.75 are correct. Whether decimals or fractions are used doesn't change the LSTM's ability to score different groups with different distributions of fractions and decimals being used.

In the final simulation study, we attempt to identify if the 'distractor' words elicit unfairness in the LSTM utilizing pre-trained word embeddings. By removing all decimals and replacing them with strings of characters that are unidentifiable by GloVe's pre-trained embeddings, we can isolate the model to generating predictions based solely on the surrounding 'distractor' words. Figure 3.3 shows that the ABROCA score does indeed increase with more unrecognizable words within GloVe's pre-trained word embeddings. When the threshold is set to 0, all the answers contain fractions, and again, we can clearly see that the ABROCA score is quite low, near 0. This is because of the fractions being recognizable to the LSTM. So when attempting to predict scores for Group A and B (B in this case is a split of random characters in place of decimals and fractions), there isn't unfairness present. However, as the fraction wane and disappear, the ABROCA score increases and continues to increase close to 0.18. This may be a bit of a surprise because as the threshold increases, and the number of answers with fractions drops, the LSTM should be able to only identify the similar amounts of words which were randomly selected. In this case, this didn't happen, because Table 3.1 shows some of the phrases used in the generated student answers were commonly associated with more correct answers. So the LSTM was able to pick up on some of these trends and identify those correlations.

In the end, these simulated studies proved the largest risk for unfairness exists when there is differential coverage of answer-related tokens within applied methods utilizing pre-trained NLP embedding methods. So when answers consist of equally recognizable words within GloVe's pre-trained word embeddings, there's unlikely to be unfairness in the grading. There wasn't evidence that the inherent bias built into the pre-trained word embeddings elicited more unfair scoring of student answers in, in terms of this simulated study. But if there are unbalanced recognizable words and tokens in the student answers, attention needs to be paid to potential unfairness in the automated scoring.

3.4 Study 2: Middle School Mathematics Automated Scoring Fairness

While a simulation study is powerful on its own, it is difficult to recreate authentic student data. For the final overall study of this research, we look to once again utilize ABROCA to identify if our own algorithm, trained on genuine student open response answers within ASSISTments, is unfair in its grading of women and men.

3.4.1 Data

The data consists of two separate datasets consisting of open response questions with associated teacher scores. In its raw state, the dataset consisted of 150,477 total student answers. Within these student answers, 27,199 unique students provided the answers and 970 teachers graded them. These grades and answers span across

2,076 unique problems. It should be noted, that this is the same dataset we used in our study [BBE+21]. All of this data comes from middle school mathematics.

However, in its raw state, this data needed filtering down. We make sure to remove any student answers that are empty strings or contained only an image. These filtration steps condensed the dataset down to a total of 141,612 graded student open response answers. In the end, there were a total of 25,069 unique students who answered and 891 teachers graded those answers. After the filtering, there were still 2,042 unique problems attempted. Lastly, the scoring. This was performed on a 5 point scale, where students receiving a 4 is a perfect score.

It should be noted, to be able to perform the fairness analysis using ABROCA, gender was inferred. This performed by cross checking names with the census data. If the name was found only on the women or only on the men's list, it was labeled as such. If any names fell into multiple genders, it was labeled as unknown and excluded from this analysis.

3.4.2 Methodology and Results

Towards developing our predictions, we utilized another pre-trained algorithm, mentioned earlier, called SBERT. This is a pre-trained sentence embedding algorithm which allowed us to generate a single vector representation of each student answer. We then utilize a Canberra distance to identify which student answers are the most similar. Whichever was the most similar, that was the score we would assign. This approach managed to out do our previous models [EBM+20][BBE+21].

While utilizing, once again, ABROCA to identify potential unfairness, we apply this to our algorithm. We were able to show that our SBERT model with Canberra distance manages to fairly score both Male and Female student open response answers. Our model managed an ABROCA of 0.007, which is quite small. Suggesting that our algorithm is indeed scoring Men and Women fairly.

3.5 Limitations and Future Work

While there were indications of unfairness in cases where there were unbalanced identifiable tokens within the student open response answers, this analysis is strictly middle school mathematics. This type of analysis would need to be applied to additional datasets to get a broader understanding of the potential unfairness in other subjects and age ranges. In terms of our analysis of our SBERT model for scoring student open response answers, while there wasn't unfairness identified, more work needs to be done to explore the embeddings themselves. Pre-trained word embeddings have been shown to have bias built in, but what bias is present in the pre-trained sentence embeddings? This is a question we look to explore further.

3.6 Conclusion

Overall, this study set out to run a simulated study to help identify potential unfairness within models utilizing pre-trained word embeddings. While there is bias present in the embeddings themselves, our simulated study didn't show this bias causing unfair scoring. However, our analysis did show that when developing models with pre-trained embeddings, unfairness can begin to occur when there is an imbalance of recognized tokens in the student answers. More specifically, our simulated study showed that when groups within the data use differing levels of recognized tokens, it increases the chance for unfair scoring.

While our simulated study showed how unfairness can present itself within a scoring model, our model did not show this unfairness. We were able to conduct an analysis of our model with ABROCA to compare our performance scoring Men and

Female. In the end, the ABROCA values was nearly 0 at 0.007.

In the end, we were able to utilize a simulated study to help identify potential unfairness in automated scoring models which utilize pre-trained word embeddings. Its been widely noted that those embeddings have bias built in, but our simulated study couldn't show unfairness in the scoring of differing groups of simulated student answers. However, this study did show that when student answers have differing levels of tokens recognized, automated scoring models which utilize pre-trained word embeddings can start to unfairly score.

Table 3.2: Training Set Corpus Generation

Answer Length	**Answer Structure**	**Answer Content**	**Fill “1”**	**Fill “2” - Fractions**	**Fill “2” - Decimals**
6	6 - A	i **1** the answer is **2**	‘think’, ‘believe’, ‘feel’, ‘suppose’, ‘guess’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
6	6 - B	**1** the answer **2**	‘i arrived at’,‘ i worked out’, ‘ended up with’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
5	5 - A	i **1** it is **2**	‘think’, ‘believe’, ‘feel’, ‘suppose’, ‘guess’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
5	5 - B	i **1** the answer **2**	‘chose’, ‘thought’, ‘picked	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
4	4 - A	**1 2**	‘i arrived at’,‘ i worked out’, ‘ended up with’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
4	4 - B	my **1 2**	‘guess was’, ‘answer was’, ‘belief was’, ‘answer is’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
3	3 - A	i **1 2**	‘chose’, ‘thought’, ‘picked	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
3	3 - B	it **1 2**	‘was’, ‘is’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60

Table 3.3: Test Set Corpus Generation

Answer Length	**Answer Structure**	**Answer Content**	**Fill “1”**	**Fill “2” - Fractions**	Fill “2” - Decimals
6	6 - A	i 1 2	‘arrived at the answer’, ‘thought the answer was’ ‘calculated the answer was’, ‘guessed the answer was’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
6	6 - B	2 1	‘is the answer i thought’, ‘was the correct choice here’, ‘is what i guessed right’, ‘was what I arrived at’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
5	5 - A	1 2	‘im guessing it was’, ‘my work arrived at’, ‘the answer is clearly’, ‘clearly the answer is’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
5	5 - B	2 1	‘was what i guessed’, ‘is what i calculated’ ‘was the clear answer’, ‘is the correct answer’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
4	4 - A	1 2	‘my guess is’, ‘my answer is’, ‘my work showed’, ‘my thought is’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
4	4 - B	2 1	‘is my choice’, ‘from my work’, ‘is my answer’, ‘is the answer’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
3	3 - A	2 1	‘is right’, ‘is correct’, ‘i found’, ‘i thought’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60
3	3 - B	1 2	‘answer is’, ‘choice was’, ‘i guessed’, ‘i thought’	3/4, 1/2, 1/3, 2/5, 3/5	0.75, 0.5, 0.33, 0.40, 0.60

Chapter 4

Are These Similar? The Development and Evaluation of a Feedback Recommendation System

Erickson, J., Botelho, A. F., Alphonsus, A.G., Benachamardi, P., Swinger, N., & Heffernan, N. T. (2020). Are These Similar? The development and Evaluation of a Feedback Recommendation System. Manuscript prepared.

Abstract

The practice of identifying and quantifying the similarity between two or more artifacts is often at the foundation of a recommendation system. In such systems it is often important to be able to effectively compare a given scenario or artifact with a pool of known data in order to make inferences into the most appropriate course of action. This comparison, between what is currently observed and what has been observed in past examples, is also often very difficult when considering abstract or ill-structured data such as natural language. While recommendation systems are by no means novel concepts within the context of educational technology, such systems have been developed and applied in areas of course selection in higher education or even to help teachers deliver content to their students. Similarly, educational technologies have devoted large amounts of research and development to leverage statistical methods and machine learning to help teachers score and assess student work. However, in both of these areas, online learning systems often lack in supporting the recommendation of meaningful feedback to student work, particularly when that work is in the form of natural language. In this paper, we explore this intersection of natural language processing, machine learning, and education with the goal of developing a system to help teachers provide feedback to their students' work. Toward this goal, we explore several methods used to identify the similarity of natural language artifacts and develop a metric which we use to evaluate these methods as recommendation policies in an offline manner.

4.1 Introduction

Technologies designed to augment humans' decision-making processes often rely heavily on their ability to effectively compare new experiences with historic data. If a particular observed scenario has been seen in the past, it is likely that a course of action that was previously successful in such a case may be appropriate in the present as well. In the context of education, for example, if a student underperformed in mathematics classes in high school, it may be appropriate to recommend that student enrolls in a remedial math course in college based on similar students benefiting from such a selection in the past. In such a practice, however, the success of the recommendation is based on the system's ability to compare and quantify similarities between two artifacts (i.e. the system needs to be able to find historic examples that are similar to what is currently being observed). In the course recommendation example, the system needs to be able to quantify and compare student performance in high school mathematics courses using, for example, letter grades or students' grade point averages.

While the manner in which a system may ultimately compare artifacts varies greatly by domain and context, it is often much easier to compare well-defined artifacts than it is to measure similarities and differences between more abstract or loosely-structured cases. If a student has an average grade of C in their high school mathematics courses, it is easy to compare this student to others who also received an average grade of C in their high school coursework. It is this type of comparison that existing learning systems are also able to leverage when providing feedback to students.

One of the largest benefits provided by computer-based learning systems in classroom settings is the ability to provide immediate feedback to students. In the domain of mathematics, even simple correctness feedback can be beneficial [KKH13]. Beyond correctness feedback, these systems may be able to provide further aid through scaffolded problems [RV06], worked examples [RAMK11], or even answer-specific feedback to help remedy known common errors [SH14]. In all of these examples, however, support from such learning systems is generally confined to close-ended problem types with structured answers; these types of problems include those such as multiple choice and fill-in questions where there are a finite and often small number of acceptable correct responses. For instance, it is easy for a learning system to understand that the accepted value for x in the equation $x + 4 = -8$ is *-12.* If a student were to answer with the value of *-4*, for example, the system may be able to provide correctness feedback along with a message that highlights the student's mistake.

While there is notable support by these systems for closed-ended problems, there is often lacking support for open-ended problems that require students to use natural language to describe their work or otherwise answer the question. In such cases, teachers must often manually score and provide feedback to students (a task that is undoubtedly tedious and time-consuming). While prior research has been conducted toward automating the scoring of student open-ended answers, this work is focused on developing a method of providing suggested feedback that a teacher may give in response to their students' work.

Considering the growing diversity and advancement of natural language processing (NLP) methods used to quantify syntactic and semantic attributes of written language, this work seeks to explore these methods within the context of education. Toward the ultimate goal of developing a system that helps teachers provide feedback to students, this paper presents three contributions

1. We propose a method of providing teachers with automated feedback messages for their students' work based on measuring the similarity of student responses.

2. We develop a data-driven metric and procedure to evaluate automated feedback recommendation policies in an offline manner.

3. Leveraging traditional and state-of-the-art NLP and machine learning methods, we conduct an empirical analysis to compare these methods as feedback recommendation policies.

4.2 Background

4.2.1 Intelligent Tutoring Systems

Intelligent tutoring systems (ITS) and other computer-based learning systems have existed for several decades [CKA97]. As is apparent today with systems such as ASSISTments [HH14], McGraw Hill's ALEKS$^{\text{TM}}$, and others becoming more ingrained in education systems, the role of feedback, both to students and to teachers, has been highlighted for its importance. Recent studies such as [RFMM16] found that ASSISTments (the learning system from which the data of this current research was derived) significantly increased scores of the students on end of the year standardized mathematics exams as compared to traditional homework students. When students worked on homework, those who were provided immediate feedback performed significantly better than those who had to wait for traditional grading and feedback. Similarly, as the study from [Van11] highlighted the often positive effects of intelligent tutoring systems on student learning. This study found that intelligent tutoring systems have the opportunity to be almost as effective as human tutoring. Additionally, the authors noted that while ITS have not replaced human tutoring, improvements to the ITS, as compared to a human tutors, potentially could be attained more easily. In the end, all of these systems look to provide immediate feedback to students given close-ended questions while providing timely reports to teachers of their student's performance.

4.2.2 Close-Ended vs Open-Ended Questions

As discussed earlier, these systems mostly focus on close-ended problem types with structured answers (although some exceptions are observed). While not restricted to such problem types, content commonly consists of multiple choice, select all that apply, and fill-in types of questions. Again, these question types are easy to support as they are fast to automatically grade and notable research has been conducted to improve feedback to students. However, [Ku09] discussed that providing only one question type, such as multiple choice, would be inadequate in capturing the students rationale or process of thinking. Similarly, when comparing close-ended questions (such as multiple choice) and open-ended questions, [Mar99] describes the different levels of cognition required for each question type. Mainly, the authors discussed how open-ended questions require a wider spectrum of cognition than that of a close-ended question. Another study, [BS06], discussed the importance of having students generate their own responses and creating their own connections between economic theories. However, while the paper acknowledges the importance of creating these connections on their own; the authors concede to the affordability of grading close-ended questions versus open-ended questions.

While ITS systems offer some support for open-ended questions, there is rarely any automation provided. This is the appeal of close-ended questions, similar to [BS06], as its more economical to provide close-ended questions to students because of the built-in automation. It is true that there is growing attention to this problem and efforts are being made to automate the assessment of open-ended answers (c.f. [AB06],[FLL99],[ZZX$^+$17]), there is still little support for teachers to provide feedback to such responses. With this study, we present an approach to enable teachers the opportunity to diversify their questions to students while providing the automation they have come to enjoy from close-ended questions.

4.2.3 Word Representations

One of the most common first steps in natural language processing (NLP) is to decide how to numerically represent the content. There have been many methods proposed in the past as to how to best represent text in a manner that captures syntactic as well as semantic meaning. The simplest way to represent language is perhaps with a bag of words approach. By adding up the number of times the word occurs, that can be the number which represents said word. While this has been apart of, and the foundation of, recent studies, [SGA+15] [GWHWH+00], the bag of words approach tend to act as a baseline of comparison for more complex methods; this is often as it is a strong foundational step, however, rather than as a strawman comparison. Similarly, some research utilizes n-gram approaches [CT+94] for text classification. Where instead of having a list of individual words, a list of grouped n words could be counted. Helping to provide some sense of context in the words. Not only is this being used in interpreting speech or text, its been utilized in detecting malicious code [AACKS04]. While counting the words provides insight into the most commonly used words, there was the concern that these words weren't the most relevant. Term frequency inverse document frequency, TF-IDF, was developed to lower the weight less relevant words [R+03]. A recent study showed success in automatically grading student open-ended questions using TF-IDF [EBM+20].

What is clearly missing from the bag of words based approaches is any relational or contextual understanding of the words. Bag of words gives provides the understanding of which words were used the most or most in relation to a document, but it fails to provide and relational information. With the recent advancements in deep learning, word embeddings have gained momentum. Two methods are the leading choice in word embeddings, Word2Vec [MCCD13] and GloVe [PSM14]. Each of these provide an opportunity to utilize deep learning to train models to create vector word representations. Once trained, sentences can be fed into the model and each word will retain a vector representation. Thus allowing models to gain relational understanding of words. For instance, given two different words accompanied by two different vectors, where these words reside within the vector space indicate how similar they are. If the two vectors are far apart within the vector space, they are less similar. An additional benefit to word embeddings is the availability of pre-trained word embeddings. It is the case that many NLP studies do not have overly robust corpus. This provides a large road block for many studies. However, publicly available pre-trained word vectors allow smaller corpus' to gain a more accurate dataset of vectors in comparison to training and generating the embedding from itself.

In recent years, word embeddings have evolved into sentence and document embeddings. Instead of developing a vector representation of each individual word, approaches such as Doc2Vec [LM14], aim to generate a single embedding to represent an entire document or sentence. Likewise, other approaches such as the Universal Sentence Encoder [CYK$^+$18] and SBERT [RG19a] have gained popularity in their ability to represent sentences as a single vector. Thus, the vectors provide insight into how similar two sentences, phrases, or paragraphs are.

4.2.4 NLP for Recommendation Systems

Perhaps one of the most widely recognized examples of natural language processing and machine learning used in a recommendation system is that of Google's SmartReply tool [KKR$^+$16]. As this and similar technology has become common in the context of email, many are familiar with this type of tool being used to help users respond to email. In this way, the tool utilizes natural language processing along with several other machine learning methods such as Long Short Term Memory (LSTM) deep learning networks [HS97] to "read" email and recommend appropriate responses.

While the original SmartReply paper describes a generative approach (where responses are being generated word-for-word), many similar recommendation systems instead rely on a case-selection method (i.e. from a pool of known possible artifacts, select the one that best applies). In either case, however, this technology, like other recommendation systems, relies on the method's ability to identify similar known examples in order to make informed recommendations as to how to proceed.

This technology forms the inspiration for the feedback recommendation system described in this paper. It is our ultimate goal to develop a system that is able to suggest teacher feedback in a similar manner as SmartReply is able to suggest email responses. This paper represents a step toward this goal.

4.3 Defining Similarity

As the concept of "similarity" is an prominent aspect of this current work, it is important to discuss our definition of this term, or more precisely, acknowledge that there are multiple definitions of this term.

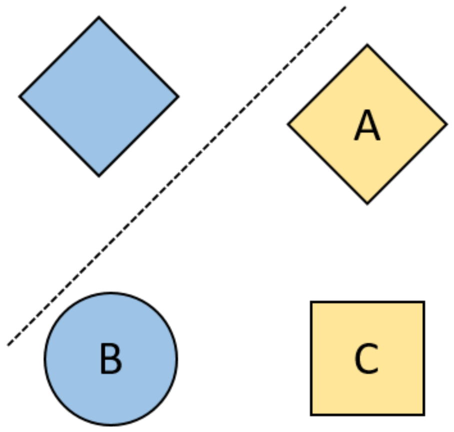


Figure 4.1: Example of how similarity can be defined along multiple dimensions of comparison.

Consider the example illustrated by Figure 4.1. Which of the three objects, A, B, or C, is most similar to the target object in the upper left? Is it possible to order these artifacts from most similar to least similar? Ignoring context, it is not likely that readers would unanimously agree on the answers to these questions due to the number of dimensions in which the artifacts can be compared. Similarity here can be expressed in regard to shape, rotation, color, or any number of other attributes, and each artifact exhibits similarities and differences along each of these dimensions; without more information (or more structure to the problem) it is impossible to know which dimensions should be given higher importance. The difficulty of this task stems from the combination of the dimensionality of the artifacts and the unknown weights of these dimensions for comparison.

In a practical sense, this challenge is even greater when even the artifacts are difficult to describe, such as natural language. Consider, for example, the sentences "see Spot run" and "Spot runs fast" for comparison. In what ways are these sentences similar? Semantically, they both refer to "Spot" and describe Spot's action of running, but there are many other ways to compare these. Both of these are similar in their count of the letter "s," both use the same number of spaces and have the same number of words. Likewise, apparent similarities could be viewed as differences; the word "Spot" appears earlier in the second sentence. In this way, there are multiple "correct" ways of measuring the similarity of these sentences; it is just likely the case, however, that some methods are more *useful* than others depending on the context.

In the most abstract sense, similarity can be defined as the distance between quantified, or more specifically, vectorized artifact representations. Once an artifact can be quantified along its various attribute dimensions, these values then represent the artifact's point within a representation space. Measuring the distance between these points reveals the similarity of the given points. Using Euclidean distance, for example, is one particular method that measures the similarity of two points based on how far they are from each other in a geometric sense. Alternatively, cosine similarity is a common method that essentially measures the distance between the angles of two vector representations.

To clarify, there are countless ways of building numeric representations of text, just as there are many common methods that can be used to determine the similarity or "distance" between representations. The problem is not in our ability to quantify attributes that describe these artifacts or even in our ability to compare quantified representations, but rather the question is how to do this in a way that provides the most utility for a particular application.

4.4 A Method for Recommending Feedback

We propose here a method for recommending feedback messages for teachers in response to their student open-ended work. As described in earlier sections, the challenge here is largely in the structure of the data. Essentially, for an observed open response problem P_0, we begin with a new student answer A_n for which we need to recommend an appropriate feedback message. As we have never seen A_n before (i.e. problem P_0 has never been answered with the exact response of A_n in the history of the learning system), we are unable to simply apply a "lookup" process; there do exist some cases where this may be possible (considering the cases where A_n is a blank response or simply "idk"), but for this formulation we will assume that A_n is not among these special cases. Instead, we must compare this student answer to all other known student answers for which we have a paired feedback message: $(A, F)_{0 \ldots n-1}$.

For each of these prior answer-feedback pairs, we compare the quantified representations of A_n and $A_{0 \ldots n-1}$ using similarity procedure S to calculate a single-valued distance measure $D_{n,0...n-1}$ for each pair-wise comparison. These distance measures are then sorted from most-similar to least-similar and the top R are selected as recommendations to the teacher; following the format of Google's SmartReply, R here would equal 3, meaning the feedback associated with the three most-similar prior answers are selected to display to the teacher.

In describing this proposed method, we must also make some reasonable assumptions. First, we must assume that we have a sizeable pool of collected student answers for a particular problem of interest, and that those student answers also be accompanied by appropriate feedback messages; as in the case of many systems that support open-ended problems and the writing of teacher feedback, this pool of appropriate answer-feedback pairs is likely to vary in size depending on the problem and may be a limiting factor of the method's effectiveness. Second, we must make the assumption that a feedback message given to one student answer is also appropriate for a "similar" student answer; while this definition may vary, as previously described, we further present and apply an evaluation procedure that allows for further comparison into the utility of different methods. The risk of this last assumption can also be mitigated through a human-in-the-loop design where the feedback is recommended for the teacher rather than being fully automated.

A further assumption here is that we have an appropriate similarity procedure S that produces a meaningful distance value for each comparison. While this is perhaps the largest assumption, it is also one that can be tested and evaluated. In reality, we have access to a large number of procedures, $S_{0...s}$, each representing a potential "recommendation policy" to identify appropriate feedback messages. In comparing different policies, however, we need a ground-truth value of similarity as defined by teachers with which we can compare. The method by which we calculate this value is described in the next section.

Table 4.1: Sample Student Answers for a Single Problem and Associated Teacher Response Categories

Student Answer	**Each Teacher's Category:** **Category T1, Category T2, Category T3**
I divided 7.5 by .75 and got 10 then I did 10 times 3.45 and got 34.5.	C, B, A
I divided 3/4 by 3.45 and got 0.21 then i multiplied 3.45 by 0.21 until i got 7.5	I, D, C
I know this because I divided 3.45 by 3/4 and got 4.6 and times 4.6 by 7.5 and got 34.5	C, A, A
I did 3.45 divided by 0.75. I got 0.75 because 3 divided by 4 is 0.75. When I divided 3.45 and 0.75, I got 4.6. I then did 4.6 multiplied by 7.5 and got 34.5.	C, A, A

4.5 Evaluating Recommendation Policies

In order to evaluate recommendation policies, defined in the previous section as $S_{0 \ldots s}$, there are potentially online and offline methods that can be used. To evaluate the policies in an online sense, we could simply build the proposed system, and compare the effectiveness of policies based on how often the recommendations are chosen by teachers. There are of course several issues with this method in that it could take a long time to evaluate a large number of policies. Ideally, we would want to use an offline method, effectively simulating or approximating teachers' choices of recommendations; in this way, a large number of policies can be evaluated simultaneously using a common dataset. While only an approximation of how teachers would utilize recommendations, offline methods are often used to first filter the number of likely-optimal policies to a small number of candidates that are then further evaluated in an online manner.

In this work, we evaluate the proposed policy in an offline manner using a dataset constructed through close collaboration with a cohort of 17 teachers from across the United States. The goal in constructing this dataset was to develop a measure representing similarity as defined by the group of teachers as a whole. Having such a measure provides a ground-truth value of similarity with which we can compare our recommendation policy distance values.

The data was constructed by first sampling student answers to open-ended problems from widely-assigned open educational resources (OER) in the context of middle school mathematics. We then randomized these responses, grouping them with other student answers from the same problem and presented them to subsets of the teachers. With these responses, per problem, we asked the teachers to group the responses into any number of desired categories. We gave no further instruction regarding how to group the student answers nor the number of categories to use. In this way, the teachers could decide, through any conscious or subconscious processes, how to identify "similar" answers by placing them within the same abstract category. Not only this, but since multiple teachers performed this categorization for the same set of responses and problems, we also are able to capture variation in how teachers define and identify similarity. There were initially 78 distinct open-ended problems with sampled student answers, but was ultimately filtered to 67 problems due to some problems having been categorized by fewer than 2 teachers. As a final filtering step, empty student responses were also dropped from the dataset. After all filtering, there were a total of 5,539 student answers across 67 problems. A sample of these responses and their associated teacher response categories are presented in Table 4.1. The category name in that table, denoted by a letter, has only abstract meaning within that teacher's set of categories (i.e. Teacher 3's "A" category in the first row is distinctive from Teacher 2's "A" category in the last row).

From Table 4.1 3 separate student responses are presented from the same problem. In this case, 34.5 is the correct answer. Within the table you can see for this problem, Teacher 1 has used the category C when a student provided a correct answer. Therefore we can infer student answers with the category C, the teacher considered similar, and would elicit a similar response.

With this data, we constructed a metric which we call the *Teacher Agreement Score* (TAS). This value is calculated for a given recommendation policy by first applying the proposed recommendation method presented in Section 4.4 to generate the top R most-similar responses (where $R = 3$ in our particular evaluation) from a selected holdout answer as A_n. From these selected answers, the sample-level TAS is calculated as follows:

$$\text{TAS}_i = \sum_{j=0}^{R} \frac{1}{T} \sum_{t=0}^{T} int(C_i = C_j) \tag{4.1}$$

This equation calculates TAS for holdout sample i by comparing the teacher-given categories of this response in comparison to the categories provided for the selected R responses for all teachers T with provided categories. This process is then repeated in a hold-one-out manner (observing each student answer as the selected holdout) and an average TAS is calculated for the observed policy in regard to the given problem. Finally, this process is repeated across all 67 problems and an average and per-problem TAS is used to compare each policy.

To give an example of this calculation, consider Table 4.1. If the last row was used as a holdout sample and the first 3 rows were the identified 3 most-similar responses, the calculated TAS_i for this sample would be 0.556. This is, again, calculated by comparing for matching categories within each teacher for each response; the categories match for 2/3 teachers when comparing to the first row, 0/3 for the second row, and 3/3 when comparing to the third row. These values are then simply averaged to find the 0.556 value. The process would then continue by rotating the holdout sample.

Ultimately, a TAS close to 1 suggests that the observed policy agrees with how teachers would define similar student responses. In this way, policies exhibiting higher scores are, in theory, more likely to be utilized by teachers. This theory can be tested through online evaluation once a number of policies have been evaluated.

4.6 Empirical Analysis: Comparing Recommendation Policies

Now that we have defined both our proposed method for recommending feedback as well as our evaluation derived from real data, we present an empirical analysis to both exemplify these methods as well as compare several potential recommendation policies of varying complexity. The methods explored in this analysis are described in this section.

4.6.1 Universal Sentence Encoder

As introduced in the Background Section, several NLP methods of representing text have grown in popularity for their ability to capture the semantic meaning of not only words, but also full sentences and even paragraphs. The first method that we explore within our empirical analysis is the Universal Sentence Encoder [CYK+18] (USE). While other NLP methods often build numeric representations of individual words, the USE builds a single vector representation for a given sequence of words within a high-dimensional vector.

Once a sentence-level embedding is generated for each response, a distance measure (described below) can be applied to measure the "closeness" of other student answers in vector-space. As this method is meant to capture the semantic meaning of the sentence, and leverages complex deep learning methods to do so, this method has the potential to allow for comparisons beyond the surface-level features of the text.

4.6.2 Sentence-BERT

Developed even more recently and arguably considered to be the current state-of-the-art of sentence representation is the second method of comparison: Sentence-BERT [RG19a]. Developed from the word-level representation method of BERT, this method constructs a high dimensional vector representation of sentence- or paragraph-level text similar to that of the Universal Sentence Encoder. This method, however, is based on what is known as a “siamese network” architecture. This type of network attempts to incorporate textual and semantic similarity into the generated embeddings. In this way, this method represents the most complex of representation methods compared in the current analysis.

4.6.3 Levenshtein Ratio

Among the simplest methods of comparing the likeness of two samples of text is that of Levenshtein Distance. This approach examines strings of characters and calculates a distance based on how many need to be changed to turn one string into its comparison string. For example, if a student A said ‘the answer is 45’ and student B submitted an answer with ‘the answer is 46’, the distance would be 1. However, if student B answered with ‘I think the answer is 46’, the distance would be 9. Clearly, there are disadvantages to this approach, mainly the distance could be larger between two answers, but their content is the same. However, when considering a character level distance metric, could this out perform more modern approaches? For the purposes of the paper, we utilize the Levenshtein Ratio which calculates the distance and converts it to a similarity ratio which is meant to account for the comparison of strings of different lengths.

This method acts as a baseline comparison method due to the simplicity of the approach. However, it is likely that surface-features of text (i.e. the use of particular key words within student answers) may actually prove to be a highly-weighted attribute among the teacher comparisons.

4.6.4 Distance Metrics for Similarity

While the above methods generate representations of student answers, the method of calculating the distance between representations is still needed. In this regard, we observe three different methods within this analysis: Euclidean Distance, Cosine Similarity, and Canberra Distance; these were chosen both for their prominent usage in previous NLP research and also for their notable differences in meaning. As described in an earlier section, Euclidean Distance observes the magnitude of the geometric distance between two vectors while Cosine similarity observes the difference in angles produced by two representation vectors. Canberra Distance, while not as widely known as the other two, has been applied in areas of computer science as a means of comparing ranked lists [JRVF09]. Each of these distance measures are applied to the above representation methods (excluding the Levenshtein Ratio) and the TAS measure is calculated for each as described in Section 4.5.

4.7 Results

As mentioned earlier, after all the filtering, there were 67 total problems with 5,539 student answers. From this, we calculated the overall teacher agreement scores for each approach and distance measure as shown in Table 4.2. First, it is clear that, in general, a contextual approach like sentence and paragraph embeddings

[1]It should be noted that the 'Number of Times Best Teacher Score Agreement Score' sums to over 76/67. This occurs because there were 9 cases where two approaches scored the same Teacher Agreement Score for that problem. Thus, either of the approaches would be considered acceptable.

Table 4.2: Overall Teacher Agreement Scores

Embedding Vector Type	**Distance Metric**	**Average TAS**	**Std. Deviation**
N/A	Levenshtein Ratio	0.536	0.107
Universal Sentence Encoder	Euclidean	0.556	0.108
Universal Sentence Encoder	Canberra	0.554	0.112
Universal Sentence Encoder	Cosine	0.556	0.108
Sentence-BERT	Euclidean	0.621	0.105
Sentence-BERT	Canberra	0.623	0.106
Sentence-BERT	Cosine	0.623	0.105

Table 4.3: Teacher Agreement Scores Per Problem Performance

Embedding Vector Type	**Distance Metric**	**% Best Teacher Agreement Score**	**Number of Times Best Teacher Agreement Score for Problem**[1]
N/A	Levenshtein	1.492	1/67
Universal Sentence Encoder	Euclidean	11.94	8/67
Universal Sentence Encoder	Cosine	11.94	8/67
Universal Sentence Encoder	Canberra	4.48	3/67
Sentence-BERT	Euclidean	17.91	12/67
Sentence-BERT	Cosine	25.37	17/67
Sentence-BERT	Canberra	40.30	27/67

performed stronger than a character level model like Levenshtein ratio. Overall, each embedding vector approach outperformed the Levenshtein ratio in terms of selecting similar student answers with the same teacher categories (per our TAS measure). Table 4.2 indicates that when utilizing a Levenshtein ratio for identifying similarity amongst student responses, it manages to agree with teachers in about 53.6% cases. While the character level Levenshtein approach managed to agree with teachers over half the time, it was below all embedding vector approaches.

With the well known universal sentence encoder, there was a bump in the ability to identify similar student answers which teachers agree are similar. When using cosine and euclidean distance measures, the universal sentence encoder saw its largest jump from Levenshtein, amassing an average overall teacher agreement score of 55.6%. However, when using the canberra distance measure there was a drop of 0.2% in the universal sentence encorders ability to select similar student answer which teachers agree; obtaining an overall teacher agreement score of 55.4%. It should also be noted that with the canberra distance measure the universal sentence encoder experienced its largest standard deviation across any approach with a standard deviation of 0.112. While it still selected similar student answers, that teachers agreed with 55.4% of the time, it varied more often in its ability to agree with teachers per each problem.

Overall, the strongest performing approach, in terms of teacher agreement scores, was Sentence-BERT. Consistently, Sentence-BERT managed the highest average teacher agreement score across all the problems with all distance measures. Additionally, it generated the lowest variation in teacher agreement scores. With a standard deviation of 0.105, 0.106 and 0.105 for euclidean, canberra and cosine distance measures, respectively, sentence-BERT more consistently selected similar responses which teachers agreed with as compared to all other models.

What is evident is that different combinations of sentence/paragraph embedding vector representations and how the distance is calculated varies our ability to suggest suitable similar student answers. Table 4.2 clearly shows this. While this exhibits our ability to evaluate our similarity calculations, and could be scaled to future answers within the same problem, we set out to see how the models performed not just overall, but on a per problem basis. Table 4.3 provides a breakdown of the performance of each combination of sentence/paragraph embedding vectorization and distance metrics at a per problem level. What is apparent is that there is not a policy which dominates all other methods. Every approach, from combination of sentence/paragraph embedding vectors and multiple distance measures to just using a simple Levenshtein ratio, manage to agree with teachers the most on at least one problem. Even though it may only be one problem, until the other approaches manage to agree with teachers more often, and beat the Levenshtein ratio approach, the Levenshtein ratio would continue to be used. Overall, utilizing sentence-BERT managed to have the most agreement with teachers on which student answers were similar. What is also apparent is the number of problems which sentence-BERT performed well varied amongst distance measures. When using canberra to calculate the distance between the vectors, it managed to have the highest Teacher Agreement Score with 27 out of the 67 problems. As compared to utilizing cosine and euclidean distance measures, which only managed to have the highest Teacher Agreement Score on 17/67 problems and 12/67 respectively. It should also be noted that the *Number of Times Best Teacher Agreement Score for Problem* in Table 4.3 will total to over 67 problems by 9. This is because there were 9 cases where two policies could be deemed acceptable for a problem; they had the same teacher agreement score.

In the end, it is evident that there is not a single policy which agrees the most with teachers on which student answers are the most similar, but sentence-BERT combined with Canberra distance is perhaps the closest of those methods explored in this work. There is a wide distribution of problems which certain method combinations out perform others, but then there are many problems in which they struggle. From this study we are able to identify those methods and problems and select when the Levenshtein ratio should be used vs universal sentence encoder or the sentence-BERT. We can use these approaches with future unseen responses (for this set of problems). By utilizing our validation results from the teacher agreement scores, we can choose the best method, find the most similar current problem we have seen and select the teacher responses associated with that student answer as the teacher response for the new answer.

4.8 Discussion

In this research we set out to develop a system to assist teachers in providing feedback to their students within open-ended questions in mathematics. To that end, we have shown our dataset collected enabled us to evaluate our suggested similar student answers. By being able to compare our suggestions to real teacher choices, we can effectively choose which policies are best for each problem. This automated approach sets forth a strategy which can be used to suggest teacher responses to an incoming new student answer.

It is apparent that this strategy is beneficial to developing a quality teacher response to a student's answer. What was evident was that there wasn't a single policy that outperformed all the other policies. There was a large distribution across all the policies. It was so wide spread that even a simple policy, such as utilizing just Levenshtein ratio, managed to beat out other, more complex, policies on a problem. With this information in hand, when a new student answer comes in for said problem, the system would look to the most similar answers when utilizing the Levenshtein ratio.

4.9 Limitations and Future work

While our approach is a jump in the right direction for automatically suggesting teacher responses to student answers, we are limited to the set of problems which we have already seen. It is the case if an answer comes through which hasn't been seen before, we will be unable to suggest a comment.

Also as it pertains to handling answers to problems we haven't seen, its clear that's something this approach can not handle. However, our future work looks to try and utilize the problem body's to find similar problems and explore their answers we have seen.

Currently, we are able to clearly evaluate our policies, and while all of them agree with teachers more than 50% of the time, our best performing policy agrees with teachers 62.3% of the time. While this is a good result, we hope to continue increasing that percentage. We hope to continue adapting our similarity calculations to more accurately identify similar answers. There are additional approaches which we could explore as well, such as Doc2Vec.

4.10 Conclusion

In the end, this study presented three main contributions, developing a method which provides teachers with automated feedback messages, developing an evaluation procedure to validate our automatically generated feedback policies in an offline manner, and finally, comparing these different policies. We have shown that our procedure can identify, offline, which policies provide the most accurate automated feedback to students.

While more modern natural language processing methods performed better overall in suggesting teacher comments to new student answers, our new validation method enabled us to identify those problems where a simpler policy may perform better than a more modern, deeper, approach. This way we can tailor our suggested teacher responses to student answers with the strongest performing policy for that problem.

By automating open-ended questions, teachers can utilize a more diversified set of questions. With a procedure in place that accurately, automatically and immediately suggests a response for a teacher, ITS can utilize this procedure to wed the benefits of a close-ended questions and open-ended questions. With modern natural language processing, this automated suggestion system can make a, previously, more costly problem type more attainable.

Chapter 5

What Kind of Tone is That? A Sentiment Analysis of Teacher Feedback

Erickson, J. A., Botelho, A. F., & Heffernan, N. T. (2021, April) What Kind of Tone is That? A Sentiment Analysis of Teacher Feedback. Manuscript prepared.

Abstract

In recent years, intelligent tutoring systems have begun to expand their question type support from structured questions, with easily identifiable answers, to open response questions. While these systems continue to adopt tools and algorithms to create automation within open response questions, teachers continue to primarily write genuine feedback to students. With these questions, the different answers students could provide are infinite and the types of feedback teachers could provide are infinite. So to be able to develop impactful and useful tools to automate this process, it is imperative to undestand what types of feedback do teachers provide. Mainly, are they

positive or negative in their feedback to students. Understanding this can add a useful variable for use in this automated process. This research performs an exploratory analysis of the feedback teachers are providing students within open response mathematical questions. More specifically, this research aims to generate and explore the sentiment teachers use when providing feedback. We explored the distribution of the sentiment associated with each teacher and whether teachers show any identifiable patterns or variations with the sentiment they use in their feedback. Additionally, we explore whether the scores students are receiving influences or elicits different levels of sentiment in teacher feedback. Essentially, asking the question whether teachers are more positive, negative, or neutral when responding to a student's answer within an open response question and whether or not the sentiment a teacher uses is impacted by the grades. Lastly, we attempt to utilize the sentiment categories to generate an algorithm which takes a student's answer and suggests the sentiment it will elicit from a teacher. Additionally, we look to see how accurately we can predict the sentiment each student answer would evoke from a teacher. An effective predictive model of sentiment of teacher's feedback would be a strong supplemental variable for an automated open response tool. In the end, we were able to show that there are identifiable patterns which suggest teachers sentiment in feedback to students is impacted by the grades. In addition, we found that there is variation amongst teachers in the type of sentiment used in their feedback. Formally, we developed models which can automatically, and accurately, identify what sentiment a teacher would use.

5.1 Introduction

Intelligent tutoring systems have been around for a time [CKA97] and have garnered popularity in recent years. Recently, that popularity has exploded and teacher's reliance on the systems have grown given the current state of the world. With this reliance, teachers have looked to intelligent tutoring systems (ITS) such as ASSISTments, McGraw Hill's ALEKS$^{\text{TM}}$ and/or Carnegie Learning's Cognitive Tutor$^{\text{TM}}$ to ease the hindrances of online teaching and enable more automation. These systems look to utilize machine learning and computer engineering to develop tools which immediately provide feedback to students, automate the grading process, and generate detailed reports on student performance. While these systems have lived up to their promise to provide these tools, most have been limited to questions with well defined answers such as multiple choice or fill in the blank. However, once a teacher wishes to expand the question types which they are assigning students, most the benefits of an online ITS begin to wane.

While questions with well defined answers benefit teachers because of the associated automated tools, its difficult to directly infer a student's comprehension aside from the general assumption that other students who have selected the same answer have often misunderstood a common concept. In our previous work, [EBM+20] we identified that teachers within ASSISTments wanted to expand their assigned question types from well defined to less defined open response questions which allow students to explain their answers and reasoning. Many teachers assigned open response questions, but a large number of them were never graded and even more didn't receive feedback. As the school year proceeded and teachers became busier, they began to rely more heavily on well defined questions. So while teachers wanted to gain a deeper insight into the students processes of thinking with open response questions, they were drawn back to well defined questions. To bridge this gap, tools need to be developed support the automation of less defined questions to a level similar in which current ITS support well defined questions.

In recent years, many ITS systems have adopted support for less defined, open response, question types for teachers to assign. To develop an effective tool which helps to bridge the gap between the automation of well defined questions and less defined open response questions, there are two aspects which need to be addressed: automated grading and automated feedback which a teacher can present a student. Our past research [EBM+20] addressed the automatic grading of open response questions. As we have turned our focus to providing automatic feedback to teachers, we began to ask the question, what type of feedback are teachers providing? In recent years, natural language processing (NLP) has been utilizing sentiment analysis across various fields. To gain a deeper understanding of the type of feedback teachers are providing, this research will be adopting sentiment tools to explore whether there are identifiable patterns, or levels of variations, within the type of sentiment a teacher uses in feedback and if the grades student's receive impact the level of sentiment which a teacher will provide. So we look to identify whether teachers tend to provide feedback with a negative, neutral or positive sentiment and whether or not that sentiment utilized is impacted by the grade the student received.

To generate an effective tool for automatically suggesting feedback for teachers, it isn't enough to just identify the trends and variations with sentiment, but be able to automatically predict what level of sentiment a student's response would elicit from a teacher. For an automatic feedback system to be impactful or useful for a teacher, it should not solely suggest similar, generic, feedback. There needs to be diversity amongst the suggested feedback. Not only should the diversity be amongst the content, but the tone in which feedback is being provided. By having a tool which can identify what level of sentiment a teacher should provide, suggested responses can be filtered down another level to have more diverse and more specific feedback.

The automation of feedback is a multi layer task, with multiple filtering steps and multiple machine learned properties. By accounting for sentiment, a more accurate, useful and impactful tool can be developed. This paper aims to present and explore:

1. Whether there are identifiable trends or variations associated with the sentiment teachers use when providing feedback
2. Whether or not different grades elicit different levels of sentiment and can we identify them
3. Whether we are able to develop models which can predict what level of sentiment a teacher should use within their feedback

5.2 Background

5.2.1 Open Response v Multiple Choice

As it has been noted, most ITS systems have adopted support for open response questions and that there are a growing number of systems which are attempting to integrate the automation of open response question assessment and feedback c.f. [AB06],[FLL99],[ZZX$^+$17]. This is because many of these systems are hoping to become a more well rounded system. To achieve this, systems must treat the student holistically, and aim to support different types of problems. This is echoed by studies studies such as [OBKM13] which showed when open response and multiple choice questions are provided, each will extract different aspects of the student's learning process and understanding of the materials.

Another work [RIM05] presented a study that showed when students worked on multiple choice questions, they were more likely to gain a sense of being correct even when they are wrong. The authors noted that when the students are being presented incorrect answers, and choosing those answers, it may create a false sense of security and imprint the wrong answer in the student's mind. This occurs because multiple choice questions explicitly and intentionally show wrong answers and attempt to convince student they're correct. As [BR08] noted, this approach is performed to lower the likelihood that students are able to guess the right answers; however, there is a balance that needs to be struck because their study found that these 'lure answers' in exams can lead to students holding onto incorrect answers. It should be noted, however, that [BR08] found that the negative effects, students holding onto purposefully presented incorrect answers, can be alleviated by providing feedback to students. This is the main feature of ITS systems such as ASSISTments [HH14]. Additionally, this ITS was proven to significantly increase end of the year test scores on standardize mathematics exams in comparison to traditional homework[RFMM16].

5.2.2 Word and Sentence Representations

In recent years, NLP has made great strides with numerical representations of corpus' for machine learning tasks. From the beginning, there were simplistic word level numeric representations such as a simple word count (often referred to as a Bag of Words approach (BOW)). As the method states, this approach takes the text and counts the number of occurrences within the entire corpus each individual word occurs. Then a list containing the count for each word is used and modeled with. Even though BOW is a simplistic method, there are numerous studies which utilize this method as the foundation of their extended work [ZJZ10][ZM17][SSW$^+$11].

However, there is a clear negative to such a simplistic approach. Words which occur most often will be given the most weight in the model. While on the surface this approach holds water, words which occur frequently do not provide insight into what the text means, just that they are more often used and most likely to impact a model's analysis. These words are commonly referred to as stops words [WS92] and these are often just dropped; however even with the words dropped, there's a large chance other words, that should have lower impact on the models decision making, will still have a larger count value. From this, a variation of the word count approach was adopted called the Term Frequency Inverse Document Frequency (often referred to as TF-IDF) calculation. In short, the TF-IDF will start by counting the occurrences each word (the *term frequency*) in each of the documents (as an example, we will refer to this as the student answer), then those frequencies are multiplied by the inverse document frequency, or the log inverse count of the number of student answers which the word occurs in. This approach has been utilized in natural language processing for years in studies such as [TMD14].

While these BOW based approaches have the benefit of simplicity, and easy interpretation, they are unable to provide any sort of relational information. With a standard BOW based approach, there isn't any relation or contextual inferences that can be made based on the values. The word 'dog' occurring frequently and the word 'table' occurring equally as frequently does not mean these are similar words. However, a machine learned algorithm would consider them closely related based on their weight (with a simple BOW approach). Deep learning has provided a solution for this, with the development of word embeddings. Simply put, a deep learning algorithm performs unsupervised learning to develop n-dimensional (can choose the size of the vectors) vector representations of words. From this, where the words lie within the vector space provides the contextual and relational information to a machine learning algorithm. For instance, if two words are close within the vector space then they are considered closely related. There are two common paths to working with embeddings, either researchers have enough data to generate their own embeddings, or they can utilize pre-trained embeddings. Luckily Stanford researchers developed their own approach for generating word embeddings, referred to as GloVe [PSM14], and shared a pre-trained model trained on either Wikipedia, Common Crawls, or Twitter. Similarly, google developed an algorithm to generate word embeddings referred to as word2vec[MCCD13] and release pre-trained embeddings trained upon google news. More recently, BERT [DCLT18] has become a popular word embedding algorithm as well and also has a pre-trained version. With the publication of these freely available pre-trained embeddings, researchers with less robust datasets and corpuses can utilize these embeddings to represent the words. Thus, generating much more accurate representations of words and consequently more accurate contextual inferences can be made about the word.

Not all is perfect, however, with these pre-trained word embeddings. Precautions need to be taken because research has been done [BCZ+16] to show that there are implicit bias in the pre-trained embeddings from the datasets which they were trained on. This study managed to show that within the pre-trained word2vec embeddings there were biases which researchers may not want within their modeling. Mainly, the research discovered that when the embeddings were developed from google new, words were being related in a very bias way. Often, gender bias were present, for example *he* was most closely related to *doctor* and *she* was closely related to *nurse*.

Not only are embeddings at the word level, recently, sentence embedding algorithms have gained popularity for their ability to generate single vector representations for entire sentences. This greatly simplifies the training data. Instead of having, for instance, a 300-dimensional vector for each word in every sentence within the entire document, there will be a single n-dimensional (changes size based on what algorithm or pre-trained approach utilized) vector for each sentence in the entire corpus. The two most pronounced approaches utilized today are the Universal Sentence Encoder (USE) [CYK+18] and SBERT (sentence BERT) [RG19b].

5.2.3 Types of Teacher Feedback

As teachers provide feedback to students, some teachers may provide a little more detail and other may be a bit more general. Many variables can affect a teachers feedback, but studies such as [Wea06] found that not all feedback were equal; there was a clear hindrance of feedback to students which was either considered too simplistic or vague. Additionally, the [Wea06] study found that students felt that detailed constructive feedback was the most helpful. Again, this shows that a generic feedback isn't always going to benefit students and that time should be spent personalizing and varying the levels of sentiment in feedback. The students in that study echoed the same thoughts, that there needs to diversity amongst the sentiment provided within feedback.

Another study focused on the type of feedback students respond to best [YPVG+14]. More specifically, this study provided minority students with feedback which is more blunt and states clearly that the teachers expected a certain level of performance from the students. The study exhibited that minority students managed to trust their teachers more when they use straight forward feedback which states their expectations of the students. While the minority students gained more trust, they showed improvement in their work as well.

5.2.4 Sentiment Analysis

While its clear that there are many different types of feedback that can impact a students learning, and that there are debates over which helps students learn the most, it is evident that the way a teacher responds to a student affects the student. This is why sentiment analysis of teacher feedback is imperative.

NLP has taken many strides in developing tools to interpret the sentiment of sentences and corpuses. A well known approach, the Valence Aware Dictionary and sEntiment Reasoner (also known simply as VADER) is a rule based algorithm, developed with social media data [GH14], that manages to beat out many more complex approaches. Social media is a common place for sentiment analysis, another study [AXV$^+$11] pulled 11,875 tweets from Twitter and had them labeled manually. This was then used to develop and compare 3 different sentiment approaches, including a tree based kernal approach (similar to a rule based approach), a unigram model and a feature based approach. Both the feature based and tree based approaches managed to outperform the baseline unigram approach. Similarly, another study [PP10] utilize data collected from Twitter, and tagged the part of speech with a TreeTagger, to develop their own sentiment classifier using Naive Bayes and a Support Vector Machine models.

Another well known sentiment tool was developed by Stanford which combined, what they referred to as, the Sentiment Treebank and their own Recursive Neural Tensor Network[SPW$^+$13]. The Stanford Sentiment Treebank is a large corpus with completely labeled parsed trees of movie reviews. These are then used within the Recursive Neural Tensor Network to more accurately decode intricate sentiment.

In the end its clear that many sentiment analysis have a foundation of rule based or tree structures. Additionally, a good amount of them are trained at some point from social media data to accurately categorize sentiment. Approaches, such as the

Stanford Sentiment Treebank and VADER have the added benefit of making these models publicly available. Thus, the models can be utilized by any researcher to help interpret and identify sentiment within their own corpus.

5.3 Dataset

For this research, we look to analyze the sentiment of feedback from 17 teachers across the United States. This dataset was collected to identify and explore the variations within teacher feedback sentiment given a students open response answer. From this, teachers were given a students answers and were expected to write what they would provide as feedback to the student.

The collection of this data set started by generating our own corpus of student answers to less defined open response questions in mathematics. Of these problems, all of them were selected from open educational resources (often referred to as OER's), a common resource in education today. We collected data from middle school students. From there, we took all the student answers, randomized them, and grouped them together by the problem. Thus each student answer was grouped with all other answers from the same problem. Each of the teachers were split into subset groups and received a set of student answers. The subset would then generate a feedback message (required) and grade (not required) to the student.

An imperative part of this dataset, and for this study, is that for each student answer, all teachers would generate feedback. This meant that each student answer would elicit a feedback message from each of the teachers within the subset. So if the randomized student answers in problem A were given to the teachers in subset 4, and this subset had 3 teachers, all 3 would provide their own genuine feedback for each given answer. With this, we could account for more of the variability within teacher feedback by being able to observe multiple teachers perspectives for a single student answer. This would allow us to identify how, or if, there were identifiable variances in the types of sentiment teachers provided. Additionally, this would allow us to examine whether there were there teachers who predominately showed more negative, neutral, or positive sentiment.

In the raw state, the dataset consisted of 78 unique problems. All of these were open response mathematics questions. This dataset was eventually filtered down from the 78 unique problems to 67 to account for the 11 problems in which fewer than 2 of the teachers performed the task of providing feedback. After all the filtering was completed, within the 67 unique problems there were a total of 5,539 student answers, 5,038 of which were unique. This difference was because some students answered with the same content. Additionally, the total number of teachers who provided feedback was filtered down to 11. A sample of authentic student answers and teacher feedback to the same question are presented in Table 5.1.

5.4 Exploratory Analysis of Teacher Feedback Sentiment

With a dataset consisting of multiple detailed teacher feedback for each student answer, as shown with examples Table 5.1, we have the ability to explore the different levels of sentiment teachers provided students. Additionally, Table 5.1 shows not only the detail, but the variation in the content. The way each teacher approaches the student's response is unique. We attempt identify whether teachers showed any variation amongst these feedback not just in content, but by attempting to identify if teachers use positive feedback as compared to others or if there are teachers who tend to provide negative sentiment.

Table 5.1: Sample Student Answers and Authentic Teacher Feedback

Student Answer	**Teacher**	**Teacher Feedback**
It would mean for 5 t-shirts it would cost a total of 110 dollars.	T1	Great job! You correctly identified that (5,110) tells us that 5 t shirts would cost $110 or identified at least two points that would be on the graph of this proportional relationship.
I don't know	T2	You should always attempt to answer the question or ask for assistance to determine what the problem is asking. Use your knowledge of constant of proportionality, proportions and/or equivalent ratios to help you solve this problem.
5 shirts for $110.	T3	The question here was, "What point(s) must be on the graph for the quantities to be proportional to each other?" Unfortunately, your answer doesn't address that question. You answered Part B of this question.

Additionally, we explore whether there is evidence to suggest the sentiment teachers utilize on feedback is impacted by grades. Essentially, we are attempting to explore whether a student receiving a grade of 0 receives differing levels of sentiment in feedback as compared to a student getting a 75% or higher? An ability to identify variant levels of sentiment across teachers would support the adoption of sentiment in tools attempting to automatically suggest diverse tailored feedback to student open response answers within ITS.

5.4.1 Methodology

VADER

As mentioned earlier, recent advancements in NLP has seen an increase in sentiment analysis studies and pre-trained models which have been trained on large social media datasets. For this analysis, our corpus wasn't robust enough to warrant training our own sentiment model. Additionally, while our corpus is decently sized in the number of total student responses and teacher feedback, the length of the content isn't usually more than a sentence or few. Thus, training our own sentiment model would likely struggle to identify sentiment accurately.

To develop VADER, the authors [GH14] generated a list of over 9,000 lexical features (words). The authors had Amazon Mechanical Turk workers rate the sentiment of each of these lexical features, providing the authors a labeled dataset of 90,000+ labeled lexical features. Additionally, [GH14] used two human raters to review 800 tweets and evaluate the level of sentiment each tweet carried. From this, the authors managed to identify 5 aspects of the tweets which caused the level of sentiment to change (a negative to a more positive sentiment for example). First, the use of punctuation's, for example, can directly impact the sentiment of a message. A sentence which ends with a '!' projects a very different sentiment than '.'. For instance, 'I can't wait to see you!!' vs. 'I can't wait to see you.' projects a different level of intensity in the sentiment. Other rules [GH14] noted were: the capitalization greatly changes the intensity of the sentiment, contrastive conjunctions ('but') can be used to identify a change in the tone and sentiment, the use of more intense words (the authors referred to them as 'Degree Modifiers') such as 'very' can impact the sentiment, and when examining the previous 3 words (tri-gram) (before an identified lexical feature by the authors) over 90% of the time it can identify a change in the sentiment.

VADER produces numerous outputs, namely, a value for negative, neutral and positive sentiment. Each of these are calculated as the percentage of words within the sentence that are either considered negative, neutral or positive. For this work, we will be exploiting the *Compound Score*, which utilizes the rules mentioned above to generate and adjust the sentiment scores. They are then normalized between the values of -1 to 1 such that -1 is the representation of the extreme negative and 1 the extreme positive sentiment. Once generated, the VADER compound score can be categorized into 3 distinctive categories; negative, neutral and positive sentiment when the score is less than or equal to -0.05, greater than -0.05 and less than 0.05, and greater than or equal to 0.05, respectively.

Generating the Sentiment Scores

To perform our exploratory analysis, VADER was applied to all the teacher feedback across all 68 problem's. However, it should be noted that VADER performs best when applied at a sentence level. With this as the case, any teacher feedback with more than one sentence was split at the sentence level. For example, Table 5.2 shows the 3 sets of teacher feedback from 3 different teachers in Table 5.1. Each of

Table 5.2: Example Student Answers and Authentic Teacher Feedback Sentiment Generation

Teacher Feedback	**Sentence Parsed**	**Sentence Level VADER Compound Sentiment Score**	**Teacher Level VADER Compound Sentiment Score**
Great job! You correctly identified that (5,110) tells us that 5 t shirts would cost $110 or identified at least two points that would be on the graph of this proportional relationship.	1.) Great Job! 2.) You correctly identified that (5,110) tells us that 5 t shirts would cost $110 or identified at least two points that would be on the graph of this proportional relationship	1.) 0.6588 2.) 0.0	0.3294
You should always attempt to answer the question or ask for assistance to determine what the problem is asking. Use your knowledge of constant of proportionality, proportions and/or equivalent ratios to help you solve this problem.	1.) You should always attempt to answer the question or ask for assistance to determine what the problem is asking. 2.) Use your knowledge of constant of proportionality, proportions and/or equivalent ratios to help you solve this problem.	1.) -0.4019 2.) 0.0964	-0.15275
The question here was, ”What point(s) must be on the graph for the quantities to be proportional to each other?” Unfortunately, your answer doesn’t address that question. You answered Part B of this question.	1.) The question here was, ”What point(s) must be on the graph for the quantities to be proportional to each other?” 2.) Unfortunately, your answer doesn’t address that question. 3.) You answered Part B of this question.	1.) 0.0 2.) -0.34 3.) 0.0	-0.1133

the feedback were split to the sentence level and run through VADER to generate an individual compound score for each sentence of the teacher's feedback. When looking at Table 5.2, the first two teacher's feedback each had two sentences within their feedback, thus, each individual sentence was fed through VADER and given a resulting sentence level compound score. For the other teacher's feedback, 3 compound scores were generated. At the sentence level, those compound scores only provide part of the picture. To get the overall sentiment of each of teachers feedback an overall score needs to be developed for the entire feedback. To accomplish this, we simply took the average of all the sentence level compound scores for each teacher. An example of this is presented in Table 5.2. After this entire process, there were a total of 17,480 teacher feedback messages and compound scores.

5.4.2 Exploratory Results

Once the compound scores for each sentence of each teacher's feedback was aggregated to the teacher level, the analysis could proceed. We initially analyzed the overall distribution of sentiment in Table 5.3 and found that there were a total of 3,039 teacher feedback that were categorized as negative, 6,310 that were categorized as neutral and 8,131 that were categorized as positive. From this distribution, it is apparent that within our study the teachers, based on our sentiment calculations, predominately attempted to provide positive feedback. The teachers chose, most often, to use words which carried more positive sentiment. What is apparent is that the sentiment analysis revealed that there is clear variation amongst the teachers sentiment within their feedback. Essentially, that teachers are varying the way they speak to students, and that for a automatic feedback model to be impactful or successful, being able to identify sentiment and trends is imperative.

Table 5.3: Distribution of Calculated Sentiment Across all Teacher Feedback

Sentiment	**Number of Occurrences**
Negative	3,039
Neutral	6,310
Positive	8,131

To take another step further, we dissected the sentiment at the teacher level. Essentially, we grouped the compound scores of all the feedback provided by each teacher and looked to see what the distribution of sentiment was from each teacher. Within Figure 5.1, each point on the plot is that teacher's calculated compound score for that individual feedback. For each of the points, they are shaded to show which compound scores fall above, below or within the discussed thresholds for positive, negative or neutral sentiment, respectively. What is again clear is that each teacher greatly varies in their sentiment utilized within their feedback. There appears to be a wide variation in the level of sentiment being used by teachers. For instance, Figure 5.1 shows that Teacher 9 has the widest dispersion of positive sentiment. Additionally, Teacher 9 provided the most positive feedback to a single student's answer. Other teachers, such as Teacher 4 and Teacher 11, have a much tighter dispersion of positive sentiment used, barely providing feedback with a compound score over 0.25 (it is still considered positive). Inversely, it is clear that Teacher's 3, 7 and 11 have the widest distribution of negative sentiment used. To clarify, we can see that Teacher's 3, 7 and 11 provided some of the most negative sentiment in feedback to students; providing feedback with compound scores lower than -0.50. Again, to reiterate, the stronger the sentiment, the closer the compound score will be to 1 or -1. This is evidence exhibiting that if an automatic feedback model is to be effective, taking into account the sentiment of the feedback is going to be imperative. If a teacher is going to use an automated tool, the feedback needs to seem appropriate and account for differing levels of sentiment. Not every student answer should receive the same sentiment, and this is proof teachers echo that sentiment.

Additionally, above the scatter plot are the density plots for the sentiment used. Figure 5.1, for instance, shows that in general teachers used positive sentiment more often than not. There were a few teachers, such as Teacher 4, Teacher 7 and Teacher 11, who used neutral sentiment the most. While for Teacher's 11 and 7 it is clear that neutral sentiment was used only slightly more than positive or negative sentiment, but for Teacher 4, there is a clear disparity between the amount of neutral sentiment and the negative/positive sentiment. In the end however, it is clear, again, that most often teacher's preferred using more positive sentiment in their feedback to students.

Another fascinating observation to Figure 5.1 is that while Teacher 9 had the widest range of positive compound scores (coming close to the most positive sentiment of a compound score of 1), the teacher also used the most positive sentiment in the feedback in comparison to any other teacher. Inverse of that, negative sentiment was clearly the least utilized sentiment amongst all teachers. In fact, not one teacher used negative sentiment majority of the time.

It should be noted that there are many cases of points within Figure 5.1 that lie on top of each other. For instance, when examining Teacher 4's density plot it is clear that the neutral sentiment was used majority of the time. However, the range of compound scores is quite narrow. This could be explained a couple of ways, first, VADER calculated that the language Teacher 4 used was quite similar across multiple feedback and thus warranted similar scores. The other explanation is that teachers like Teacher 4 had a common feedback they used and repeated. In the end, its most likely the smaller dispersion's are from teachers repeating favorite feedback

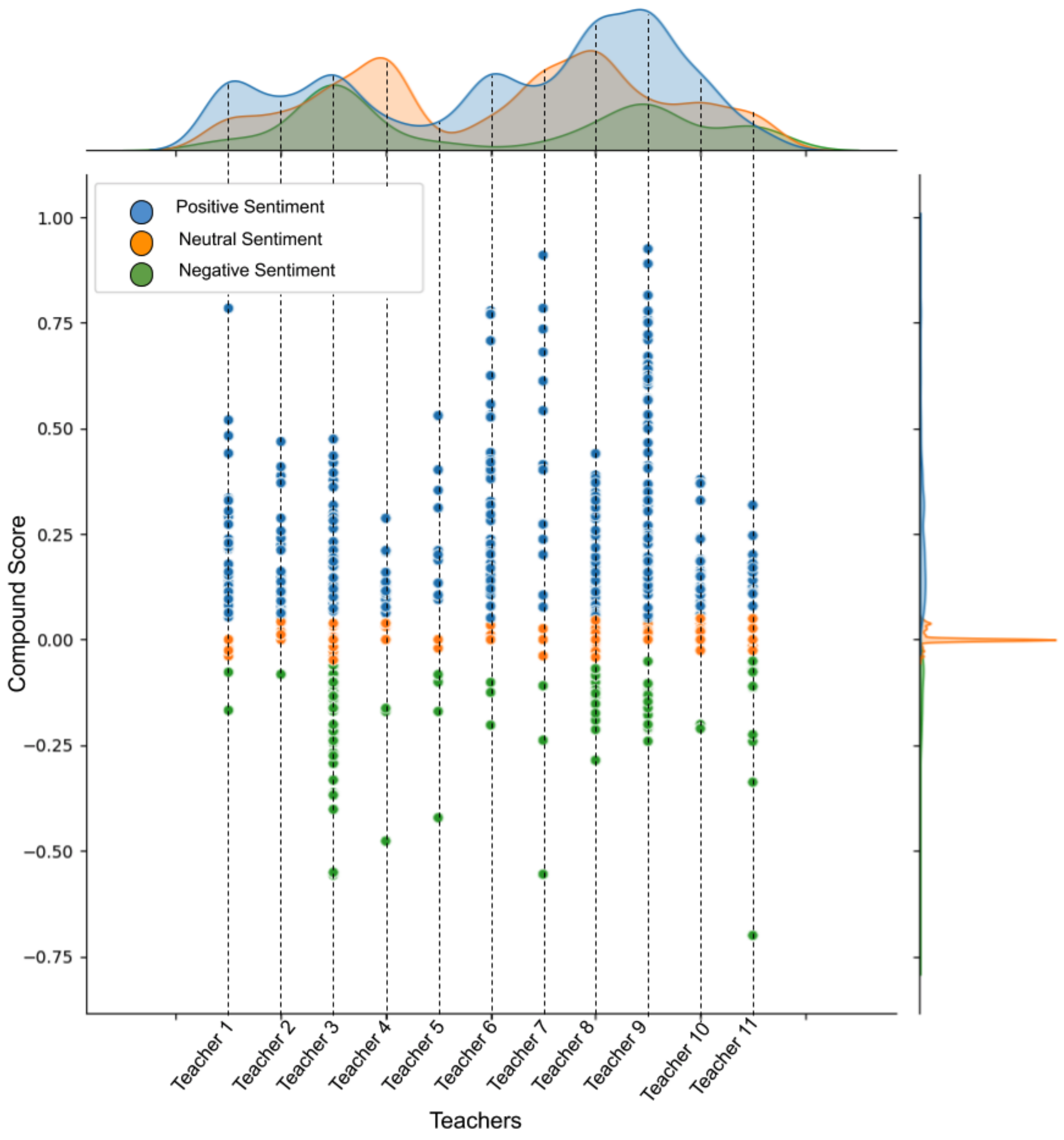


Figure 5.1: Exploratory analysis of sentiment of feedback across all unique teachers

for common missed student answers. This was evident upon further exploration into the feedback data we collected.

It is the case there are large variations in the levels of positive and negative sentiment being used by teachers and that there are large variations in the amount of each of the sentiment categories being used; however, the question has to be raised, does the grade the student received impact or influence the level of sentiment teachers used? Figure 5.2 aims to answer this question by plotting the grades student's received versus the compound score of feedback which was provided by teachers. An initial observation shows that the range of the compound scores (the intensity of the sentiment) are quite similar across all grades given out. It should be noted, however, that those who received a grade 0 had a tighter dispersion of positive sentiment compound scores. Meaning that most of the positive sentiment used for grade 0 is lower than any other grade's most positive feedback. In fact, most of the positive sentiment compound scores falls under 0.50. While this is still a pretty high compound score, the other grades have more compound sentiment scores above 0.50. Additionally, grade 0 had the lowest compound score for negative sentiment in teacher feedback.

Aside from the variation in compound score, when analyzing the density plots on top of Figure 5.2 there is a clear trend with the number of teacher feedback which carry a negative sentiment. At grade 0, more teachers used negative sentiment in their feedback than neutral or positive. Then, as the grades increased for students, the number of feedback which carried a negative sentiment from teachers decreased almost linearly. As the student's performed better, and obtained a grade of 0.25, there was a sharp increase in the amount of feedback in which teachers used more neutral or positive sentiment. In fact, at grade 0.25, its the only grade where teachers tended to use more neutral sentiment in their feedback. While neutral

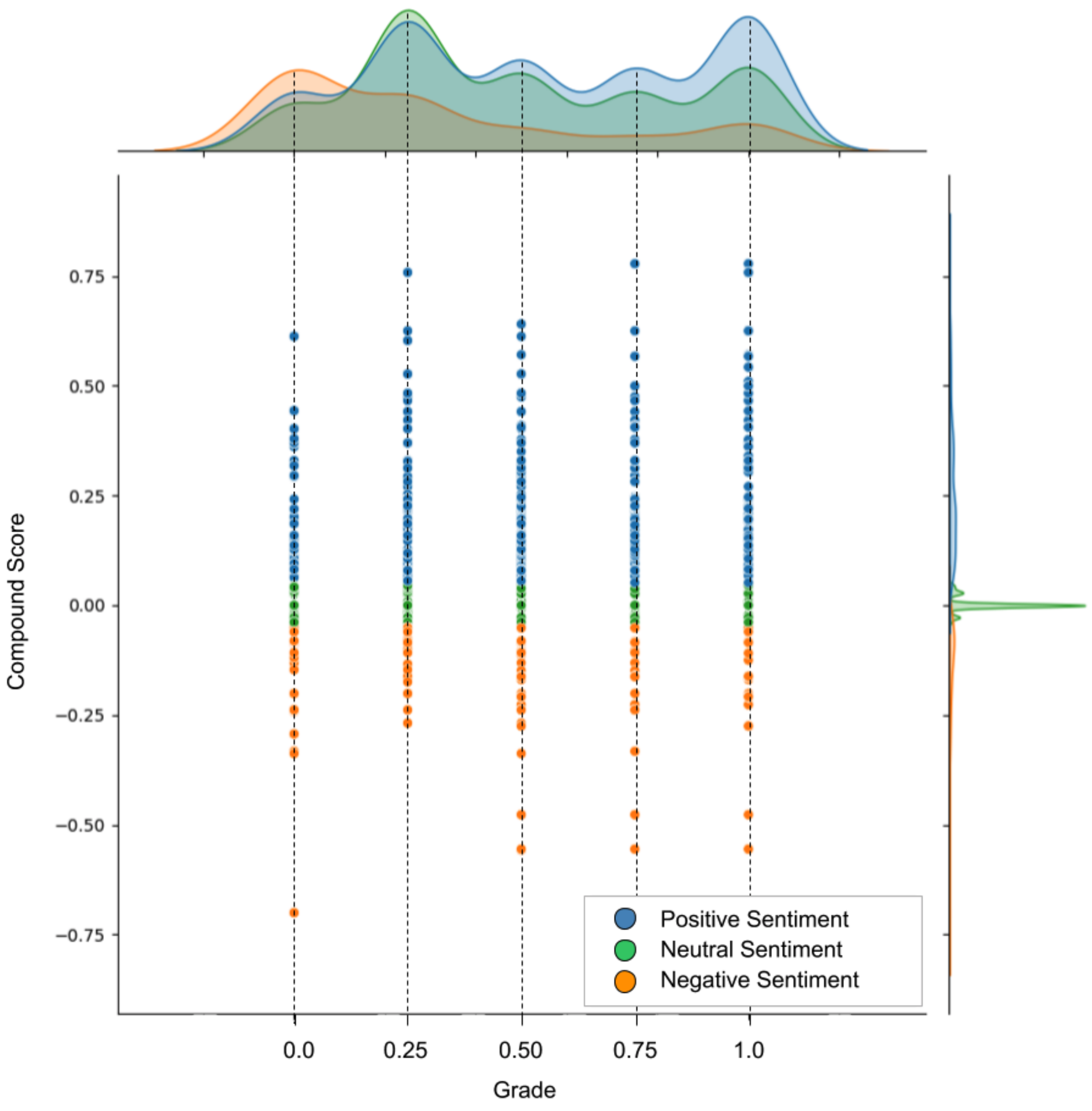


Figure 5.2: Exploratory analysis of the sentiment of teacher feedback across the grades provided

sentiment was still used often, once students earned at least a 0.50 grade teachers gravitated towards a more positive sentiment. The chasm between positive and neutral sentiment in feedback continued to widen as the grades increase. With the grades increasing past 0.50, the use of neutral sentiment leveled off and teachers shifted to more positive sentiment.

In the end, this exploratory analysis of the sentiment within teacher feedback proved to be fruitful. Mainly, we were able to identify trends in the amount of each sentiment used as it pertained to grades. Grades showed evidence of impacting the type of sentiment teachers used within their feedback to students. Similarly, it was clear that certain teachers veered away from negative sentiment, while others tended to stay more neutral or positive. From all this one thing is clear, to properly develop a tool that diversifies and automatically suggest quality feedback for teachers to use, variation in the sentiment used should be accounted for. Figure 5.1 and Figure 5.2 assist in visualizing this.

5.5 Predictive Modeling

To be a useful addition to a comment suggesting tool within ITS, a model needs to developed which can predict what type of sentiment a teacher would use while providing feedback to a student's open response mathematics answer. To that end, this research looks to develop this model using student answers and the compound sentiment scores calculated using VADER on our collected teacher feedback.

5.5.1 Methodology

As discussed earlier, with recent strides in NLP and deep learning, sentence embeddings have grown in popularity and trust amongst researchers. For this model, we utilize SBERT [RG19b] to generate a single vector representation of each student answer within our corpus. From this, each student answer was given a 768 dimensional vector representation that was used as the independent variables within our models to predict what category of sentiment a student's response would elicit from a teacher. We developed these models using 10 fold cross validation.

Data Preparation

Earlier in the paper, we discussed how the sentiment scores were developed for each teacher's feedback. Some examples were presented in Table 5.2. However, one more step needed to be included. Since every student answer would obtain feedback from all teachers in the subgroup, multiple teachers would generate a different compound score for the same student answer. It is because of this that all the student answers were grouped together and the average of the different teachers compound score were taken. This would then leave a dataset which consisted of a unique student answer per row and an average teacher compound score. A final step to preparing this data consisted of taking the newly acquired average compound score for each unique student answer and categorize the compound score into negative, neutral or positive sentiment. Just as we had previously discussed and performed. With this, each row in the data for training consisted of a student answer per row and the category of sentiment which the average of teachers would respond with.

Models Built

With the dataset in the correct form and vector representation for the student answers, various machine learning algorithms could be attempted. A wide spectrum of models were tested with varying degrees of flexibility. In the end, two different machine learning and one deep learning technique were applied to this data. The machine learning techniques consisted of the clustering algorithm K-Nearest Neighbors (KNN) and the tree based method Random Forest. While these methods have been around for years, their increased flexibility and depth allows them to perform well within this study. A slightly deeper model was attempted as well, a multilayer perceptron (MLP). The structure of this model is that of a simple feed forward neural network.

To reiterate, these models will take the SBERT vector representations of the student answers and predict which of the sentiment categories a feedback message should contain. With that goal in mind, we developed 3 different types of models, one model will attempt to predict all 3 categories of sentiment, one will attempt to predict whether the feedback should use negative sentiment or not, and finally a model which attempts to predict whether the the type of feedback should contain more of a positive sentiment. This will provide a strong summary of our ability to predict the sentiment and develop an algorithm which could be used in supplement to an automatic feedback suggesting tool within an ITS.

5.5.2 Results

Overall, Table 5.4 shows that we can accurately predict what sentiment a student answer should receive in its feedback. All of our models, while using SBERT vector representations, managed to obtain an AUC well above chance. At their worst, our models still managed to predict which class of sentiment with an AUC of 0.7306. While all the models performed well, we were best at identifying whether or not a negative sentiment should be utilized in a teachers feedback given a student's answer. When attempting to predict whether or not negative sentiment should be used, we obtained our highest performance garnering and AUC between 0.78-0.79. Even though the negative sentiment was easier to predict, being able to

Table 5.4: Performance predicting sentiment score of teacher feedback given a student open response answer in mathematics with 10-fold cross validation

Model	**Dependent Variable**	**AUC**
Random Forest	Negative, Neutral, Positive Sentiment Category	0.7473
KNN	Negative, Neutral, Positive Sentiment Category	0.7504
MLP	Negative, Neutral, Positive Sentiment Category	0.7306
Random Forest	**Negative Sentiment or Not**	**0.7798**
KNN	**Negative Sentiment or Not**	**0.7796**
MLP	**Negative Sentiment or Not**	**0.7896**
Random Forest	Positive Sentiment or Not	0 .7446
KNN	Positive Sentiment or Not	0.7359
MLP	Positive Sentiment or Not	0.7381

predict whether positive sentiment should be used or not proved to be slightly more difficult with its best AUC at 0.7446. It makes sense then that while we do well predicting negative sentiment for teacher feedback, but perform slightly worse identifying positive sentiment; when we attempt to predict whether the answer should elicit positive, neutral or negative sentiment together, the AUC falls right in the middle with the KNN performing best with an AUC of 0.7504.

5.6 Discussion

With the adoption of more automation within ITS increasing year by year, open response questions have fallen behind in their support of these automated tools. Well defined problems benefit most from the automation tools of ITS. To be able to bridge the gap between the benefits of the automation of well defined questions, this paper looked to explore teacher sentiment and develop a predictive model which could help predict the level of sentiment a feedback could require for a student's answer. The ability to accurately predict the level of sentiment is a step in the direction of understanding what types of feedback should be suggested to teachers in an automated fashion.

Similarly, the exploratory analysis revealed that teachers do in fact vary in the level of sentiment and amount of times they use each sentiment. There were those teachers who used more negative sentiment in their feedback, while other tended to use more positive sentiment within their feedback to students. Additionally, this study also showed that grades did in fact show a tend in the way sentiment was used amongst teachers. Namely, students which received a grade of 0 had the most negative sentiment within their feedback. As the grades improved, the negative sentiment subsided, decreasing almost linearly across the rest of the grades. Inversely, teachers provided feedback with more positive sentiment as the students performed better.

5.7 Future Work

We wish to continue to train more models in predicting which sentiment a teacher would use in their feedback to students. Of our models built, we acknowledge that none of the approaches are conducive to learning any sequential or contextual aspects of the student answers when training the models. In future work we wish to expand our predictive modeling to include word embeddings and sequential deep learning algorithms such as a Long Short Term Memory (LSTM).

We hope to use these predictive models, and the discoveries from the exploratory sentiment analysis, as supplemental tools for automatically suggesting feedback to teachers within an ITS. To be able to suggest effective, impactful and useful feedback automatically will require us to use these models. This will help to diversify the feedback suggested and more accurately be in tuned to what a teacher would normally want to suggest. If we can use this sentiment analysis as a way to suggest more diverse and accurate feedback, the more teachers will utilize the tool.

Chapter 6

NITELITE: Nonsynchronous Integrated Technology Environment - Learning from Interdependent Terminologies and Explanations

Abstract

Most support for students within online educational technologies (i.e. intelligent tutoring systems), such as hints, common wrong answer messages, or scaffolding problems; have been limited to questions with structured answers. More specifically, questions such as multiple choice or fill in the blank. With the use of natural language processing, this research aims to provide support for students working through open response questions in the form of a tool called NITELITE (Nonsynchronous Integrated Technology Environment - Learning from Interdependent Terminologies and Explanations).

Open response questions require students to elaborate their work and steps taken. A common practice in a classroom would be to get into groups and discuss your answer, steps and reasoning. This tool sets out to utilize collaboration as a support for students working through open response questions within intelligent tutoring systems. Collaborations provide students with the opportunity to interact with their peers, discuss approaches, and share their process taken. This interaction has shown to promote more critical thinking for students as compared to working individually. When presented with other student's rationales and approaches, students are required to validate their work and be confident enough to support their work over others. However, collaborations are becoming tougher and tougher to support as learning becomes more and more asynchronous. While there are online collaborative tools, the process is synchronous; requiring students to be active at the same time. From this, NITELITE was developed to wed the benefits of collaboration with the continued advancement towards more asynchronous learning. More specifically, in this research NITELITE is developed, utilizing natural language processing methods discussed in previous chapters, to provide an asynchronous collaborative tool to teachers and students. This research presents how students successfully utilized asynchronous collaboration within NITELITE. Additionally, this study successfully completed a pilot study of NITELITE with randomized control trial which provided deeper insights into the student's interaction with such a tool.

6.1 Introduction

Recently, there has been a move to more online learning and online educational technologies are being integrated more and more into every day schooling. From this, learning has become inherently asynchronous. Students are required to utilize these systems to complete homework and move forward in class. While these systems, such as intelligent tutoring systems (ITS), have a plethora of support for both teachers and students alike, this dissertation has attempted to address a major pitfall with the support; most of the support is built for questions with structured answers, such as multiple choice or fill in the blank. Questions with well defined answers. Up until this chapter, this dissertation has addressed the major hindrance by utilizing natural language processing to develop automated scoring and an approach to identifying similar student answers (for suggesting automated feedback to students working through open response questions). While those benefit both students and teachers, they are primarily support for teachers.

Open response questions provide the opportunity for students to clearly identify their process, the steps taken and justify their work. Teachers can gain a deeper understanding of the student's comprehension of the materials on a more personal level and can tailor their help for that particular student. Similarly, teachers can clearly see at what point a student has begun to misinterpret or misunderstand a problem or theory. This is a major advantage open response questions have over questions with structured answers, especially in an area such as mathematics. More specifically, there are many different ways to approach a question in mathematics. In a question with a structured answer, such as multiple choice, a teacher can only infer what the student did to arrive at the correct/wrong answer based on correlated assumptions. However, given that there are multiple correct ways to answer most math questions, its difficult for a teacher to pinpoint what the student misinterpreted or what step they misunderstood. This is where open response questions thrive. Again, earlier chapters address developing automated scoring and a method for suggesting feedback automatically to help teachers diversify their content. However, online educational technologies look to support both teachers and students, alike.

So the question becomes, how can support be developed for students with open response questions within online education technologies in an increasingly online learning environment?

Online educational technologies, such as intelligent tutoring systems, that support open response questions do not provide the same support students receive from questions such as multiple choice or fill in the blank. ITS can simply have a set of rules in place to return hints or feedback messages given a student's choice. However, students progressing through an open response question lack this support. This final pilot study aims to utilize natural language processing and open response questions to develop a tool deemed NITELITE (Nonsynchronous Integrated Technology Environment - Learning from Interdependent Terminologies and Explanations). This tool sets out to support students by utilizing the major aspects of collaboration but in an asynchronous fashion. Students using NITELITE will be able to see other student's answers to the same open response question and can have the option to go back an edit their answer after. This simulated collaboration will allow the student to see other student's perspectives and reasoning, requiring the students to process their work and justify why their work is either better not as accurate as the other rationales. To pilot this tool, I ran a randomized controlled trial (RCT) with middle school students in 7th grade mathematics. This research set out to analyze student interactions with NITELITE and answer:

1. Did students who utilized NITELITE perform significantly better in the post test, did learning occur?

2. For those who can edit, was there a reliably higher score in the final submitted answer?

3. If students are provided the opportunity to edit their work, is there a reliable

difference in the answer length across the two conditions?

4. Whether or not the asynchronous collaboration's, the standard NITELITE or random rank list condition, effect on the difference in answer length (between final and initial answers) is significantly different from 0.

5. When presented the opportunity, do students edit their answers after asynchronous collaboration? Whether or not, if given the chance to edit, there is a reliable difference in whether a student edits or not by condition when controlling for the initial score.

6. If there is any evidence to show that students are more or less likely to rank their answer in the top half over the bottom half of the rankings.

6.2 Background

6.2.1 Intelligent Tutoring Systems

For years, intelligent tutoring systems have been deployed and utilized by educators [ABR85] [Nwa90][CKA97] and these have provided teachers and educators with automatic grading of student answers, immediate feedback to students and automatically generated reports, just to name a few. More specifically, [Ric88] breaks down 3 criteria a system must meet to be considered an intelligent: the system must be able to automatically solve, evaluate and infer information from problems within its domain; it must be able to interpret and evaluate the users understanding of the material at hand; and lastly, the tutor within these systems must be intelligent in its way of providing assistance to students for improved student learning. A study from [Van11] discussed the common positive effects ITS have on student learning; that these systems could potentially be close to as effective as human tutors. Systems such as ASSISTments, McGraw Hill's ALEKS^{TM} and/or Carnegie Learning's Cognitive Tutor^{TM} meet those definitions of an ITS. A few ITS, such as ASSISTments, have even begun to accept open education resources (OER) content like EngageNY. However, as has been discussed in previous chapters of this dissertation, most of the intelligence is limited to questions and problems with structured answers.

Most of the tools and automation that make online educational technologies 'intelligent' are for structured problems with easily defined answers. These consist of questions that are, for instance, multiple choice or fill in the blank. There are many advantages to structured answers, such as [WB05], however, a question with structured answers, such as multiple choice, can inherently mislead teachers. When students guess right, it can leave the student with a false sense of understanding the materials and the teacher with a false sense that the student understood the answer. Studies such as [BR08], do note that this is why intentionally wrong answers are put into these multiple choice questions, to lower the likelihood of guessing correctly. However, this can also be dangerous; [RIM05] showed when students are exposed to the incorrect answers, they can latch on to that incorrect answer and think they're right. Creating a false sense of security amongst students.

6.2.2 Open Responses

Work has been done to show that there is a benefit to providing teachers with a variety of problem types; both structured questions and less structured open response questions [Ku09]. Similarly, [Mar99] and [OBKM13] discussed how questions with structured (multiple choice) and less structured (open response) answers elicit differing levels of cognition. [BS06] agrees with this, noting that students generating their own rationale and answers, while connecting them to economic theory, is imperative to their learning. Open response questions essentially are a median through which a student is able to explain, reason and justify their work or answer. Thus, providing deeper insight into the students process of thinking. Open response questions, and the elaboration from students, allows teachers a more personal understanding of students and their work. More specifically, open response questions allow teachers to not rely on historically correlated mistakes, but the actually steps written by the students. Again, this isn't to say multiple choice or fill in the blank questions do not have their place. However, certain subjects can utilize multiple approaches to arrive at a final answer. Case and point, Mathematics (the subject used within this study and all previous chapter's studies).

Mathematics has always been, at its core, a subject which benefits from open responses. Mainly, there are various steps, and combinations of those steps, that can be taken to solve problems. There are more efficient routes students can take to arrive at answers, but other routes work as well. Within those steps and routes taken, a mistake can be made a many different points. Given that students approach questions in different ways, its difficult for a teacher to infer where the student specifically went wrong from a multiple choice, or fill in the blank, answer. Open response questions can provide this insight. However, as discussed previously, within online learning technologies, support for those question types are limited. While few support open responses, the few that do, fail to live up to the 'intelligent' aspect of intelligent systems. Previous chapters in this dissertation helped to address this by developing automated scoring and feedback to students. Those chapters have assisted in the development of support for teachers in providing open response questions and helping teachers diversify the content which they provide students. This, however, is only addressing support for teachers. As mentioned earlier, students are supported working through questions with structured answers in the form of hints, common wrong answer messages, and or scaffolding problems.

This pilot study looks to develop this support by adopting collaboration into online learning technologies, such as intelligent tutoring systems.

6.2.3 Collaborative Learning

When students answer open response questions, they are elaborating their steps, their process of thinking and explaining why their answer is correct. They are essentially embracing an aspect of collaboration. In education, collaborative learning is commonly defined as students working together in groups of two or more to learn; communicating with others, discussing how they would approach or answer the problem, seeing how others may be interpreting a questions, and seeing whether or not the way they are thinking is similar to that of the other students. In both collaboration and individual open responses, students are detailing their steps taken and explaining their reasoning (which is one of the main mechanisms that drives collaborative learning) [Dil99].

Some argue the difficulties and challenges of collaborative learning, such as free-riding (case where some students did most of the work and others did much less) and opinions of students who are considered 'low-competence status' not being valued [LJW18]. Similarly, studies such as [RAS17] discussed how both worked examples and collaboration are considered effective in supplementing learning. However, in their study, they were unable to show that collaborative learning improved comprehension over independent learning within more complex problems.

Studies on collaborative learning have been around for years [Gok95] and have shown the benefits students experience when working in groups. In fact, [Gok95] showed that students who collaborated out performed those who worked individually on a critical thinking exam. Its not just that the other perspective can help the students learn, but it requires them to think critically through the problem and decipher which approach and which answer is indeed the best. [LG12] surmised that, academically, collaborative learning requires students to think more critically and improves critical thinking skills, engages students more directly in the learning process, improves classroom results, and can bring to light the student's problem solving technique. Another study explored the effect of collaborative learning had on undergrad engineering students and reported learning gains[TCC+01]. The students who experienced collaboration reported statistically significant higher learning than those who didn't[TCC+01].[ML07] showed that when students collaborated, they managed to go into greater depth and expand their reasoning and argument within the selected topic. As [CL14] notes, group work, if it is a problem which requires the student to think conceptually (not just applying) and the student has the skill set to perform the task at hand, will elicit learning for students when they talk, interact and contribute to the discussions with others in the group. Essentially showing that collaboration enlightened other students to contrasting perspectives and impacted their learning. It is clear there is support for collaboration in the classroom. However, in recent years, there has been a move to more online learning and asynchronous styles of learning.

6.2.4 Asynchronous Collaboration and OER

While collaboration in classrooms has its benefits, recently, there has been a move more to online learning (even more, recently, due to COVID-19). Online educational technologies, such as ITS, are becoming more integral to the education of students, and many schools have begun utilizing online learning in a myriad of new ways. A result of this is learning has become more asynchronous. Many ITS do a good job enabling the interactions between instructors and students asynchronously. As discussed previously, these systems provide support to both students and teachers, alike. Teachers are able to get automated student reports, provide automated scoring to students (for both open response questions and questions with structured answers, as Chapter 2 presented) and students are able to read the automated feedback from the systems in the form of immediate automated scoring, hints and common wrong answer messages. Other works in this dissertation, such as Chapter 2 and 4, have attempted to help build this support for a more diverse set of questions by utilizing NLP.

It is the case that collaboration has been researched and used within ITS in many studies [TRM10], [OBAR14], [OBA$^+$14], [DWRK10], [WRK14], just to name a few. However, these studies require students to perform tasks live to create the collaborative aspect of the tool. While this indeed is an effective tool, in today's learning, where online learning is ever increasing, its very difficult to set up, schedule and follow students using a synchronous collaborative tool. Studies, like [OBA$^+$14], require students to be on the computer at the same time and communicating via audio as they work through problems. Other systems were developed with people communicating live via chat while they worked through problems, such as [WRK14] and their Adaptive Peer Tutoring Assistant. To be able to set this up for collaboration when classes are asynchronous, and most learning is online, is quite a difficult task. Therefore, an asynchronous collaboration tool would provide students with many of the benefits of collaboration while mitigating the extra steps to become synchronous. Namely, students on their own time, and whenever it fits their schedule, can have a collaborative experience.

While collaboration has been shown to be effective, there is an aspect of collaboration which has been shown to be a negative. Free riders[LJW18][PBB$^+$12]. In the simplest form, free riders are the students who do not contribute. These students just sit by and let others do the work, thus getting credit. It can then be perceived that the group understood and learned, but its the case only certain students learned. The free rider just waited until the right answer was agreed on and used that information without learning. NITELITE attempts to mitigate this by having students rank rationales and answers which are presented to them (this is discussed in following sections).

An asynchronous collaborative tool could wed the benefits of face to face collaboration and the capabilities of ITS. More specifically, by utilizing open response questions, student's will engage in a major mechanism of collaboration by being presented other student's rationales and answers to open response questions (will detail in following sections). One system, myDALITE[CLB+19], has provided an asynchronous peer instruction. In fact, this research was inspired by that work. I wish to enable asynchronous collaboration within an ITS that supports open response OER content with my tool NITELITE.

OER content has provided teachers with a source of content which is freely available for use as their teaching sees fit and meets the core standards. As mentioned earlier, ASSISTments is one of the very few systems who have adopted the freely available OER content. NITELITE looks to utilize this OER content and since ASSISTments [HH14] [RFMM16] does not offer a tool which enables asynchronous collaboration and freely supports OER content, it is utilized as the system which NITELITE is built upon. This research aims to explore if the benefits of collaboration can be transferred to learning within ITS and in an asynchronous way while using OER content.

ASSISTments has the foundational support to develop such an asynchronous collaboration tool. Mainly, this system supports open responses question types; allowing teachers to give out questions which require students to rationalize their answer and provide descriptions for why and how they arrived at their answer. As mentioned earlier, open response questions can elicit aspects and mechanisms of collaboration such as explaining your reasoning and others explaining theirs.

An unique aspect to an asynchronous collaboration tool, like NITELITE sets out to be,is that it can provide the opportunity to identify the how confident a student is in their work. By examining the way a student performs and behaves when presented with another student's rationale, inferences can be made based on how long the student studies the other answers and whether their rationale changes for better or worse (in terms of predicted grade, details in following sections). For instance, maybe a student is confident in their answer, but the answer is a 30% score. So when a student is presented another students rationale (which is correct) from asynchronous collaboration and they refuse to change their answer, this could provide greater insight into the student's capabilities, confidence and comprehension. This insight into confidence, which has similarly been explored with multiple choice questions in studies such as [CLBD13], could be very insightful to a teacher and provide additional information about the students comprehension of the topic they're answering. However, with that study, the inferences could be made only by students labeling their confidence when they selected an answer. While this is valid, NITELITE and its utilization of open responses allows for this inference to be made by exploratory analysis while also having the students ranking the other answers seen (in greater detail in following sections). This automation of identifying the confidence is done by utilizing natural language processing to automatically assess the student's initial open response answer.

6.2.5 Natural Language Processing for Asynchronous Collaboration

Toward developing NITELITE to support asynchronous collaboration, and exploring whether collaboration impacts students performance within ITS, there are multiple natural language processing (NLP) approaches which can be utilized for generating numerical representations of the student's answers. This ranges from using more traditional numerical representations of words, such as generating term frequency inverse document frequency (tf-idf)[R$^+$03] (which has been used in past research such as [TMD14]), to more advanced deep learning methods such as utilizing word embeddings such as GloVe[PSM14], word2vec[MCCD13] or BERT[DCLT18] which stands for Bidirection Encoder Representations from Transformers. While utilizing deep learning and word embeddings are enticing, bias still needs to be considered because studies such as [BCZ$^+$16] showed that there are bias built into the pre-trained versions of these embeddings. While many can develop their own embeddings, most datasets wont match the robustness of GloVe being trained on Wikipedia or word2vec being trained on Google News. Thus, this study utilizes pre-trained embeddings, more specifically, pre-trained sentence level embeddings.

While word level numerical representations have been common in NLP research, recent research has explored sentence level embeddings/representations. In the simplest terms, this approach is similar to utilizing deep learning to develop a word embedding, except instead of generating a vector for each individual word, a vector space is developed where each point in the vector space is a vector representation of a sentence. This study utilizes these sentence level embeddings within its architecture. Mainly, SBERT[RG19b] is used to generate student answer level embeddings. A single embedding, as shown in Chapter 4, can provide an avenue toward comparing how similar groups of text are; additionally, this relationship can help to generate accurate prediction power as well[BBE+21]. It should be noted that, as mentioned above, there is bias and unfairness built into pre-trained word embeddings. However, Chapter 3, presented that when using SBERT in a predictive manner there was very low unfairness in the model's prediction. Given those results, SBERT was finalized as the main NLP approach utilized within NITELITE. Other sentence approaches are available and were testing in Chapter 4, such as the the Universal Sentence Encoder[CYK+18]. More recently, work has been done to develop a single vector representation of full documents and not just sentence level embeddings. This approach, referred to as doc2vec [LM14], can generate a single embeddings for an entire document. It is the case that more verbose documents could benefit from such an approach, but for this pilot study, mathematics is the content for the questions and most answers range from a couple words to a couple sentences at most. For this reason, a sentence level embedding, such as SBERT, was sufficient.

6.3 The Software, NITELITE

6.3.1 NITELITE Overview

Collaboration has been shown to be an effective way to supplement learning for students. In the most general sense, collaboration will force students to discuss and reason with why they answered they way they did. Student are then presented perspectives from other students in the collaboration, and they must work together to decide what is right. This requires the student to not only be confident in their response and answer, but be able to support it and trust it when presented with other answers and rationales. As noted earlier, this is one of the mechanisms that drives collaborative learning. The collaborative aspect comes with the seeing others explanations and rationales and considering their perspectives to the questions [Dil99]. While there are some tools developed to attempt this asynchronously, such as myDALITE [CLB+19], most systems have adopted collaborative tools which require students to perform tasks synchronously. This research sets out to explore how the benefits of collaboration, and the language used by students, transfer to an ITS with OER content asynchronously. To explore this further, this research generates the asynchronous collaboration tool with open response question within the ITS ASSISTments, called NITELITE.

The concept of collaboration within online educational technologies has been attempted before, as mentioned earlier. However, to reiterate, those approaches require students or teachers to be involved synchronously. This includes requiring student users to be on the systems as the same time. For NITELITE, this is not a requirement for the tool to be implemented. The asynchronous aspect to NITELITE embodies collaboration by presenting the student with various student answers and rationales to the same problem. By utilizing this central theme of collaboration, I can artificially generate a collaborative experience for the student. The collaboration is artificial in the sense that everything is asynchronous, not live. Similar to the research discussed earlier, students who work in groups supplement learning from talking, interacting and contributing to discussions with others within the group[CL14]. For NITELITE, since its asynchronous, students won't talk or discuss face to face, but NITELITE will present what other students have argued or reasoned for their answer. The way these are chosen are discussed in subsequent sections.

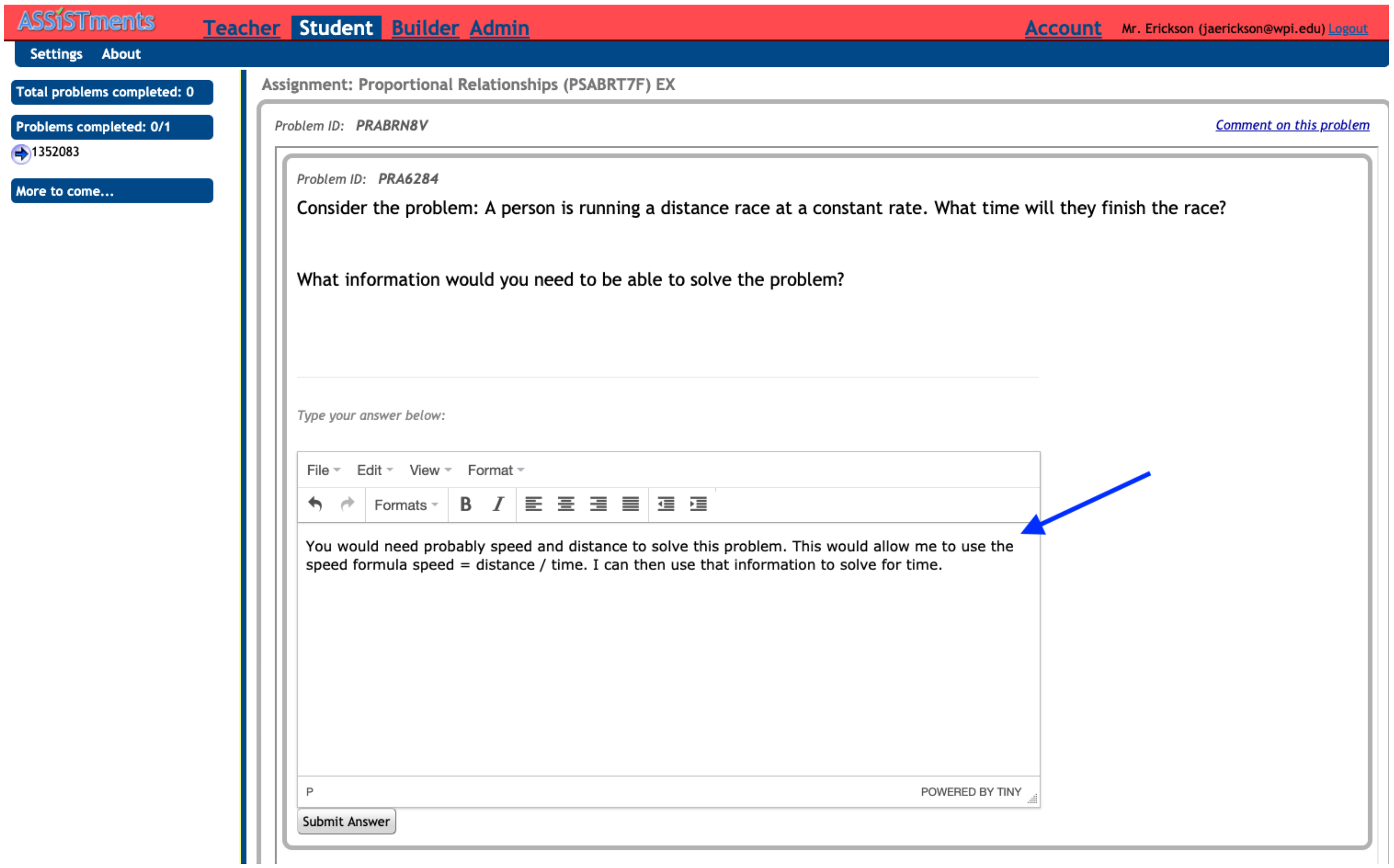
Another task which was integrated into NITELITE was asking students to rank the order in which they feel the answers are of the best quality. This ranking task was chosen for the potential ability to identify if students see their work as correct when in fact they are incorrect. Essentially, this task asks the student: if given other students perspectives and approaches to a question, do you still fee as though your answer is better? This task can help to identify if the student is confident, but incorrect. This finding would provide teachers with deeper insight into the student's understanding of the material. Essentially, a teacher can then characterize incorrect answers into two broad categories: Confident in their process and wrong, fully incorrect. Each will require different interventions by teachers, but being able to identify this is powerful for teachers.

Once all the tasks are finished the student is then allowed to either edit their original answer or submit their original answer. This then completes the tasks for NITELITE. Students then can progress through the rest of a problem set which NITELITE is associated with.

6.3.2 NITELITE Design

As lightly discussed previously, there are up to 3 steps within NITELITE. A student starts with an open response questions (for this study, an OER Open Up question), as shown in Figure 6.1, and is required to type their answer below in the box provided (blue arrow in Figure 6.1). This gives the student the chance to elaborate their understanding of question and what their reasoning is for their answer.

After an answer is entered by the student, a student is then prompted with a message (as shown in Figure 6.2) which states 'You are about to see some answers from other students along with the answer you have just submitted. Please click-and-drag these answers to order them based on how you believe your teacher would score them.'. NITELITE prompts the student that they will not only be able to see theirs and other student's answers, but they are also asked to order the answers based on how the teachers would score them. Again, this is with the hopes that



Assignment: Proportional Relationships (PSABRT7F) EX

Problem ID: PRABRN8V

Problem ID: PRA6284

Consider the problem: A person is running a distance race at a constant rate. What time will they finish the race?

What information would you need to be able to solve the problem?

Type your answer below:

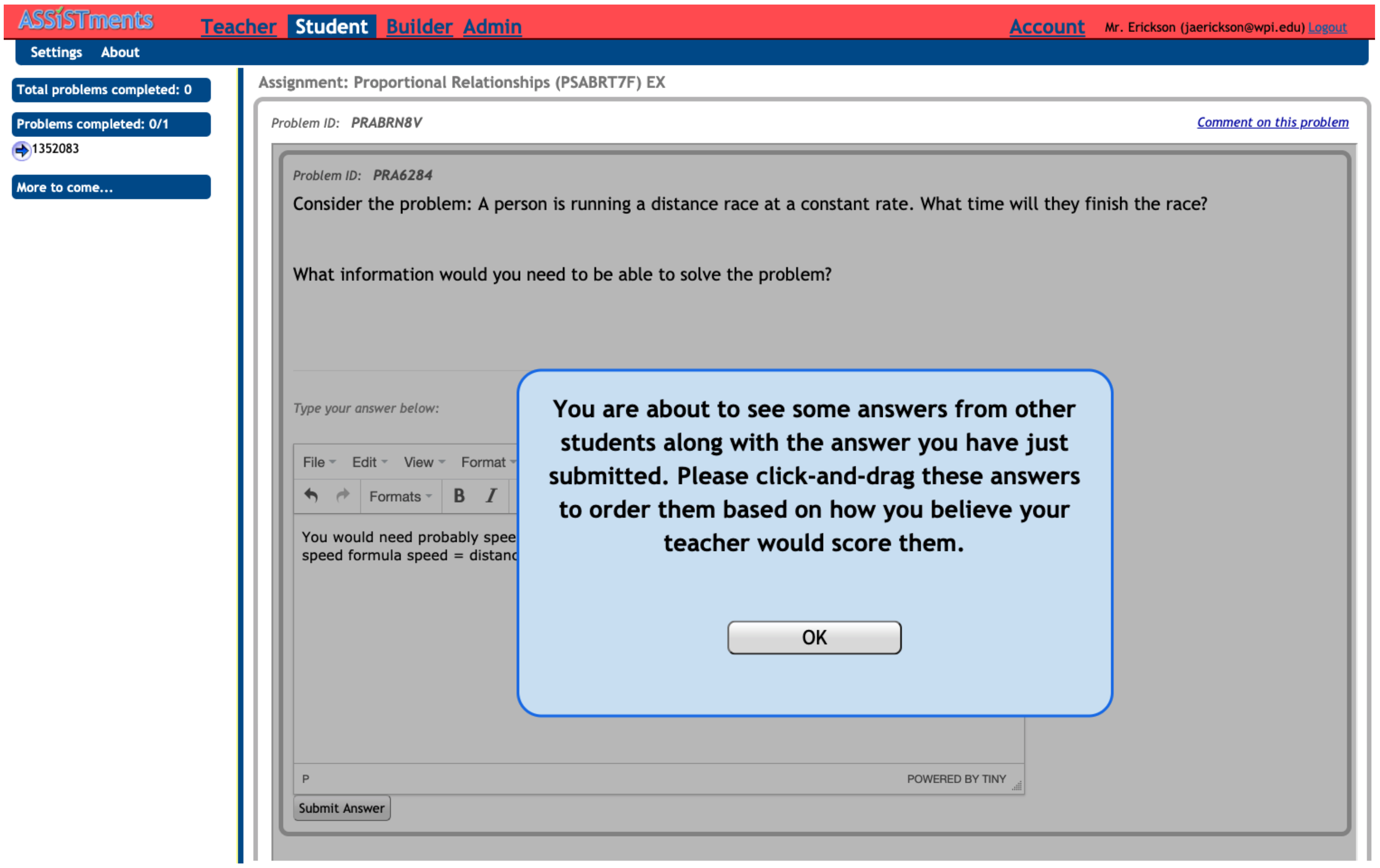
You would need probably speed and distance to solve this problem. This would allow me to use the speed formula speed = distance / time. I can then use that information to solve for time.

Figure 6.1: Initial screen of NITELITE and example answer

inferences can be made into the student's understanding. If the student's original answer is wrong and they rank their answer above other correct answers, this could be evidence that the student is confident in their incorrect work.

Once the student clicks 'OK', the student progresses within NITELITE to the asynchronous aspect of NITELITE. Once again, I present the student with a message of what the task is at hand in green and shown in Figure 6.3. This was put in place to make certain students had a clear understanding of what is required. From Figure 6.3 there is a column which articulates where the 'Highest Score', 'Second Highest Score', 'Second Lowest Score' and 'Lowest Score' should be placed. Student then could click and drag the answers (by clicking where the blue arrow is pointing) to the final order they would expect. Once the student is satisfied with their ranking, they could click the 'Submit Ranking' button (green arrow is pointing to the button in Figure 6.3).

The list, which the student is given, is chosen utilizing multiple NLP approaches to identify both similar and dissimilar student answers to their own. However, the first row ('Highest Score' row in Figure 6.3) is always populated with the student's original answer. The following 3 rows utilize the modern natural language processing. A mentioned previously, and utilized effectively in Chapter 4, SBERT is an approach which generates a single vector representation of groups of text. Instead of a single embedding per word, SBERT can generate a single embedding for each student answer. So once a student submits an answer, NITELITE instantly generates a single SBERT embedding vector representation for said answer. Once the embedding is generated, whichever student answer embedding vectors are near the student's original answer in the vector space would be considered more similar to it. This is the foundation of the steps taken to populate the rest of the rows within NITELITE.



Assignment: Proportional Relationships (PSABRT7F) EX

Problem ID: PRABRN8V

Problem ID: PRA6284

Consider the problem: A person is running a distance race at a constant rate. What time will they finish the race?

What information would you need to be able to solve the problem?

Type your answer below:

You would need probably spee
speed formula speed = distanc

You are about to see some answers from other students along with the answer you have just submitted. Please click-and-drag these answers to order them based on how you believe your teacher would score them.

Figure 6.2: NITELITE notification of ranking task

Since the goal of NITELITE isn't only to share similar answers to the student's (as mentioned previously, being able see other student's perspectives to the question will cause the student to justify why their answer is correct or the other student's answer is correct), I populate the rows with both similar and dissimilar answers to the student's original answer. The second row ('Second Highest Score in Figure 6.3) first aims to provide the student with another similar student answer. By generating the SBERT embedding vector, NITELITE can identify the closest answer (with a score greater than 0-for quality answer control). NITELITE is then able to calculate the canberra distance between all the SBERT embedding vectors of student answers to the same problem. Then whichever previous student's answer vector has the smallest canberra distance to the new student's answer populates row 2 (Second Highest Score) in Figure 6.3.

While having similar and dissimilar answers is at the core of NITELITE, I decided to have an exemplary row for students. A row which contains another student's answer which received 100%. Therefore, there is always an answer which will promote learning for the student. The choice is simply made by randomly selecting a previous student's answer which received a 4/4. This randomly selected exemplary answer populates the 3rd row ('Second Lowest Score' in Figure 6.3).

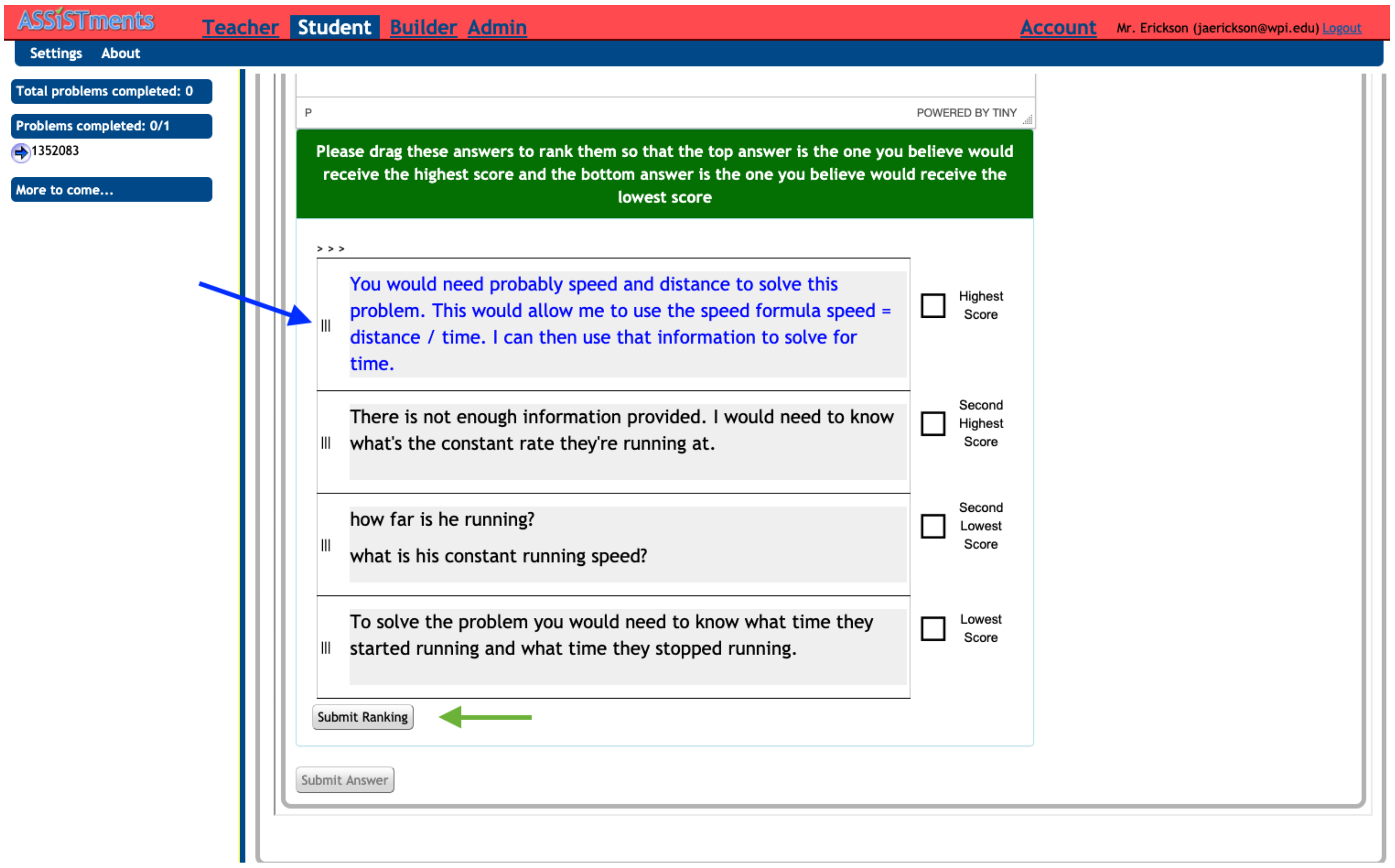
For final row, 'Lowest Score' row in Figure 6.3, NITELITE fills the row with an answer which was "different" from the original answer with a score greater than the predicted score. Essentially, there are two steps to populating the final row. First NITELITE utilizes SBERT, again shown to be accurate at scoring student answers [BBE+21], as a predictive model to predict a score for the original answer the student submitted. Then NITELITE utilizes the SBERT embedding of the student's original answer to identify a different, opposite of row 1, student answer to the problem. This is performed by identifying which other student answer embedding vectors are more than 1 standard deviation (utilizing canberra distances) away from the submitting student's original answer. So NITELITE then randomly selects a student's answer which is a score greater than the predicted score and more than 1 standard deviation away from the submitting student's original answer.

What should also be noted in Figure 6.3 the 'Submit Answer' button is not able to be clicked. This was set in place to make sure students would understand that they aren't submitting their answer, yet. They are just submitting their ranking of the other answers. Once they click the 'Submit Ranking', they are given one last prompt as shown in Figure 6.4. This final prompt simply gives the students two options, would they like to 'Edit Original Answer' or 'Submit Original Answer'. If the student chooses to submit their original answer, the task is over and it moves on to the next problem. If the student chooses to edit their original answer, they are sent back to the screen in Figure 6.1. From there, they may edit their answer and submit their new final answer. Once submitted, the student is finished with NITELITE.

6.4 NITELITE Randomized Controlled Trial

6.4.1 Randomized Controlled Trial Design

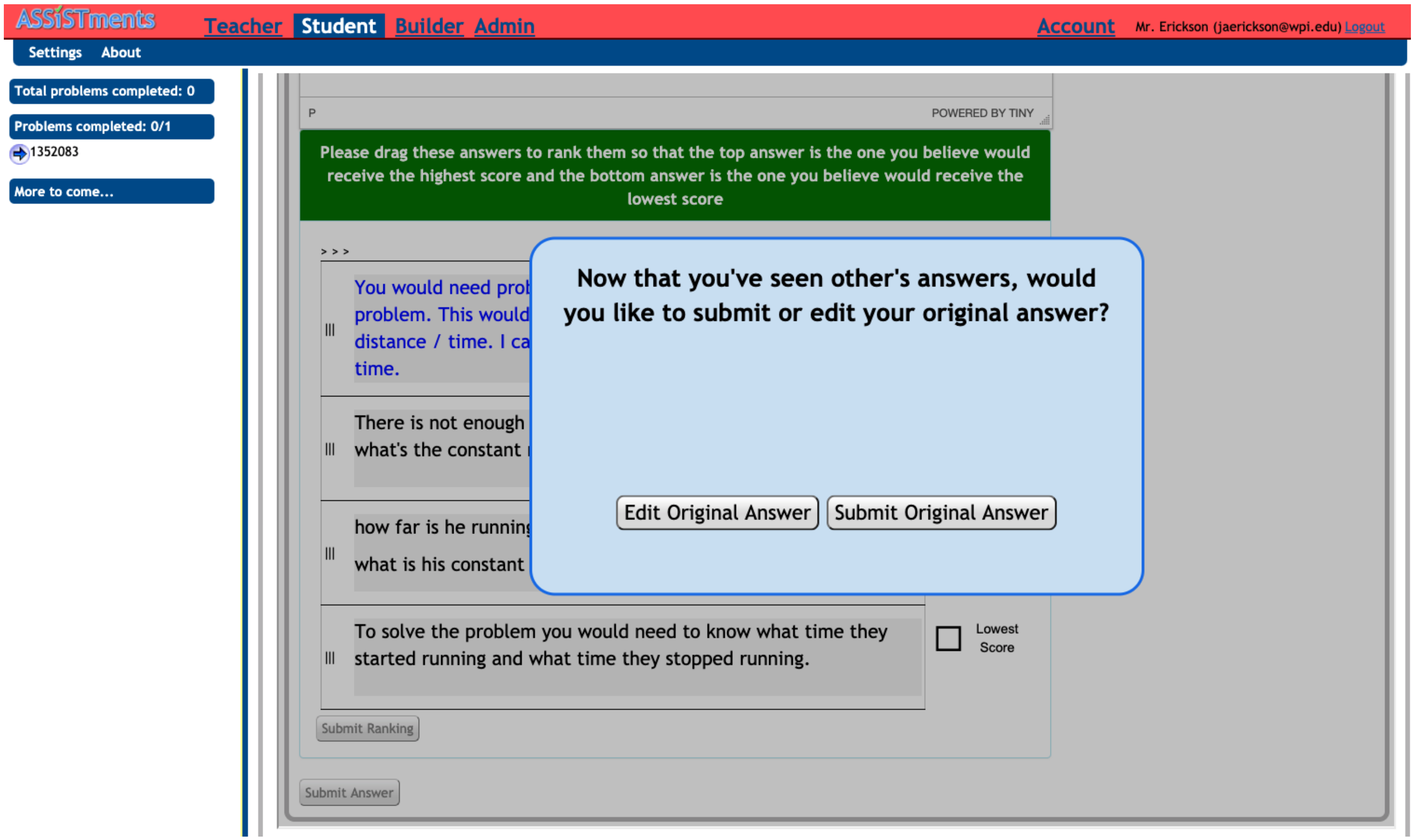
Along with developing NITELITE, this project attempted to run a pilot study. To accomplish this, I ran a RCT with 4 teachers and their 7th grade math students. This provided the opportunity to have students interact with NITELITE and apply it to their own work. Going in, there were a couple hypothesis I was hoping to explore. Mainly there were 7 questions: 1.) Did students who utilized NITELITE perform significantly better in the post test; did learning occur? 2.)For those who can edit, was there a reliably higher score in the final submitted answer? 3.)If



Please drag these answers to rank them so that the top answer is the one you believe would receive the highest score and the bottom answer is the one you believe would receive the lowest score

You would need probably speed and distance to solve this problem. This would allow me to use the speed formula speed = distance / time. I can then use that information to solve for time.	Highest Score
There is not enough information provided. I would need to know what's the constant rate they're running at.	Second Highest Score
how far is he running? what is his constant running speed?	Second Lowest Score
To solve the problem you would need to know what time they started running and what time they stopped running.	Lowest Score

Figure 6.3: NITELITE list of answers for the student to rank



Please drag these answers to rank them so that the top answer is the one you believe would receive the highest score and the bottom answer is the one you believe would receive the lowest score

> > >

You would need prob
problem. This would
distance / time. I ca
time.

There is not enough
what's the constant

how far is he running
what is his constant

To solve the problem you would need to know what time they started running and what time they stopped running.

Now that you've seen other's answers, would you like to submit or edit your original answer?

Figure 6.4: NITELITE final prompt to students

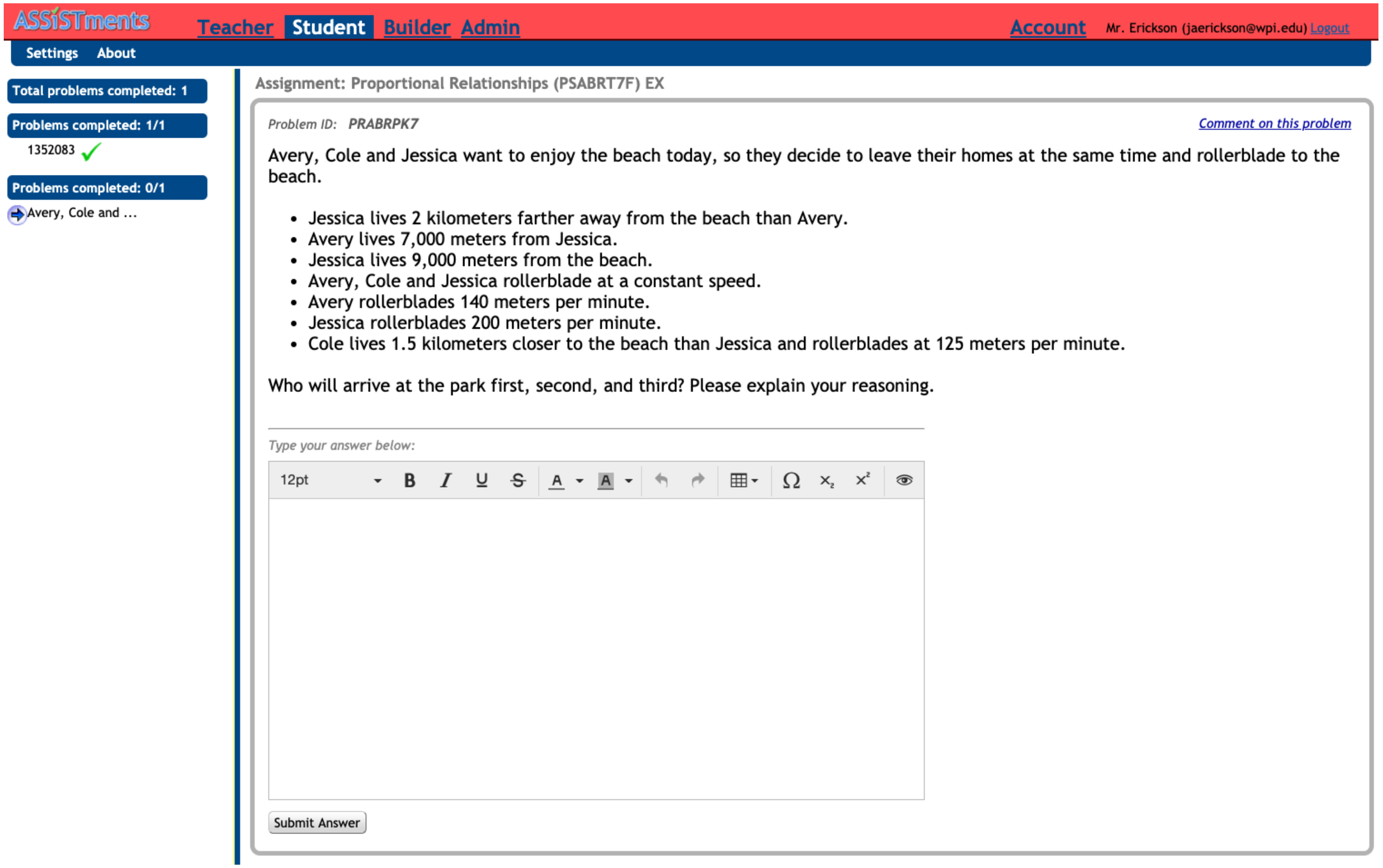
students are provided the opportunity to edit their work, is there a reliable difference in the answer length across the two conditions? 4.)Whether or not the asynchronous collaboration's, the standard NITELITE or random rank list condition, effect on the difference in answer length (between final and initial answers) is significantly different from 0. 5.)When presented the opportunity, do students edit their answers after asynchronous collaboration? Whether or not, if given the chance to edit, there is a reliable difference in whether a student edits or not by condition when controlling for the initial score. 6.)If there is any evidence to show that students are more or less likely to rank their answer in the top half over the bottom half of the rankings.

To answer these questions, the RCT consisted of 2 7th grade mathematics problems. The first would be deemed the pre-test and the following question would be the post-test. The problems selected for this study were chosen from OER content. More specifically, I tailored the questions to be closely related to work students had worked with not too long prior. At the time of the study, this consisted of solving problems about proportional relationships.

The questions chosen for the study needed to be not only OER content, but questions which the students hadn't already worked on up to this point. Likewise, it was important to find a pre-test question which had enough previous students, who are not in the RCT, that have answered and teachers have graded. It is clear from the prior section that NITELITE populates itself using historical student answers to the same problem. After scouring 7th grade OER content similar to the student's current work, the final problem selected for the pre-test was one which asked "Consider the problem: A person is running a distance race at a constant rate. What time will they finish the race? What information would you need to be able to solve the problem?". To solve this problem the student would need to understand how to utilize the speed formula to solve for time (i.e. speed = distance/time).

Within the pre-test question is where the conditions split. There are 3 separate conditions a student could be randomly placed into: Standard NITELITE condition, random ranking list condition, and the control condition. Every student will see the same initial screen, as shown by Figure 6.1. Every student will see the same OER constant rate question. However, once they click submit, their paths diverge. The standard NITELITE condition will progress through the problem as shown in the previous section from Figure 6.1 - Figure 6.4. The student will be prompted to rank the other student answers shared with them, then they are able to either submit or edit their original answer. From there, they advance on to the post-test question. The random ranking list condition looks identical to the NITELITE condition. However, instead of using NLP to populate the 3 rows (recall the first is always going to be the student's original answer in NITELITE; the random condition keeps this the same as well) the random rank list condition will populate the 3 rows with completely random student answers. This can help to justify whether or not the natural language processing is necessary to improve learning. Similar to NITELITE, once the student ranks all the answers from best to worst, they can either edit or submit their original answer. Once completed the student can progress on to the post test. As for the control condition, the student will progress through the open response pre-test in a traditional manner without and asynchronous collaborations. The student will just answer the open response business as usual and progress on to the post-test.

As for the post-test, since this study set out to see if there were any learning gains from utilizing NITELITE or a random set of student answers in an asynchronous fashion, I chose an OER inspired question. While the pre-test required students to understand the formula speed = distance/time, the post test required students to be able to now use the speed formula to calculate how long it would take Jessica, Avery



Assignment: Proportional Relationships (PSABRT7F) EX

Problem ID: PRABRPK7

Avery, Cole and Jessica want to enjoy the beach today, so they decide to leave their homes at the same time and rollerblade to the beach.

- Jessica lives 2 kilometers farther away from the beach than Avery.
- Avery lives 7,000 meters from Jessica.
- Jessica lives 9,000 meters from the beach.
- Avery, Cole and Jessica rollerblade at a constant speed.
- Avery rollerblades 140 meters per minute.
- Jessica rollerblades 200 meters per minute.
- Cole lives 1.5 kilometers closer to the beach than Jessica and rollerblades at 125 meters per minute.

Who will arrive at the park first, second, and third? Please explain your reasoning.

Figure 6.5: Randomized Controlled Trail: Post-test question

and Cole to travel to the beach. This is shown in Figure 6.5. Having a post-test which utilizes the formula from the pre-test, but requires students to apply it, can help me infer if any learning did occur. For instance, if a student did poorly on the pre-test, but managed to perform well on the post test (and were in the NITELITE condition), this could be evidence of learning occurring. This is why the post-test question was selected for this RCT.

In the end, I was able to recruit 4 teachers who taught middle school mathematics. More specifically, 7th grade mathematics. From this, I was able to get 193 students to participate in the RCT. With 69 residing in the standard condition, 68 in the random condition and 56 in the control condition. Teachers were given a week to assign the problem set I generated for the study. Teachers were then able to grade the answers to the pre-test and post-test question from students within the control condition directly within ASSISTments. However, the teachers were given a spreadsheet to grade the standard NITELITE condition student answers and the random rank list condition student answers to the pre-test question. This was because the teachers could have up to 2 answers per student. If the student did edit their original answer, the teacher would be given both the original and the edited answer to score. As for the post-test, the teachers were able to grade the random rank list and standard NITELITE condition student's post test answers within ASSISTments.

6.4.2 Results

As mentioned earlier, in the end I was able to recruit 4 7th grade mathematics teachers and their students to participate in my NITELITE RCT. In the end, 193 students opened the assignment and were assigned a condition. Figure 6.6 shows that 69 students were assigned to the "Standard" condition (which represents the

NITELITE condition), 68 students were assigned to the "Random" condition (mentioned earlier as the "Random Ranking List" condition) and 56 were a part of the Control condition.

What is apparent from Figure 6.6 is that there are a number of students who drop off and do not complete all the milestones within the RCT. There are 3 to 4 milestones a student is expected to complete within the RCT. Initially, every student who opens the problem set is then assigned a condition (top level of Figure 6.6). From there, if the student is assigned the standard or random condition, they will have 3 more milestones to complete. Mainly, they submit an initial answer, submit their final answer after completing the ranking task(edited or not) and then submit a post-test answer. A student who has been assigned the control condition does not have an initial answer to submit, but they do submit a final answer. They then continue onto the post-test problem without any asynchronous collaboration. This is the business as usual case.

From an initial glance, there is quite a drop off in the number of students completing milestones within this RCT. Naturally, the question arises if there is differential completion of milestones within the study across different conditions. Essentially, if there is any differential completion at certain milestones, I will be unable to make inferences at that level. With there clearly being drop offs at differing milestones within my study, I ran multiple chi-square tests. Figure 6.6 presents the results of all the tests. Each of the p-values are reported on on the left side of Figure 6.6. From this analysis, I can deduce that while there are a number of students not submitting an initial or a final answer, the chi-square tests did not come back significant. Therefore, there is not differential submissions of initial (p-value = 0.255) or final answers (p-value = 0.165) across condition. With that being the case, I can make inferences at the final answer level. However, one chi-square test did come back

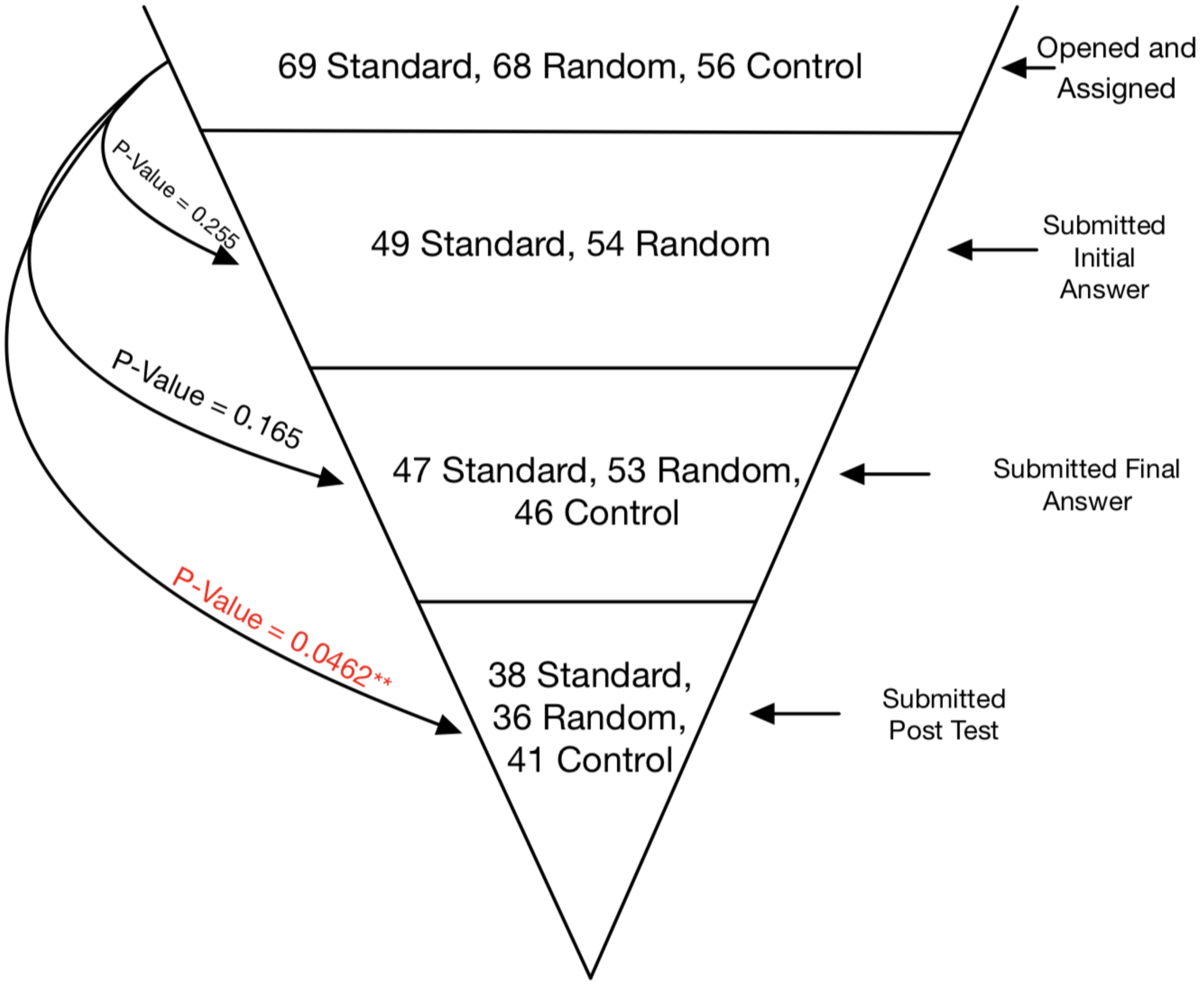


Figure 6.6: # of students who completed milestones in RCT and chi square results

significant: those who completed the milestone of submitting a post-test answer. Essentially, those who completed the entire RCT/study. From the analysis, it is clear that with a p-value of 0.046, that I will be unable to make any inferences at the post-test level. More specifically, there are differential completion of the RCT across the different conditions. With this being the case, I am unable to compare post-test score or perform post-test analysis because of potential selection bias. This rules out analyzing if learning occurred between conditions in the RCT.

While the analysis on the post test is not attainable, inferences can still be made at the student final answer level (the answer they submit after the ranking task). First, of the student who are given the option to edit their original responses (the standard and random condition), is there a reliably better score in the final answer in a condition. So for those with the opportunity to edit, is there a reliable difference, by the condition, in their final answer score? A linear regression was run utilizing the final score of the students final answers as the dependent variable and their initial score (rows were dropped where initial score was NaN - a chi-square test proved not significant, so there is not differential scoring across condition) and assigned condition as the independent variables. Table 6.1 shows that for those who are given the opportunity to edit their original answers, there is no reliable difference in the final score, by condition, when controlling for the initial score (p-value of 0.484 for the standard condition variable).

Even though there isn't a reliable difference in the final score across the conditions, the question remains that if student's are provided the opportunity to edit their work, is there a reliable difference in the answer length across the two conditions? Similar to Model 1, in Table 6.1 , a linear regression was developed using the conditions (standard NITELITE and random) and initial score (once again, dropping any rows where the score was NaN - a chi-square test proved not significant,

Table 6.1: Model 1: Final Answer Score = Intercept + Standard Condition + Initial Score

Residuals:					
Min: -0.30877	1Q: -0.09479	Median: -0.01196	3Q: 0.07087	Max: 2.69123	
	Estimate	**Std. Error**	**t value**	**p-value**	**CI Interval**
(Intercept)	0.261	0.088	2.957	0.004**	[0.085, 0.436]
Initial Score	0.917	0.026	35.586	<2e-16***	[0.866, 0.968]
Standard Condition	0.048	0.069	0.703	0.484	[-0.088, 0.185]

Residual standard error: 0.3268 on 89 degrees of freedom				
Multiple R-squared:	0.9348		**Adjusted R-squared:**	0.9334
F-statistic: 638.4 on 2 and 89 DF, p-value: < 2.2e-16				

so there is not differential scoring across conditions) as the independent variables. This time, however, the dependent variable is the difference in the string lengths between the final answer and the initial answer. In the end, while holding initial score constant and controlling for the initial score, Table 6.2 shows that there is no reliable difference in the change in answer length between conditions.

Another way to approach the question of editing is to examine whether or not the presence of asynchronous collaboration, the standard NITELITE or random rank list condition, effect the difference in answer length between final and initial answers reliably different from 0. To accomplish this, the model changes slightly. By including the control condition, which inherently will have a difference in initial answer and final answer length of 0, this inference can be made. Therefore, the linear model will consist of the initial score a student received, conditions being "Standard or Random" or "Control". Another linear regression is run with the difference in the initial answer and final answer length as the dependent variable. Table 6.3 shows that conditions with asynchronous collaborations (standard NITELITE or random rank list conditions) effect on the length difference between the final answer and initial answer is not reliably different from 0, while controlling for initial score. Again, since the control condition is set within the intercept, and the difference is always 0 for the control, this inference can be made.

As a final analysis with final answers, I set out to identify whether or not, if given the chance to edit, there is a reliable difference in whether a student edits or not by condition when controlling for the initial score? Essentially, is a student in the standard NITELITE condition significantly more likely to edit than students in the random condition. For this analysis, initial score and the condition the student is in that can edit (i.e. standard NITELITE or random rank list) will be the indepedent variables. However, this time a linear regression would not suffice. This

Table 6.2: Model 2: Length Diff Final and Initial Answer = Intercept + Standard Condition + Initial Score

Residuals:					
Min: -147.272	1Q: -0.741	Median: 2.234	3Q: 3.728	Max: 26.752	
	Estimate	**Std. Error**	**t value**	**p-value**	**CI Interval**
(Intercept)	2.248	4.429	0.508	0.613	[-6.552, 11.047]
Initial Score	-1.494	1.296	-1.153	0.252	[-4.069, 1.081]
Standard Condition	1.481	3.454	0.429	0.669	[-5.383, 8.345]

Residual standard error: 16.43 on 89 degrees of freedom				
Multiple R-squared:	0.01811		**Adjusted R-squared:**	-0.003957
F-statistic: 0.8206 on 2 and 89 DF, p-value: 0.4434				

Table 6.3: Model 3: Length Diff Final and Initial Answer = Intercept + Standard or Random Condition + Initial Score

Residuals:					
Min: -148.589	1Q: -0.024	Median: 1.421	3Q: 2.411	Max: 27.453	
	Estimate	**Std. Error**	**t value**	**p-value**	**CI Interval**
(Intercept)	2.003	2.714	0.738	0.462	[-3.365, 7.371]
Initial Score	-0.990	0.850	-1.164	0.247	[-2.672, 0.693]
Standard or Random	-0.455	2.609	-0.175	0.862	[-5.616, 4.705]

Residual standard error:13.6 on 131 degrees of freedom			
Multiple R-squared:	0.01187	**Adjusted R-squared:**	-0.003218
F-statistic: 0.7867 on 2 and 131 DF, p-value: 0.4575			

is because the dependent variable for this analysis is binary, whether they edit or not. With this being the case, a linear model could work, however it would introduce heteroskedasticity. Given this, I train a ordinary least squares linear model with robust standard errors [SA12]. Once again, it is clear that there is not a significant difference in whether or not a student edits their original answer by condition when controlling for the initial score. Table 6.4 supports this given that the independent variable condition standard NITELITE is not statistically significant.

In the end, all models trained to identify whether the conditions are significantly different from each other in terms of the final answer length different or if they edited in general where inconclusive. There was no evidence of the conditions being significantly different. In the end, this is to be expected given that only 9 students chose to edit their original answer out of the 92 students who submitted a final answer. It is difficult to deduce anything from such a small sample size utilizing the editing aspect of NITELITE.

While the student final answers did not show any significant difference between the conditions, the students still had another task which could provide insights into the tools effect on them. It was mentioned earlier that one could infer a student's confidence in their answer, and work, given how they would rank it in comparison to the other student answers. Recall, NITELITE always randomly selects an answer to share from answers which received a perfect grade. This exemplary answer should remain close to the top. For instance, if a student receives a grade of 1, and still ranks their answer above the 4, this could be evidence that the student was confident in their work, but incorrect in the answer. This, of course, is built on the assumption that the students do change the rankings.

When examining the student's interaction with the asynchronous collaboration aspect of NITELITE (and random case), the ranking list in Table 6.5, it is clear that

Table 6.4: Model 4: Edit = Intercept + Standard + Initial Score

Standard Error Type: HC2

	Estimate	**Std. Error**	**t value**	**p-value**	CI Lower	CI Upper	DF
(Intercept)	0.198	0.104	1.916	0.059	-0.007	0.404	89
Initial Score	-0.026	0.031	-0.8404	0.403	-0.088	0.036	89
Standard	-0.060	0.060	-0.999	-0.321	-0.180	0.05973	89

Multiple R-squared:	0.02135		**Adjusted R-squared:**	0.00064
F-statistic: 0.8997 on 2 and 89 DF, p-value: 0.4104				

most of the students put their answers as the highest score possible in the ranking. In the end, a total of 69 of the students, across both the standard NITELITE condition and random condition, put their initial answer as the top one.

Table 6.5: Distribution of original answer final ranking position via condition

	Random	Standard
Highest Score	35	34
Second Highest Score	11	10
Second Lowest Score	3	1
Lowest Score	1	1

What this suggests is that student's often didn't participate in the ranking. NITELITE has the submit ranking button and the student didn't have to move anything to be able to hit the submit ranking button. While it appears that student's didn't utilize the ranking aspect of NITELITE (or the random ranking condition), another way to check was to examine the distribution of the rankings given their initial answer's score. From Table 6.6 it is more apparent that many students did not participate in the ranking activity. Of those who didn't get a perfect 4/4 score, only 13 out of the 54 non perfect grades moved their answer below the top spot. It should be noted that the slight difference in totals is because of dropping the rows which didn't have a score associated with them.

Table 6.6: Distribution of original answer final ranking position via initial answer score

	Initial Answer Score				
	0	1	2	3	4
Highest Score	9	4	10	18	24
Second Highest Score	1	0	4	4	11
Second Lowest Score	1	0	0	2	1
Lowest Score	0	0	1	0	0

Another question which could be ask is if there was any evidence to show that students are more or less likely to rank their answer in the top half over the bottom half of the rankings. After a quick chi-squared test, this came out to be not significant. A final analysis, which could provide more insight into the student's behavior with the ranking in the standard NITELITE and random ranking list condition, utilizes the Spearman rho correlation. This is rank based correlation calculation. Essentially, this will show whether or not there is a correlation between x and the rank of y. In this case, I examine whether there is a correlation between the initial score of a student's initial/original answer and the rank position of their answer. This can help illustrate if students are interacting with the rankings properly. If they are, it would be expected that there would be a mild amount of correlation between the two variables. In the end, the Spearman rho correlation was not significant with a rho value 0.095. The closer to 0 the rho value, the lower amount of correlation between the variables. In this case, its nearly 0. This helps to identify that students didn't interact with the ranking system in the way it was meant to.

6.5 Limitations and Future Work

In the end, most of the analysis came back not significant and that the conditions were not significantly different from each other in terms of any analysis. This could be attributed to many factors. First, in terms of ranking, it would be beneficial to adjust the ranking aspect of NITELITE to shuffle the order of other student answers being shown. This could help encourage students to move their answers around when their answer isn't always on top. Secondly, it was surprising to see how few students decided to edit their original answers. There was a wide range of scores which the students obtained, and they were always shown an exemplary answer along with similar and dissimilar student answers. So when a student received a 0 score on their initial answer, as some did, I'm surprised only a few decided to edit.

Most students, specially those who know they didn't know the answer, would want to attempt to get some points by editing their original answer. Therein lies one limitation of this work; students treated this study less like an assignment and more like an extra activity. With this being the case, students went in with a different mentality than if the study was assigned and presented as an actual assignment. Students will naturally behave differently for an assignment versus a pilot study.

Another limitation of this work was the number of students who dropped out and never finished the study. Given that there was differential completion of this RCT across the different conditions, analysis could not be performed to make any inferences at the post test level. This meant that any analysis on potential learning occurring would be inaccurate. What is interesting is that most of the drop out came in the conditions which provided students the asynchronous collaboration. However, based on the analysis of the student behavior with the rankings, there may be evidence that the students didn't want to take the time to rank the answers. Thus, fatigue may have set in on the students in the random rank list condition and the standard NITELITE condition. In the future, I would possibly look to changing the ranking. Possibly simplifying it, maybe even removing it for the next study. My goal is to attempt this again in the future but simplify NITELITE and the asynchronous collaboration. This would help narrow down the cause of the dropout amongst students. It is the case that, similar to the students not editing, student may not have had a similar mentality as to a standard assignment.

Even though there were some shortcomings to this pilot study of NITELITE, I felt this was a successful trial of the tool. Students did successfully utilize aspects of the tool and it was successfully integrated into a problem set within ASSISTments. From this successful pilot study, I look to adapt a couple pieces of NITELITE and run another RCT in the near future. It would be my goal to be even more specific in the instructions to the teachers about how to present the tool. To help students work through the tool as a true assignment and less like a pilot study.

6.6 Conclusion

Overall, the RCT was a successful pilot study for my asynchronous collaboration tool NITELITE. Students were able to work through their problem set with NITELITE integrated into ASSISTments. NITELITE successfully utilized modern natural language processing presented in previous chapters within this dissertation. More specifically, NITELITE was able to successfully and efficiently run SBERT, as presented in Chapter 4, as a prediction model for NITELITE. This is also the model I tested in Chapter 3 for potential unfairness (and was not able to identify any unfairness). Additionally, SBERT was able to efficiently populate 2 out of the 3 rows of the asynchronous tool. NITELITE was efficient in its presentation and students progressed through it with minimal issue.

In the end, NITELITE embodies much of the work within this dissertation. Utilizing language to help develop a support system for students working through open response questions. I feel this is a stepping stone for future work with NITELITE. Tweaks can be made and future studies can be run to test out different variations of the milestones within NITELITE. Even with students dropping out, it was clear from the data that students have similar and differing rationale to solving both the pre-test and the post-test within this study. This helps to support the diversification of content teachers can deliver to students. NITELITE embodies that by supporting students through open response questions and teachers gaining deeper insights into the student's true understanding of the material with the ranking system. Language is a powerful tool for learning. While there will be correlational mistakes made in math, what makes mathematics beautiful is the fact that there are multiple perspectives to solving a problem. As long as a student follows the proper set of rules, there isn't a limitation to how they can approach a problem. This is the beauty of critical thinking, the beauty of mathematics and the beauty of language in learning.

Bibliography

[AACKS04] Tony Abou-Assaleh, Nick Cercone, Vlado Keselj, and Ray Sweidan. N-gram-based detection of new malicious code. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, volume 2, pages 41–42. IEEE, 2004.

[AB06] Yigal Attali and Jill Burstein. Automated essay scoring with e-rater® v. 2. *The Journal of Technology, Learning and Assessment*, 4(3), 2006.

[ABR85] John R Anderson, C Franklin Boyle, and Brian J Reiser. Intelligent tutoring systems. *Science*, 228(4698):456–462, 1985.

[AXV$^+$11] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca J Passonneau. Sentiment analysis of twitter data. In *Proceedings of the workshop on language in social media (LSM 2011)*, pages 30–38, 2011.

[BBE$^+$21] Sami Baral, Anthony Botelho, John Erickson, Priyanka Benachamardi, and Neil Heffernan. Improving automated scoring of student open responses in mathematics. In *Proceedings of the Fourteenth International Conference on Educational Data Mining, Paris, France.*, 2021.

[BCZ$^+$16] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*, pages 4349–4357, 2016.

[BGS15] Steven Burrows, Iryna Gurevych, and Benno Stein. The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, 25(1):60–117, 2015.

[BR08] Andrew C Butler and Henry L Roediger. Feedback enhances the positive effects and reduces the negative effects of multiple-choice testing. *Memory & cognition*, 36(3):604–616, 2008.

[BS06] Stephen Buckles and John J Siegfried. Using multiple-choice questions to evaluate in-depth learning of economics. *The Journal of Economic Education*, 37(1):48–57, 2006.

[CG16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[CKA97] Albert T Corbett, Kenneth R Koedinger, and John R Anderson. Intelligent tutoring systems. In *Handbook of human-computer interaction*, pages 849–874. Elsevier, 1997.

[CL14] Elisabeth G Cohen and Rachel A Lotan. *Designing groupwork: Strategies for the heterogeneous classroom Third Edition*. Teachers College Press, 2014.

[CLB+19] Elizabeth S Charles, Nathaniel Lasry, Sameer Bhatnagar, Rhys Adams, Kevin Lenton, Yann Brouillette, Michael Dugdale, Chris Whittaker, and Phoebe Jackson. Harnessing peer instruction in-and out-of class with mydalite. In *Education and Training in Optics and Photonics*, page 11143_89. Optical Society of America, 2019.

[CLBD13] Donald A Curtis, Samuel L Lind, Christy K Boscardin, and Mark Dellinges. Does student confidence on multiple-choice question assessments provide useful information? *Medical education*, 47(6):578–584, 2013.

[CT+94] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*, volume 161175. Citeseer, 1994.

[CYK+18] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dil99] Pierre Dillenbourg. What do you mean by collaborative learning?, 1999.

[DWRK10] Dejana Diziol, Erin Walker, Nikol Rummel, and Kenneth R Koedinger. Using intelligent tutor technology to implement adaptive support for student collaboration. *Educational Psychology Review*, 22(1):89–102, 2010.

[EBM$^+$20] John A Erickson, Anthony F Botelho, Steven McAteer, Ashvini Varatharaj, and Neil T Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics & Knowledge*, pages 615–624, 2020.

[FLL99] Peter W Foltz, Darrell Laham, and Thomas K Landauer. Automated essay scoring: Applications to educational technology. In *EdMedia+ innovate learning*, pages 939–944. Association for the Advancement of Computing in Education (AACE), 1999.

[GBB19] Josh Gardner, Christopher Brooks, and Ryan Baker. Evaluating the fairness of predictive student models through slicing analysis. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 225–234, 2019.

[GF12] Wael H Gomaa and Aly A Fahmy. Short answer grading using string similarity and corpus-based similarity. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 3(11), 2012.

[GH14] CHE Gilbert and Erric Hutto. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf*, volume 81, page 82, 2014.

[Gok95] Anuradha A Gokhale. Collaborative learning enhances critical thinking. 1995.

[GVR$^+$01] Arthur C Graesser, Kurt VanLehn, Carolyn P Rosé, Pamela W Jordan, and Derek Harter. Intelligent tutoring systems with conversational dialogue. *AI magazine*, 22(4):39–39, 2001.

[GWHWH$^+$00] Arthur C Graesser, Peter Wiemer-Hastings, Katja Wiemer-Hastings, Derek Harter, Tutoring Research Group Tutoring Research Group, and Natalie Person. Using latent semantic analysis to evaluate the contributions of students in autotutor. *Interactive learning environments*, 8(2):129–147, 2000.

[HH14] Neil T Heffernan and Cristina Lindquist Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[HT01] David J Hand and Robert J Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 45(2):171–186, 2001.

[Joa96] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science, 1996.

[JRVF09] Giuseppe Jurman, Samantha Riccadonna, Roberto Visintainer, and Cesare Furlanello. Canberra distance on ranked lists. In *Proceedings of advances in ranking NIPS 09 workshop*, pages 22–27. Citeseer, 2009.

[KIK20] Zenun Kastrati, Ali Shariq Imran, and Arianit Kurti. Weakly supervised framework for aspect-based sentiment analysis on students' reviews of moocs. *IEEE Access*, 8:106799–106810, 2020.

[KKH13] Paul Kehrer, Kim Kelly, and Neil Heffernan. Does immediate feedback while doing homework improve learning? In *The Twenty-Sixth International FLAIRS Conference*, 2013.

[KKR$^+$16] Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, Laszlo Lukacs, Marina Ganea, Peter Young, et al. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 955–964, 2016.

[Ku09] Kelly YL Ku. Assessing students' critical thinking performance: Urging for measurements using multi-response format. *Thinking skills and creativity*, 4(1):70–76, 2009.

[LG12] Marjan Laal and Seyed Mohammad Ghodsi. Benefits of collaborative learning. *Procedia-social and behavioral sciences*, 31:486–490, 2012.

[LJW18] Ha Le, Jeroen Janssen, and Theo Wubbels. Collaborative learning practices: teacher and student perceived obstacles to effective student collaboration. *Cambridge Journal of Education*, 48(1):103–122, 2018.

[LM14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[LVWB15] Andrew S Lan, Divyanshu Vats, Andrew E Waters, and Richard G Baraniuk. Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 167–176. ACM, 2015.

[LXZ19] Jiawei Liu, Yang Xu, and Yaguang Zhu. Automated essay scoring based on two-stage learning. *arXiv preprint arXiv:1901.07744*, 2019.

[Mar99] Michael E Martinez. Cognition and the question of test item format. *Educational Psychologist*, 34(4):207–218, 1999.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[ML07] Miika Marttunen and Leena Laurinen. Collaborative learning through chat discussions and argument diagrams in secondary school. *Journal of Research on Technology in Education*, 40(1):109–126, 2007.

[MSB+14] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[Nwa90] Hyacinth S Nwana. Intelligent tutoring systems: an overview. *Artificial Intelligence Review*, 4(4):251–277, 1990.

[OBA+14] Jennifer K Olsen, Daniel M Belenky, Vincent Aleven, Nikol Rummel, Jonathan Sewall, and Michael Ringenberg. Authoring tools for collaborative intelligent tutoring system environments. In *International conference on intelligent tutoring systems*, pages 523–528. Springer, 2014.

[OBAR14] Jennifer K Olsen, Daniel M Belenky, Vincent Aleven, and Nikol Rummel. Using an intelligent tutoring system to support collaborative as well as individual learning. In *International Conference on Intelligent Tutoring Systems*, pages 134–143. Springer, 2014.

[OBKM13] Yasuhiro Ozuru, Stephen Briner, Christopher A Kurby, and Danielle S McNamara. Comparing comprehension measured by multiple-choice and open-ended questions. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 67(3):215, 2013.

[OT20] Aytuğ Onan and Mansur Alp Toçoğlu. Weighted word embeddings and clustering-based identification of question topics in mooc discussion forum posts. *Computer Applications in Engineering Education*, 2020.

[PBB$^+$12] Vitaliy Popov, Dine Brinkman, Harm JA Biemans, Martin Mulder, Andrei Kuznetsov, and Omid Noroozi. Multicultural student group work in higher education: An explorative case study on challenges as perceived by students. *International Journal of Intercultural Relations*, 36(2):302–317, 2012.

[PP10] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326, 2010.

[Pri] Automated Student Assessment Prize. The hewlett foundation: Automated essay scoring.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[R$^+$03] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ, 2003.

[RAMK11] Ido Roll, Vincent Aleven, Bruce M McLaren, and Kenneth R Koedinger. Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and instruction*, 21(2):267–280, 2011.

[RAS17] Endah Retnowati, Paul Ayres, and John Sweller. Can collaborative learning improve the effectiveness of worked examples in learning mathematics? *Journal of educational psychology*, 109(5):666, 2017.

[RFMM16] Jeremy Roschelle, Mingyu Feng, Robert F Murphy, and Craig A Mason. Online mathematics homework increases student achievement. *AERA Open*, 2(4):2332858416673968, 2016.

[RG19a] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[RG19b] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[RHC+17] Brian Riordan, Andrea Horbach, Aoife Cahill, Torsten Zesch, and Chong Min Lee. Investigating neural architectures for short answer scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 159–168, 2017.

[Ric88] Jeffrey Ralph James Richardson. *Foundations of intelligent tutoring systems*. Psychology Press, 1988.

[RIM05] Henry L Roediger III and Elizabeth J Marsh. The positive and negative consequences of multiple-choice testing. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 31(5):1155, 2005.

[RV06] Michael A Ringenberg and Kurt VanLehn. Scaffolding problem solving with annotated, worked-out examples to promote deep learning. In *International conference on intelligent tutoring systems*, pages 625–634. Springer, 2006.

[RYR+16] Carly Robinson, Michael Yeomans, Justin Reich, Chris Hulleman, and Hunter Gehlbach. Forecasting student achievement in moocs with natural language processing. In *Proceedings of the sixth international conference on learning analytics & knowledge*, pages 383–387. ACM, 2016.

[SA12] Cyrus Samii and Peter M Aronow. On equivalencies between design-based and regression-based variance estimators for randomized experiments. *Statistics & Probability Letters*, 82(2):365–370, 2012.

[SB09] Jana Zuheir Sukkarieh and John Blackmore. c-rater: Automatic content scoring for short constructed responses. In *Twenty-Second International FLAIRS Conference*, 2009.

[SGA$^+$15] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714*, 2015.

[SH14] Douglas Selent and Neil Heffernan. Reducing student hint use by creating buggy messages from machine learned incorrect processes. In *International conference on intelligent tutoring systems*, pages 674–675. Springer, 2014.

[SK05] Mark G Simkin and William L Kuechler. Multiple-choice tests and student understanding: What is the connection? *Decision Sciences Journal of Innovative Education*, 3(1):73–98, 2005.

[SPR03] Jana Z Sukkarieh, Stephen G Pulman, and Nicholas Raikes. Automarking: using computational linguistics to score short, free text responses. 2003.

[SPW$^+$13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[SSS16] Md Arafat Sultan, Cristobal Salazar, and Tamara Sumner. Fast and easy short answer grading with high accuracy. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1070–1075, 2016.

[SSW$^+$11] Hao Sun, Xian Sun, Hongqi Wang, Yu Li, and Xiangjuan Li. Automatic target detection in high-resolution remote sensing images using spatial sparse coding bag-of-words model. *IEEE Geoscience and Remote Sensing Letters*, 9(1):109–113, 2011.

[TCC$^+$01] Patrick T Terenzini, Alberto F Cabrera, Carol L Colbeck, John M Parente, and Stefani A Bjorklund. Collaborative learning vs. lecture/discussion: Students' reported learning gains. *Journal of Engineering Education*, 90(1):123–130, 2001.

[TMD14] Bruno Trstenjak, Sasa Mikac, and Dzenana Donko. Knn with tf-idf based framework for text categorization. *Procedia Engineering*, 69:1356–1364, 2014.

[TN16] Kaveh Taghipour and Hwee Tou Ng. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, 2016.

[TRM10] Pierre Tchounikine, Nikol Rummel, and Bruce M McLaren. Computer supported collaborative learning and intelligent tutoring systems. In *Advances in intelligent tutoring systems*, pages 447–463. Springer, 2010.

[Van11] Kurt VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[WB05] Karyn Woodford and Peter Bancroft. Multiple choice questions not considered harmful. In *Proceedings of the 7th Australasian conference on Computing education-Volume 42*, pages 109–116. Citeseer, 2005.

[Wea06] Melanie R Weaver. Do students value feedback? student perceptions of tutors' written responses. *Assessment & Evaluation in Higher Education*, 31(3):379–394, 2006.

[WRK14] Erin Walker, Nikol Rummel, and Kenneth R Koedinger. Adaptive intelligent support to improve peer tutoring in algebra. *International Journal of Artificial Intelligence in Education*, 24(1):33–61, 2014.

[WS92] W John Wilbur and Karl Sirotkin. The automatic identification of stop words. *Journal of information science*, 18(1):45–55, 1992.

[YPVG$^+$14] David Scott Yeager, Valerie Purdie-Vaughns, Julio Garcia, Nancy Apfel, Patti Brzustoski, Allison Master, William T Hessert, Matthew E Williams, and Geoffrey L Cohen. Breaking the cycle of mistrust: Wise interventions to provide critical feedback across the racial divide. *Journal of Experimental Psychology: General*, 143(2):804, 2014.

[ZJZ10] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4):43–52, 2010.

[ZM17] Rui Zhao and Kezhi Mao. Fuzzy bag-of-words model for document representation. *IEEE transactions on fuzzy systems*, 26(2):794–804, 2017.

[ZZX+17] Siyuan Zhao, Yaqiong Zhang, Xiaolu Xiong, Anthony Botelho, and Neil Heffernan. A memory-augmented neural model for automated grading. In *Proceedings of the Fourth (2017) ACM Conference on Learning@ Scale*, pages 189–192. ACM, 2017.