# Automatic Short Math Answer Grading via In-context Meta-learning

Mengxue Zhang
Massachusetts Amherst
mengxuezhang@cs.umass.edu

Sami Baral
Worcester Polytechnic Institute
sbaral@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
nth@wpi.edu

Andrew Lan
Massachusetts Amherst
andrewlan@cs.umass.edu

## ABSTRACT

Automatic short answer grading is an important research direction in the exploration of how to use artificial intelligence (AI)-based tools to improve education. Current state-of-the-art approaches use neural language models to create vectorized representations of students responses, followed by classifiers to predict the score. However, these approaches have several key limitations, including i) they use pre-trained language models that are not well-adapted to educational subject domains and/or student-generated text and ii) they almost always train one model per question, ignoring the linkage across question and result in a significant model storage problem due to the size of advanced language models. In this paper, we study the problem of automatic short answer grading for students' responses to math questions and propose a novel framework for this task. First, we use MathBERT, a variant of the popular language model BERT adapted to mathematical content, as our base model and fine-tune it on the downstream task of student response grading. Second, we use an in-context learning approach that provides scoring examples as input to the language model to provide additional context information and promote generalization to previously unseen questions. We evaluate our framework on a real-world dataset of student responses to open-ended math questions and show that our framework (often significantly) outperform existing approaches, especially for new questions that are not seen during training.

## Keywords

Automated scoring, Short-answer scoring, Math grading

## 1. INTRODUCTION

Automated scoring (AS) refers to the problem of automatically scoring student (textual) responses to open-ended questions with multiple correct answers, often utilizing various machine learning algorithms. AS approaches can potentially scale up human grading effort: by training on a small number of example scores provided by human experts, they can automatically score a large number of responses. With the advancement in online learning platforms in recent years, there has been a growing body of research around the development of AS methods. AS has been studied in many different contexts, including automated essay scoring (AES) [1, 30] and automatic short answer grading (ASAG) [39, 51], which has been studied in various different subject domains [5, 12, 14, 37, 17, 3, 29, 49]. The majority of AS approaches follow two steps: First, obtaining a *representation* of student responses, often using methods in natural language processing, and second, applying a *classifier* on top of this representation to predict the score [4, 27]. Over the years, AS approaches have gradually shifted from classic text representations such as bag-of-words or human-crafted features [8, 16, 19, 30, 32, 41] that are human-interpretable to more abstract representations based on pre-trained neural language models [24, 26, 38, 40, 45].

In this paper, we focus on ASAG in one particular subject domain: Mathematics. Math questions, or questions that involve mathematical reasoning, are ubiquitous in many science, technology, engineering, and mathematics (STEM) subject domains. Recently, several works [5, 14] have studied ASAG for the responses students provide to math-based open-ended questions that (are often concise) include their reasoning or thinking process about a particular concept. As noted in prior work, a key technical challenge in this domain is that student responses to math-based open-ended questions often are a combination of text (natural language) and mathematical language (symbols, expressions, and equations). However, most existing pre-trained language models such as BERT [13] and GPT [7] are not specifically designed for mathematical language. Therefore, existing approaches for math ASAG that do not address the mathematical language present in student responses [5, 14] may not be able to accurately represent student reasoning processes in their responses. On the other hand, existing methods that focus entirely on mathematical language [22, 42, 50] cannot process natural language contained in open-ended responses.

Another significant limitation of existing AS approaches is that, in most cases, we need to train a separate AS model for each question. In contexts such as AES where questions (essay prompts) may not have high similarities, this approach can often be effective. However, in other contexts where reading comprehension or reasoning is involved,

multiple questions may be linked to each other through the background information provided. In the context of math questions, many questions share similar skills or are different parts of a multi-step question. Therefore, training a separate AS model for each question would result in models that can only identify typical patterns in student responses to each individual question but cannot really understand how to differentiate good responses from bad ones. It is likely that these models would not be able to generalize well to previously unseen questions, as noted in [12]. More importantly, training a separate AS model for each question may create a significant problem for model storage and management. This problem is especially significant for state-of-the-art AS approaches that fine-tune pre-trained language models that have millions of parameters.

## 1.1 Contributions

In this paper, we develop an ASAG framework for students' open-ended responses to math questions. Building on a grand prize winning solution [15] to the National Assessment of Educational Progress (NAEP) Automated Scoring Challenge[1], our framework is based on fine-tuning a pre-trained BERT language model on actual student responses, with several main innovations:

- First, we use MathBERT [35], a version of the popular BERT language model adapted to mathematical content, as our base model. This model is capable of understanding math symbols and expressions to some extend and help us obtain a better representation of open-ended student responses.
- Second, we leverage in-context learning ideas in NLP research [11, 28] and develop an ASAG approach using on multi-task and meta-learning tools (that are popular machine learning tools to promote model generalizability). Specifically, we fine-tune MathBERT with a carefully designed input format that uses *example responses and scores* as additional input (together with question and response texts) to provide additional context of each question. This input format helps us train a shared AS model across all questions and outperformances the current state-of-arts approach [5].
- Third, we show that meta in-context learning leads to highly generalizabile AS models. Our intuition on why our approach is highly effective is that, by explicitly using example responses and scores as input, we reduce the AS task to a *similar response finding* task, which is easier for the model to learn.

We evaluate our ASAG framework on a real-world dataset which contains students' solution processes to open-ended math questions and grades provided by teachers. Through a series of quantitative experiments, we show that our framework (sometimes significantly) outperforms existing approaches in terms of score prediction performance. More importantly, we show that our framework significantly outperforms existing approaches [5, 12] (by up to 50% on some metrics under some settings) when applied to questions that are previously unseen during training, using only a few scored examples for these new questions. Perhaps surprisingly, we found that MathBERT does not provide additional benefit on top of the original BERT model while the in-context fine-tuning setup is key to the excellent generalization performance. We also summarize observations from qualitative evaluations of scoring errors, discuss the limitations of our framework, and outline several avenues for future work. Our implementation is publicly available.[2]

[1] https://github.com/NAEP-AS-Challenge/info

# 2. RELATED WORK

In recent years, there have been many developments in ASAG methods across various domains. Most of the prior works have focused on non-mathematical domains [6, 10] where student responses are purely textual. However, more AS works have started to focus on more specific domains that contain non-textual symbols, e.g., Math, Physics, Chemistry, Biology and Computer Science [19, 30, 41]. In these domains, a combination of natural language processing methods for the representation of responses and machine learning methods for score classification has shown promising results [5, 14, 23, 36, 38].

Here, we discuss two recent AS works in the mathematical domain, [5] and [12], that are the most relevant to our research. The authors of [5] proposed a scoring approach for short-answer math questions using sentence-BERT (SBERT)-based representation of student responses. Compared to this approach, our approach differs in many aspects and we highlight the following: First, we use MathBERT, a model pre-trained on mathematical content to represent student responses, while the approach in [5] ignores mathematical language in student responses. Second, we use an in-context meta-training approach to train one AS model for all questions while the approach in [5] trains one AS model for each question, which likely limits its generalizability to previously unseen questions.

The authors of [12] proposed a similar scoring approach for short-answer critical reasoning questions that combines various pre-trained representations, including SBERT, with classifiers for AS. Instead of using only student responses as input to the classifiers, they also use a series of question context information such as question text, rubric text, and question cluster identifier. As a result, they showed that their AS approach can generalize to previously unseen questions. Compared to this approach, our approach mostly differs in two aspects: First, we fine-tune MathBERT on actual student responses while the approach in [12] leaves the pre-trained representations fixed, which likely limits the accuracy of their student response representations. Second, we use scoring examples as input to MathBERT in addition to question text to further provide the AS model context of the question, which further enhances the generalizability of our model to previously unseen questions in a *few-shot* learning setting.

# 3. METHODOLOGY

In this section, we detail both the ASAG setup for math questions and our in-context meta-learning framework.

## 3.1 Problem Statement

[2] https://github.com/kikumaru818/meta_math_scoring

We treat math ASAG as a classification problem where our goal is to train a scoring model that is capable of generalizing to new, previously unseen questions using a few examples. This setting is well studied in machine learning, commonly referred to as few-shot learning [7, 11, 28], where the goal is to train robust models that excel at multiple tasks. Formally, we have a set of questions $T = \{Q_1, Q_2, Q_3, \ldots, Q_n\}$, where each question $Q_i \in T$ can be seen as a classification task. Each question $Q_i$ comes with numerous graded, training examples: $\{e_i^1, e_i^2, \ldots\}$. Each example consists of multiple fields of information: $e_j^i = \langle q_{text}, q_{id}, x, y\rangle$, where $q_{text}$ is the textual statement of the question, $q_{id}$ is an unique question id, $x$ is the text of student's response and $y$ is the grade from the teacher. We study on two problem settings in this work: i) generalization to new responses and ii) generalization to new questions.

#### 3.1.1 Generalization to new responses

This problem setting follows from that used in prior work [5]: we train a scoring method on scored responses for all questions and test it on held-out responses. We treat this problem setting as supervised learning classification and learn a scoring model $f : x \mapsto \hat{y}$ that predicts an estimated score $\hat{y}$ for a student response $x$ with true score $y$ by minimizing a loss function $L(y, \hat{y})$. For each question $Q_i$, we split the corresponding scored responses into two subsets, $Q_i^{Train}$ and $Q_i^{test}$, such that $Q_i^{Train} \cup Q_i^{test} = Q_i$ and $Q_i^{Train} \cap Q_i^{test} = \emptyset$. Instead of treating each question separately and train a model for each, we train one unified model on the union of training datasets for all questions, i.e., $\bigcup_{i=1}^{|T|} Q_i^{train}$. We detail the scoring model and our in-context learning setup in Section 3.3.

Let $\theta$ represent the model parameters, the optimization objective $\mathcal{L}_i$ for question $i$ is simply the cross entropy, i.e., the negative log-likelihood loss

$$\mathcal{L}_i(\theta) = \sum_{j:(x_j^i, y_j^i) \in Q_i^{train}} [-\log p_\theta(y_j^i | x_j^i, \ldots)].$$

We minimize the total objective that spans all questions

$$\mathcal{L}(\theta) = \sum_{i=1}^{|T|} \mathcal{L}_i(\theta)$$

to learn the model parameters $\theta$.

#### 3.1.2 Generalization to new questions

This problem setting can be formulated as a few-shot (or zero-shot) classification problem: we train a scoring model on scored responses for some questions and test its generalization capability to student responses to held-out questions. We first split the set of questions $T$ into $T_{train}$ and $T_{test}$ such that $T_{train} \cup T_{test} = T$ and $T_{train} \cap T_{test} = \emptyset$. We train the scoring model on all scored responses for the training questions $\bigcup_{i=1}^{|T_{train}|} Q_i$. Let $\gamma$ represent the model parameters for this problem setting, the optimization objectives for each question and across all training questions change to

$$\mathcal{L}_i(\gamma) = \sum_{j:(x_j^i, y_j^i) \in Q_i} [-\log p_\gamma(y_j^i | x_j^i, \ldots)]$$

and

$$\mathcal{L}(\gamma) = \sum_{i=1}^{|T_{train}|} \mathcal{L}_i(\gamma),$$

respectively.

At test time, we applied the trained model to new questions $Q_i \in T_{test}$ to see how it can adapt using few (or zero) scored examples for these new questions. We study two cases: i) we do not update the original model with gradient updates, i.e., $\gamma$ remains unchanged, which we call the **Meta** setting, and ii) we update $\gamma$ by backpropagating gradients calculated on a few scored responses for new questions, which we call the **Meta-finetune** setting.

### 3.2 BERT-based classification

We now detail our scoring method based on fine-tuning a pre-trained language model. BERT [13] is a pre-trained language model that produces contextualized representations of text and is also capable of encoding text. We use MathBERT [35], a variant of BERT pre-trained on a large mathematical corpus containing mathematical learning content ranging from pre-kindergarten (pre-k), high school, to college graduate levels. We use MathBERT as our base language model and fine-tune it on our data for downstream ASAG classification.

Figure 1 visualizes our method. The input to BERT is a sequence of tokens, starting with the [CLS] token, a special symbol added in front of every input during training process for BERT based model. Since [CLS] doesn't have meaning itself and BERT-based models learn contextualized representations of text, we can use the [CLS] embedding as a representation that encodes the entire input. We then feed the [CLS] embedding to a classification layer followed by softmax [18], obtaining the predictive score class probabilities. A key difference between our work and prior works [5, 12] that use BERT is that we also fine-tune the BERT model, i.e., update its parameters and adapt it to ASAG. Prior works only use BERT-based models to extract the representation of student responses; these methods are not likely going to be effective since they cannot adapt to student-generated content. During training, we backpropagate the gradient on the prediction objective to both i) the classification layer, which is learned from scratch, and ii) BERT, which is updated from its pre-trained parameter values.

### 3.3 In-context Meta-learning

Our key technical insight is that we need to use a well-crafted input format to provide context to the model and help it adapt to the scoring task for each question. Therefore, instead of only inputting the target student response we want to grade, we also include several other features as the input. These features are important to ground the model in the context of each question. For each possible feature, we also add additional textual instructions as input to the model about the semantic meaning of the feature.

Table 1 shows all possible features we include as model input and the corresponding template. Student response denotes the target responses to be scored. Thus, the corresponding textual instruction is "score this answer." Since student

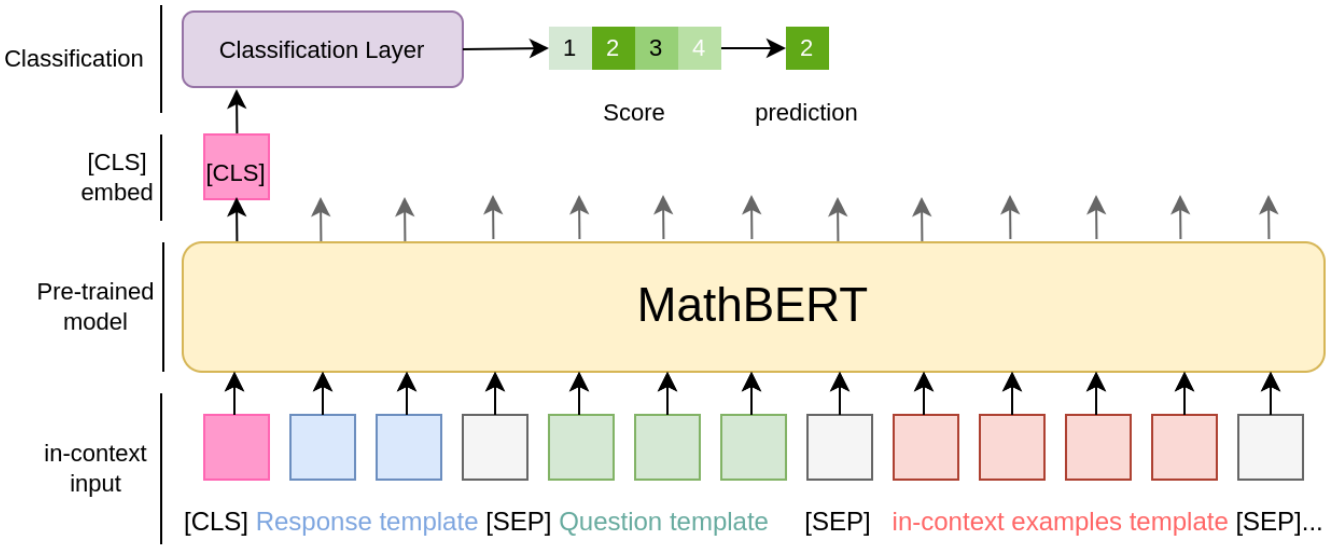


**Figure 1: Overview of our in-context meta-learning-based ASAG method for math questions.**

**Table 1: Templates for different components we use as input into our scoring method.**

| Input Feature | Template | Sample Text |
|---|---|---|
| Student Response | *score this answer:* $x_j^i$ | *score this answer: expand the equation we get* $2x+2=1$ *then* $x=-0.5$ |
| Question | *question text:* $q_{text_j}^i$ | *question text: Solve the equation* $2(x+1)=1$ |
| Question ID | *question id:* $q_{id_j}^i$ | *question id: 21314* |
| Scale | *scale: possible grade for question i* | *scale: poor, fair, good, excellent* |
| Example | *example:* $x_{\neg j}^i$, *score:* $y_{\neg j}^i$ | *example: move 2 to the right* $x=1/2$, *score: fair* |

responses are essential to the grading task, we place it directly after the [CLS] token. After the student response, we add either the question text or the question ID as input to the BERT model. Question text can help the model understand the question context and generalize across questions by leveraging their semantic relations. Question IDs enable the model to identify which question the target response belongs to, which can be helpful when the question text is not semantically meaningful; see Section 4.3 for an example. We can also add textual descriptions of the grading scales to the input. Since we use language models that are better at understanding text than numbers, we use "bad, poor, fair, good, excellent" to represent scores of 0, 1, 2, 3, and 4, respectively.

Another key innovation is that, following recent approaches [11, 28] for meta-training based in-context learning, we also input examples of scored responses, i.e., responses and corresponding scores, $(x_{\neg j}^i, y_{\neg j}^i)$ from training dataset that belong to same question of the target response $x_j^i$. These examples provide further context to the model that the model can use to relate the target response to. Intuitively, when these examples are presented in the input, the AS model only needs to find example responses that are similar to the current response and use their scores to help score the current response. This task is easier for the AS model to learn than the real AS task when examples are not in the input.

# 4. EXPERIMENTS

This section details the experiments we conducted to validate our in-context meta-training approach for ASAG. Section 4.1 discusses details on the real-world dataset student response dataset we use and how our pre-processing steps. Section 4.2 details evaluation metrics and baselines. We design three groups of experiments to test our approach's performance. In section 4.3, we examine how the approach performs on generalizing new student responses while having an assumption that the questions have already been seen during the training process. In section 4.4, we examine the performance of our approach generalizing to scoring student response to new questions; in section 4.3.3, we run experiments to test which part of the in-context has the most significant impact on the performance of our approach.

## 4.1 Dataset

In this study, we use data collected from an online learning platform that has been used in prior work [5, 14]. The dataset contains student responses to open-ended questions paired with scores provided by human graders. The dataset used in [5, 14] consists of 141,612 total student responses from 25,069 students to 2,042 questions, scored by 891 different graders. The numeric score given to each response is in a 5-point scale from 0-4 with 4 as full credit and 0 as no credit. We refer to this dataset as $D_{orig}$.

$D_{orig}$ contains some noisy data points that increase the difficulty of learning. First, some student responses are the same, but the teacher grades are different. Second, all cor-

responding student responses get full credit for some questions. For example, even the student's response is "I do not know", the response's grade is still full credit. Third, some students' responses are answered by image, making the text content empty. Fourth, similar issues on question body; some questions do not have semantic meaning (such as questions that refer to a question in a book that we cannot access) or are represented as tables or images. For this work, we mainly focus on questions with corresponding students' responses and scoring the responses no matter which student is and who is grading. Thus we hope to reduce the effect of these noisy data points and further clean up the dataset. We found that some student responses are the same but the teacher grades are different; therefore, we re-label 2,130 inconsistent responses with the highest grade that the corresponding response text can get. We remove 8,835 student responses that contain only images or broken characters (non-English words, non-math terms). Since our in-context meta-learning approach needs to learn coherent information between questions using question text, we need high qualify question text. We remove responses (9,930 number of responses) with a question body (231 number of questions) that does not have semantic meaning. We also remove questions (478) that contain less than 25 number of students' responses. We called the new dataset $D_{clean}$, it contains 131,046 responses in total and 1,333 questions. Table 2 shows some examples data points of this dataset. For each data point, it contains the student response, problem text, problem id and teacher grade.

## 4.2 Metrics

For the evaluation of math ASAG methods, we utilize three evaluation metrics for categorical, integer-valued scores, following prior work [5, 14]. The first metric is area under the receiver operating characteristic curve (**AUC**), which is designed for binary classification problems. Instead, we calculate the AUC in a way similar to [21] by averaging the AUC numbers over each possible score category, treating them as separate binary classification problems. The second metric is the root mean squared error (**RMSE**) which simply treats the score categories as numerical values. The third and most important metric is the multi-class Cohen's **Kappa** that is often used for ordered categories, which fits the setting of our ASAG data.

## 4.3 Scoring new responses

### 4.3.1 Experimental Setting

For this experiment, we focus on comparing the performance of our approach to baselines on generalizing to new responses. We randomly divide all example **responses** in $D_{orig}$ (we use this dataset for a fair comparison to [5, 14]) into 10 equally-sized folds for cross validation. For each run, we use 8 folds for training, 1 fold for validation to select a training epoch with the best performance on this fold and 1 fold for the final testing of all methods. Under this setting, we ensure every question is contained in the training set so for every response in the test set, our models have seen scored response examples from the exact same question in the training set.

For our approach we use MathBERT [35] as the pre-trained model with 110M parameters as the base scoring model [3]. We use the Adam optimizer, a batch size of 16, a learning rate of 1e-5 for 5 epochs on an NVIDIA RTX 8000 GPU. We do not perform any hyper-parameter tuning and simply use the default setting. For each training response, we randomly sample one in-context example per score class and fill up with the rest of training examples up to 25 in total from the training dataset for the corresponding question. Due to the restriction on input length for language models (512 for MathBERT), we truncate an example to a maximum of 70 tokens if necessary to ensure that the question, the target response to score, and all examples all fit in. For testing, we repeat the process of randomly sampling examples eight times for each target student response to be scored and average the predicted score class probabilities.

We use an evaluation setting that follows from the one used in [14], for a fair comparison to compare it with SBERT-Canberra (SBERT-C) [5], the current state-of-the-art method. The evaluation utilizes a 2-parameter Rasch model [44]; We include three groups of terms as covariates in the Rasch model: i) the student ability and question difficulty parameters, ii) the score category predictive probabilities according to the trained scoring method, and iii) the number of words in the response. After training the scoring model, we use the predicted scoring probabilities to learn regression coefficients and the ability/difficulty parameters. Intuitively, this evaluation setup studies how textual information in open-ended responses help *in addition to* student ability and question difficulty during scoring; its purpose is not to evaluate how accurately response scoring models are themselves.

For this evaluation, we use Problem ID as input for each training response to help the model adapt to the task. We do not use question text as input since $D_{orig}$ contains many (709 out of 2,042) question texts that have no semantic meaning (e.g., "For Page 100 question b, answer the question"). This noisy question text cannot help the model recognize different questions and may confuse the model.

### 4.3.2 Results and Analysis

Table 3 shows the average value for all metrics across the 10 folds for our method (Meta In-context), the SBERT-C baseline, and other baselines studied in [5]. We see that our method is able to achieve a 0.02 (or 4.2%) improvement over the best performing baseline, SBERT-C, on the most important metric, Kappa, while also outperforming on the other two metrics with smaller margins. This improvement validates the effectiveness of our overall method and further pushes the boundary on math ASAG. This improvement is more significant on the cleaned dataset $D_{clean}$, which we use for further evaluation next. We further note that there is a discrepancy between metric values (high AUC, low Kappa) on this experiment compared to other experiments due to the Rasch model-based setup.

### 4.3.3 Ablation Study

We conduct an ablation study to verify the effectiveness of each component of our scoring method: using question text as input vs. using only question ID as input, adding textual

[3] `https://huggingface.co/tbs17/MathBERT`

**Table 2: Example questions, student responses, and scores in the dataset.**

| $q_{id}$ : question unique id | $q_{text}$ : question text | $x$ : student response | $y$ : teacher grade |
|---|---|---|---|
| 112348 | Write a function rule and a recursive rule for a line that contains the points (-4, 11), (5, -7), and (7, -11) | Don't know what a recursive rule is | 0 |
| 32147 | Ryan had $800 of his summer job earnings remaining when school started. He plans to use this amount as spending money throughout the 10 months of his school year. please indicate the 3 most important words/phrases in the question | The 3 most important words or phrases in the question are $800, 10, and months. | 4 |
| 32149 | Ryan will divide the $800 into 10 equal amounts of $80. If he completely spends $80 during each month of his school year, how much of his earnings will remain at the end of the third month of his school year? Explain how you got your answer. | he will have $560 left. 800-240=560 | 4 |

**Table 3: Evaluation results using the same dataset and under the same evaluation setting as [14, 5] show that our scoring method outperforms existing methods.**

| Model | AUC | RMSE | Kappa |
|---|---|---|---|
| Rasch* + Meta In-context (ours) | **0.861** | **0.541** | **0.496** |
| Rasch* + SBERT-Canberra | 0.856 | 0.577 | 0.476 |
| Baseline Rasch | 0.827 | 0.709 | 0.370 |
| Rasch + Number of Words | 0.825 | 0.696 | 0.382 |
| Rasch* + Random Forest | 0.850 | 0.615 | 0.430 |
| Rasch* + XGBoost | 0.832 | 0.679 | 0.390 |
| Rasch* + LSTM | 0.841 | 0.637 | 0.415 |

instructions to provide information on the scoring scale, using scored examples to provide additional context, and using MathBERT as the base language model to fine-tune vs. using BERT. For this evaluation, we use the cleaned dataset $D_{clean}$ and a different experimental setting to directly evaluate the scoring accuracy of ASAG methods without using the Rasch model. The rest of the experimental settings, from cross-validation to model training, remain the same. Table 4 shows the results for all variants of our approach on all three metrics. We see that removing question text, textual instructions on scoring scale, and scored examples as input all result in significant degradation in scoring accuracy on some (or all) metrics. Specifically, removing scored examples results in the most significant drop in scoring accuracy, by around 0.02 in Kappa; this result validates the effectiveness that providing in-context examples can significantly benefit language models by helping them adapt to the current task (question). This result clearly validates our intuition that in-context examples reduce the difficulty of the AS task by changing the nature of the task from scoring to finding similar responses, which is easier. Removing question text also results in a (less significant) accuracy drop off: this result directly contradicts our observations in the previous experiment using the original dataset in [5, 14] where we found that inputting the question text results in worse performance than inputting only the question ID. The likely reason for this result is that the cleaned dataset $D_{clean}$ contains much more questions that are semantically meaningful, which are helpful to include in the scoring method to provide important information on the scoring task.

A surprising but important result of this experiment is that using MathBERT results in a small drop off in performance (0.015 on Kappa, 0.007 on RMSE, and a 0.002 improvement on AUC) compared to using BERT. This observation is counter-intuitive since MathBERT is specifically designed to handle math expressions and trained on mathematical content, while BERT is not. To further examine why MathBERT underwhelms on the scoring task, we further investigate its performance on subsets of responses divided according to how much math information is contained in them. Specifically, we divided responses in the test set into two groups according to the amount of mathematical expressions involved: $D_{math}$ that contains responses where more than half of the tokens in the response are mathematical tokens and $D_{text}$ that contains the rest of the responses. Table 5 shows scoring accuracy for our approach using MathBERT and BERT as the base language model on these different response subsets. We see that on responses that are primarily textual, BERT outperforms MathBERT, which suggests that MathBERT loses some ability to encode textual information. On responses that are primarily mathematical, MathBERT performs similarly to BERT on RMSE and Kappa while outperforming BERT on AUC. This result suggests that MathBERT may have some benefit in handling mathematical tokens but the advantage may be minimal. Therefore, an important avenue for future work is to develop language models that are capable of representing and understanding mathematical content.

## 4.4 Scoring new questions

### *4.4.1 Experimental Setting*

For this experiment, we focus on testing the performance of our approach on generalizing to new questions (tasks) without seeing scored examples and how quickly our approach can adapt to them using few examples. Therefore, we randomly divide all questions in $D_{clean}$ into 5 equally-sized folds in terms of the number of questions instead of the number of responses. As a result, the number of responses in each fold may vary $(26,229 \pm 689)$ since the number of student responses to each question is different. For each run, we use 4 folds for training and 1 fold for testing.

On the test set, we make $n \in \{0, 1, 3, 5, 7, 10, 25, 50, 80\}$

**Table 4: Ablation results for different design components of our method on $D_{clean}$. Most components contribute significantly.**

| Method Component | | | | | Metric | | |
|---|---|---|---|---|---|---|---|
| Question Text | Question ID | Scale | Example | MathBERT | AUC | RMSE | Kappa |
| ✓ | | ✓ | ✓ | ✓ | **0.733** ±0.006 | 1.077 ±0.002 | 0.589 ±0.004 |
| | ✓ | ✓ | ✓ | ✓ | 0.724 ±0.007 | 1.083 ±0.003 | 0.585 ±0.006 |
| ✓ | | ✓ | | ✓ | 0.710 ±0.006 | 1.278 ±0.002 | 0.568 ±0.004 |
| | | ✓ | ✓ | ✓ | 0.720 ±0.008 | 1.088 ±0.001 | 0.583 ±0.009 |
| ✓ | | | ✓ | ✓ | 0.719 ±0.008 | 1.091 ±0.003 | 0.582 ±0.005 |
| ✓ | | ✓ | ✓ | | 0.731 ±0.007 | **1.051** ±0.004 | **0.604** ±0.010 |

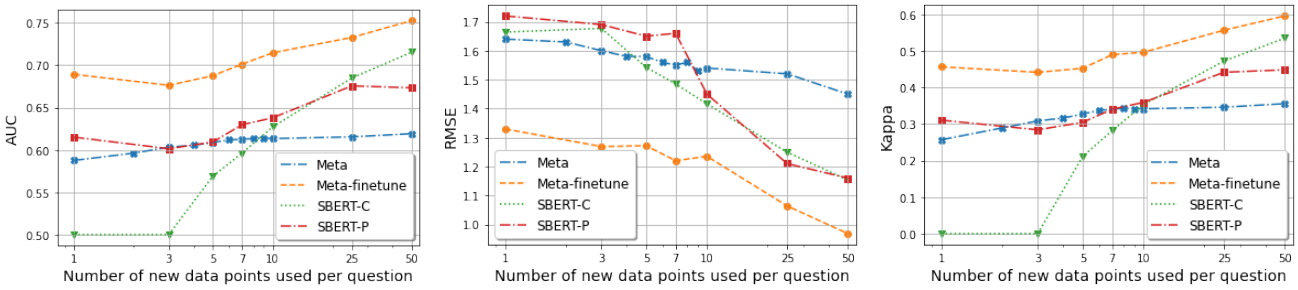


**Figure 2: Results on generalizing to previously unseen questions using a few scored examples on all three metrics. Our approach, Meta-finetune, consistently outperforms SBERT-C and SBERT-P. Even without adjusting the model and using the scored examples as input (Meta), we outperform SBERT-C when the number of examples is small.**

**Table 5: Scoring accuracy on responses that contain more mathematical tokens vs. more text tokens.**

| Data | approach | AUC | RMSE | Kappa |
|---|---|---|---|---|
| D_math | MathBERT | **0.755** ±0.008 | **0.587** ±0.003 | 0.690±0.008 |
| | BERT | 0.741 ±0.010 | 0.610 ±0.009 | **0.691** ±0.020 |
| D_text | MathBERT | 0.713 ±0.006 | 1.022 ±0.004 | 0.523 ±0.006 |
| | BERT | **0.716** ±0.006 | **1.001** ±0.003 | **0.542** ±0.008 |

scored responses per question available to methods trained on the training dataset and evaluate their ability to score other responses. We emphasize that there is no overlap between training responses and test responses for these previously unseen questions. We use two settings for our method. For the first setting, **Meta**, we do not further adjust the trained scoring model; instead, we only feed these responses and their scores, i.e., in-context examples, to the trained scoring model. For cases where $n < 25$, we only feed in $n$ examples even though the method was trained with 25 examples. For cases where $n > 25$, we follow randomly sample 25 examples from the $n$ total examples as input, following the same setting above. This experimental setting can be seen as "*zero-shot*" learning where we directly test how a scoring method trained on other questions works on new questions without observing any scored responses.

For the second setting, **Meta-finetune**, we further fine-tune our trained method on the $n$ new scored responses per question. During this process, for each response as the scoring target, we use the other $n-1$ responses as in-context examples. This experimental setting can be seen as "*few-shot*" learning where we test how quickly a scoring method trained on other questions can adapt to new questions.

Since **SBERT-C** is the current state-of-the-art math ASAG method on this dataset, we use it as our baseline. According to [5], it calculates similarities between the target response and other responses to the same question. Then it picks the score of the response with the highest similarity to the target response as its prediction, which means that it is not capable of zero-shot generalization to new questions. Therefore, we use $n$ scored examples on these new questions to train the scoring method and evaluate on the other responses. We emphasize again that in both the zero-shot and few-shot settings, the scored examples are excluded from performance valuations.

We also use an additional baseline [12], which we refer to as **SBERT-P**. This method uses SBERT to encode responses and questions and feed the resulting representations to a classifier for predictions. This method also trains a single unified model across and is thus capable of zero-shot generalization to previously unseen questions. We use $n$ scored examples on these new questions for SBERT-P to train on to evaluate it in the few-shot learning setting.

### 4.4.2 *Result and Analysis*

Table 6 shows the experimental results averaged over all folds. We see that Meta-finetune outperforms the other three approaches on all values of $n$ for all metrics, achieving satisfactory results of AUC = 0.689, RMSE = 1.329 and Kappa = 0.456 in the one-shot learning setting ($n = 1$), significantly outperforming Meta, SBERT-P and SBERT-C (by up to 50% on Kappa). The performance of Meta-finetune stabilizes as $n$ increases and still outperforms SBERT-C (0.03 on AUC, 0.154 on RMSE and 0.055 on Kappa) and SBERT-P (0.11 on AUC, 0.161 on RMSE and 0.113 on Kappa) at $n = 80$. These results clearly demonstrate that, compared to SBERT-C and SBERT-P, our method is highly

**Table 6: Scoring accuracy for different methods on generalization to new questions not seen during training, using a small number of scored examples.**

| num-of new-data points / question | Method | AUC | RMSE | KAPPA |
|---|---|---|---|---|
| 0 | Meta | $0.533 \pm 0.017$ | $1.650 \pm 0.020$ | $0.100 \pm 0.052$ |
| | SBERT-P | $0.558 \pm 0.006$ | $1.931 \pm 0.001$ | $0.170 \pm 0.013$ |
| | SBERT-C | – | – | – |
| 1 | Meta | $0.588 \pm 0.012$ | $1.641 \pm 0.013$ | $0.257 \pm 0.041$ |
| | Meta-finetune | $0.689 \pm 0.033$ | $1.329 \pm 0.009$ | $0.456 \pm 0.048$ |
| | SBERT-P | $0.615 \pm 0.022$ | $1.721 \pm 0.011$ | $0.310 \pm 0.043$ |
| | SBERT-C | $0.500 \pm 0.001$ | $1.664 \pm 0.009$ | $0.000 \pm 0.001$ |
| 3 | Meta | $0.606 \pm 0.012$ | $1.620 \pm 0.013$ | $0.308 \pm 0.041$ |
| | Meta-finetune | $0.676 \pm 0.010$ | $1.269 \pm 0.010$ | $0.441 \pm 0.017$ |
| | SBERT-P | $0.601 \pm 0.040$ | $1.691 \pm 0.010$ | $0.284 \pm 0.071$ |
| | SBERT-C | $0.501 \pm 0.001$ | $1.677 \pm 0.009$ | $0.000 \pm 0.001$ |
| 5 | Meta | $0.589 \pm 0.013$ | $1.581 \pm 0.013$ | $0.289 \pm 0.043$ |
| | Meta-finetune | $0.688 \pm 0.009$ | $1.272 \pm 0.013$ | $0.452 \pm 0.021$ |
| | SBERT-P | $0.610 \pm 0.028$ | $1.650 \pm 0.010$ | $0.284 \pm 0.050$ |
| | SBERT-C | $0.569 \pm 0.061$ | $1.543 \pm 0.080$ | $0.211 \pm 0.016$ |
| 7 | Meta | $0.611 \pm 0.011$ | $1.548 \pm 0.011$ | $0.341 \pm 0.040$ |
| | Meta-finetune | $0.701 \pm 0.010$ | $1.220 \pm 0.008$ | $0.489 \pm 0.022$ |
| | SBERT-P | $0.630 \pm 0.037$ | $1.662 \pm 0.012$ | $0.340 \pm 0.064$ |
| | SBERT-C | $0.569 \pm 0.006$ | $1.485 \pm 0.011$ | $0.282 \pm 0.019$ |
| 10 | Meta | $0.614 \pm 0.010$ | $1.543 \pm 0.013$ | $0.342 \pm 0.043$ |
| | Meta-finetune | $0.716 \pm 0.008$ | $1.235 \pm 0.009$ | $0.496 \pm 0.021$ |
| | SBERT-P | $0.638 \pm 0.031$ | $1.453 \pm 0.018$ | $0.359 \pm 0.080$ |
| | SBERT-C | $0.627 \pm 0.008$ | $1.416 \pm 0.009$ | $0.353 \pm 0.019$ |
| 80 | Meta | $0.626 \pm 0.024$ | $1.550 \pm 0.016$ | $0.373 \pm 0.074$ |
| | Meta-finetune | $0.765 \pm 0.010$ | $0.940 \pm 0.015$ | $0.636 \pm 0.042$ |
| | SBERT-P | $0.704 \pm 0.033$ | $1.101 \pm 0.011$ | $0.523 \pm 0.020$ |
| | SBERT-C | $0.735 \pm 0.017$ | $1.094 \pm 0.008$ | $0.581 \pm 0.042$ |

effective at "warm-starting" scoring models on new questions since it is able to get a sense of how responses should be scored from scored responses to other questions. Again, we note that in-context examples changes the nature of the task from AS to finding similar responses; as a result, models can learn this task quicker and adapt to new questions using only a few examples.

SBERT-C, on the other hand, can barely work in few-shot learning settings, i.e., $n \in \{1, 3\}$. This observation is not surprising since SBERT-C does learn a scoring model from scratch and cannot work when the number of training data points is less than the number of possible score categories. The performance of SBERT-C starts to gradually increase when $n > 5$ but is still significantly worse than Meta-finetune.

Meta, the method for zero-shot learning, although fails to generalize well (only 0.533 in AUC and 0.1 in Kappa at $n = 0$) without seeing any training data, still significantly outperforms SBERT-C with $n \in \{1, 3\}$ and performs similarly to SBERT-P. This advantage only disappears at $n = 10$. To further illustrate this difference, we plot the three metrics vs. $n$ for all methods in Figure 2. We see that Meta's AUC and Kappa values are higher than that for SBERT-C until $n$ reaches around 8, which indicates that even without re-training the model, it is more suitable for few-shot learning than SBERT-C on new questions.

## 4.5 Qualitative Error Analysis

In this section, we qualitatively analyze the prediction errors made by our ASAG method. We use the model trained on $D_{clean}$, with problem text + scale + examples as input into MathBERT for our analysis.

### 4.5.1 *Feature analysis*

To analyze the difference between correct predictions and incorrect predictions, we extract several features that capture properties of the questions and responses to better understand the strengths and weaknesses of the trained ASAG method. As shown in Table 7, "Response math tokens" represents the percentage of math tokens in a response; "Response contains img/table" represents whether a response has images or tables; "Response length" represents the number of tokens a response; "score" represents the actual score given by the graders; "Number of graders" represents the number of graders that graded each response to the question; "question length" represents the number of tokens the corresponding question has and "question math tokens" represents the percentage of math tokens in the question.

**Table 7: Features analysis between correct predictions and incorrect predictions. * means the difference is significant ($p\_value < 0.005$).**

| Features (avg.) | Correct Prediction | Incorrect Prediction |
|---|---|---|
| Response math tokens (%)* | 30.6 | 25.1 |
| Response contain img/table (%)* | 1.29 | 2.88 |
| Response length* | 17.4 | 29.5 |
| Score* | 3.25 | 2.13 |
| Number of graders | 2.53 | 2.48 |
| Question length* | 37.1 | 39.1 |
| Question math tokens (%) | 8.12 | 7.31 |

We observe a significant difference ($p\_value < 0.005$) between values of the correct predictions and values of the incorrect predictions. We make the following observations:

- The scoring method is more accurate at scoring responses with higher percentage of math tokens and it becomes less accurate when there are higher percentage of plain texts.
- The scoring method is more accurate when the response contains images or tables that words can not represent.
- The scoring method is more accurate at scoring shorter responses.
- The scoring method is more accurate at scoring responses with shorter question description.
- The scoring method is more accurate when the average score of the response is higher. This observation indicates that the model is better at scoring responses with higher quality.
- There is no obvious distinction in grading accuracy for responses with different numbers of graders or to questions with different math tokens percentages.

### 4.5.2 *Question topic and type error analysis*

Table 8 lists the summarization of scoring accuracy on different question topics and types. We extract the topics and types from question text using BERTopic [20]. BERTopic is a topic modeling technique that leverages transformers

and term and document frequencies [33] to create easily interpretable topics. Overall, we see that the trained scoring method has better Kappa scores on questions that are primary text-based or involve equations. The result is not surprising since we adapted MathBERT, which likely sees many text-based questions during its pre-training stage. Questions that require students draw graphs in their response also have high Kappa scores; however this result is mainly due to the fact that most of these responses are given full credit, making them easy for scoring methods to make predictions. On the other hand, the trained scoring method has worse Kappa scores on estimation-type and (only a few) multiple-choices questions. This observation can be explained by language models not being trained to capture number sense and thus struggle at numerical reasoning [**?**]. For multiple-choice questions, the response, i.e., the multiple-choice option, is semantically meaningless, which does not provide meaningful context to the scoring method.

**Table 8: Scoring accuracy on different question topics and types. Results are shown in increasing order of the Kappa score. * means the score is better than the average across all responses.**

| Topic | Type | AUC | RMSE | Kappa |
|---|---|---|---|---|
| Misc. | Multiple-choice | 0.631 | 1.472 | 0.400 |
| Math | Table calculation | 0.659 | 1.345 | 0.445 |
| Algebra | Estimation | 0.702 | 1.310 | 0.536 |
| Calculus | Estimation | 0.716 | 1.241 | 0.546 |
| Algebra | Table creation | 0.731 | 0.823* | 0.606* |
| Algebra | Equation writing | 0.732 | 1.023 | 0.612* |
| Algebra | Graph drawing | 0.734* | 0.725* | 0.629* |
| Math | Word question | 0.735* | 0.663* | 0.647* |
| Calculus | Graph drawing | 0.736* | 0.610* | 0.758* |

### *4.5.3 Error type analysis*

For this analysis, we choose a question with scoring accuracy below the average on our dataset to analyze the types of errors made by our trained scoring method. Table 9 shows selected responses with erroneous score predictions and the types of these errors. The question asks students to write an equation with a popular correct response $15/3 = 5$. We make the following observations on typical error types (apart from some obvious human grader errors, which we omit):

- The first error type indicates that our trained scoring method can still struggle on mathematical reasoning and handling numerical tokens. The incorrect responses "15*5=3" and "5/3=15" have the same numerical tokens but with different ordering and an incorrect operator token compared to the correct response, which completely changes their meaning. The trained scoring method tends to overestimate their scores. This observation suggests that we need base language models with stronger numerical reasoning abilities.
- The second error type indicates that our trained scoring method can struggle with spelling errors in student responses. When the word "equals" is spelled incorrectly in a student response, it does not affect the human grader's ability to understand the student's intention. However, the trained scoring method puts a penalty on this spelling error.
- The third error type indicates that our trained scoring method may not recognize paraphrased responses. As shown in the examples, student may add text such as "I think that it is" which does not alter the meaning of the response; however, it adds noise and misled the prediction.
- The fourth error type indicates that our trained scoring method cannot handle responses in unparsable format such as an attachment.

## 5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a language model fine-tuning-based method for automatic short answer grading for open-ended, short-answer math questions. Our method has two main components: a base MathBERT model pre-trained with educational content on math subjects, and a meta-learning-based, in-context fine-tuning method that promotes generalization to new questions with a carefully designed input format. Experimental results on a large real-world student response dataset revealed surprisingly contradicting findings: Using MathBERT instead of regular BERT, which is not trained on mathematical content, results in a decrease in scoring accuracy, while the in-context fine-tuning method results in significantly improved scoring accuracy compared to existing methods, especially on previously unseen questions.

There are plenty of avenues for future work. First, the observation that MathBERT [35] cannot outperform BERT as the base language model suggests that there is a need to develop more effective models for mathematical language. One promising direction is perhaps taken by another simultaneously proposed version of MathBERT [31] that leverages the inherent tree structure of mathematical expressions. Moreover, the noisiness of human grading that we observed in our experiments suggests that there is a need to develop ASAG methods that take inter-rater agreement into account [43].

Second, there is a need to further improve the completeness of the context information we provide to the base language model. Several possible sources of additional contextual information include the grade level of the question, the common core standard codes, and mathematical skill/concept tags, which can all provide information on the level of the question. Additionally, we may even directly incorporate relevant mathematical content into the model's input, e.g., by retrieving content chunks in textbooks or online resources using information retrieval methods [9]. However, a potential challenge that needs to be resolved is how to concisely pack all relevant contextual information into the model without exceeding the input length limit of language models (usually 512 tokens).

Third, in order to make ASAG methods more applicable in realworld educational scenarios, there is a need to thoroughly study the fairness aspects of these methods and ensure all students are treated fairly. There is a need to investigate how ASAG methods performs on different student populations; recent work has raised the concern that it is not clear that whether one should explicitly incorporate student

**Table 9: Examples of scoring errors made by our trained method.**

| **Question:** Chelsea collects butterfly stickers. The picture shows how she placed them. Write a division sentence to show how she equally grouped her stickers. _÷ _ = _ **Most frequency correct response :** 15/3=5 | | | |
|---|---|---|---|
| Error type | Response | Grade | Predict |
| Poor reasoning on math operator and numerical token | 15* 5=3 | 2 | 4 |
| | 5/3=15 | 0 | 2 |
| | 15_3=12 | 1 | 2 |
| Spelling error | 3 times 5 eques 15 | 4 | 2 |
| Confused by paraphrased responses | I think that it is 5x3=15 | 4 | 1 |
| | she place them like in 3 groups and she even did the answer but she did not new the each group | 2 | 0 |
| Meaningless response | see attachment | 3 | 0 |

demographic information during model training [46]. Future work should explore how to incorporate fairness regularization into the training objective to promote methods that are fair across students [2, 34, 47, 48, 25].

## 6. ACKNOWLEDGEMENT

The authors would like to thank the National Science Foundation for their support through grant IIS-2118706.

## 7. REFERENCES

[1] The hewlett foundation: Automated essay scoring. Online: `https://www.kaggle.com/c/asap-aes`, 2021.

[2] A. Agarwal, A. Beygelzimer, M. Dudík, J. Langford, and H. Wallach. A reductions approach to fair classification. *arXiv preprint arXiv:1803.02453*, 2018.

[3] M. Ariely, T. Nazaretsky, and G. Alexandron. Machine learning and hebrew nlp for automated assessment of open-ended questions in biology. *International Journal of Artificial Intelligence in Education*, 1:34, 2022.

[4] Y. Attali and J. Burstein. Automated essay scoring with e-rater® v. 2. *The Journal of Technology, Learning and Assessment*, 4:3, 2006.

[5] S. Baral, A. F. Botelho, J. A. Erickson, P. Benachamardi, and N. T. Heffernan. Improving automated scoring of student open responses in mathematics. *International Educational Data Mining Society*, 2021.

[6] S. Basu, C. Jacobs, and L. Vanderwende. Powergrading: a clustering approach to amplify human effort for short answer grading. *Transactions of the Association for Computational Linguistics*, 1:391–402, 2013.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] J. Burstein. The e-rater® scoring engine: Automated essay scoring with natural language processing. 2003.

[9] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. In *Annual Meeting of the Association for Computational Linguistics*, pages 1870–1879, 2017.

[10] H. Chen and B. He. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752, 2013.

[11] Y. Chen, R. Zhong, S. Zha, G. Karypis, and H. He. Meta-learning via language model in-context tuning. *arXiv preprint arXiv:2110.07814*, 2021.

[12] A. Condor, M. Litster, and Z. Pardos. Automatic short answer grading with sbert on out-of-sample questions. *International Educational Data Mining Society*, 2021.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[14] J. A. Erickson, A. F. Botelho, S. McAteer, A. Varatharaj, and N. T. Heffernan. The automated grading of student open responses in mathematics. In *Proceedings of the Tenth International Conference on Learning Analytics  Knowledge*, LAK '20, page 615–624, New York, NY, USA, 2020. Association for Computing Machinery.

[15] N. Fernandez, A. Ghosh, N. Liu, Z. Wang, B. Choffin, R. G. Baraniuk, and A. S. Lan. Automated scoring for reading comprehension via in-context bert tuning. In *International Conference on Artificial Intelligence in Education*, page 0, 2022.

[16] P. W. Foltz, D. Laham, and T. K. Landauer. The intelligent essay assessor: Applications to educational technology. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 1(2):939–944, 1999.

[17] M. Fowler, B. Chen, S. Azad, M. West, and C. Zilles. Autograding" explain in plain english" questions using nlp. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, pages 1163–1169, 2021.

[18] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[19] A. C. Graesser, D. S. McNamara, M. M. Louwerse, and Z. Cai. Coh-metrix: Analysis of text on cohesion and language. *Behavior research methods, instruments, & computers*, 36(2):193–202, 2004.

[20] M. Grootendorst. Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics., 2020.

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

[22] A. S. Lan, D. Vats, A. E. Waters, and R. 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, 2015.

[23] C. Leacock and M. Chodorow. C-rater: Automated scoring of short-answer questions. *Computers and the Humanities*, 37(4):389–405, 2003.

[24] S. Lottridge, B. Godek, A. Jafari, and M. Patel. Comparing the robustness of deep learning and classical automated scoring approaches to gaming strategies. Technical report, Cambium Assessment Inc., 2021.

[25] N. Madnani, A. Loukina, A. Von Davier, J. Burstein, and A. Cahill. Building better open-source tools to support fairness in automated scoring. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pages 41–52, 2017.

[26] E. Mayfield and A. W. Black. Should you fine-tune bert for automated essay scoring? In *15th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 151–162, 2020.

[27] D. S. McNamara, S. A. Crossley, R. D. Roscoe, L. K. Allen, and J. Dai. A hierarchical classification approach to automated essay scoring. *Assessing Writing*, 23:35–59, 2015.

[28] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*, 2021.

[29] R. H. Nehm, M. Ha, and E. Mayfield. Transforming biology assessment with machine learning: automated scoring of written evolutionary explanations. *Journal of Science Education and Technology*, 21(1):183–196, 2012.

[30] E. B. Page. The imminence of... grading essays by computer. *The Phi Delta Kappan*, 47(5):238–243, 1966.

[31] S. Peng, K. Yuan, L. Gao, and Z. Tang. Mathbert: A pre-trained model for mathematical formula understanding. *arXiv preprint arXiv:2105.00377*, 2021.

[32] I. Persing and V. Ng. Modeling prompt adherence in student essays. In *52nd Annual Meeting of the Association for Computational Linguistics*, pages 1534–1543, 2014.

[33] A. Rajaraman and J. D. Ullman. *Data Mining*, page 1–17. Cambridge University Press, 2011.

[34] C. Russell, M. J. Kusner, J. Loftus, and R. Silva. When worlds collide: integrating different counterfactual assumptions in fairness. In *Advances in Neural Information Processing Systems*, pages 6414–6423, 2017.

[35] J. T. Shen, M. Yamashita, E. Prihar, N. Heffernan, X. Wu, B. Graff, and D. Lee. Mathbert: A pre-trained language model for general nlp tasks in mathematics education. *arXiv preprint arXiv:2106.07340*, 2021.

[36] S. Srikant and V. Aggarwal. A system to grade computer programming skills using machine learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1887–1896, 2014.

[37] C. Sung, T. Dhamecha, S. Saha, T. Ma, V. Reddy, and R. Arora. Pre-training bert on domain resources for short answer grading. In *Conference on Empirical Methods in Natural Language Processing*, pages 6071–6075, 2019.

[38] K. Taghipour and H. T. Ng. A neural approach to automated essay scoring. In *Empirical methods in natural language processing*, pages 1882–1891, 2016.

[39] M. Uto and Y. Uchida. Automated short-answer grading using deep neural networks and item response theory. In *International Conference on Artificial Intelligence in Education*, pages 334–339, 2020.

[40] M. Uto, Y. Xie, and M. Ueno. Neural automated essay scoring incorporating handcrafted features. In *28th Conference on Computational Linguistics*, pages 6077–6088, 2020.

[41] S. Valenti, F. Neri, and A. Cucchiarelli. An overview of current research on automated essay grading. *Journal of Information Technology Education: Research*, 2(1):319–330, 2003.

[42] Z. Wang, M. Zhang, R. G. Baraniuk, and A. S. Lan. Scientific formula retrieval via tree embeddings. In *2021 IEEE International Conference on Big Data*, pages 1493–1503, 2021.

[43] J. Whitehill, T.-f. Wu, J. Bergsma, J. Movellan, and P. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in neural information processing systems*, 22, 2009.

[44] B. D. Wright. Solving measurement problems with the rasch model. *Journal of educational measurement*, pages 97–116, 1977.

[45] R. Yang, J. Cao, Z. Wen, Y. Wu, and X. He. Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking. *Findings of the Association for Computational Linguistics: EMNLP*, 2020:1560–1569, 2020.

[46] R. Yu, H. Lee, and R. F. Kizilcec. Should college dropout prediction models include protected attributes? In *8th ACM Conference on Learning@ Scale*, pages 91–100, 2021.

[47] M. B. Zafar, I. Valera, M. Gomez Rodriguez, and K. P. Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International Conference on World Wide Web*, pages 1171–1180, 2017.

[48] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.

[49] X. Zhai, Y. Yin, J. W. Pellegrino, K. C. Haudek, and L. Shi. Applying machine learning in science assessment: a systematic review. *Studies in Science Education*, 56(1):111–151, 2020.

[50] M. Zhang, Z. Wang, R. Baraniuk, and A. Lan. Math operation embeddings for open-ended solution analysis and feedback. In *Proc. International Conference on Educational Data Mining*, pages 216–227, June 2021.

[51] Y. Zhang, R. Shah, and M. Chi. Deep learning+ student modeling+ clustering: A recipe for effective automatic short answer grading. In *International Conference on Educational Data Mining*, page 562, 2016.