ASAGeR: Automated Short Answer Grading Regressor via Sentence Simplification

1st Mohammad Iqbal
Department of Mathematics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
iqbal@matematika.its.ac.id or 0000-0003-1115-3486

3rd Rosita Laili Udhiah
Department of Mathematics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
rositalaili22@gmail.com

2nd Tsamarah Rana Nugraha
Department of Mathematics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
tsamarahrana@gmail.com

4th Hsing-Kuo Pao
Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology
Taipei, Taiwan
pao@mail.ntust.edu.tw or 0000-0002-5518-9475

***Abstract*—We propose an automated short answer grading system (ASAG) to estimate the student answer scores via text summarization from LLMs. The step of text summarization provides enough question answer normalization so that the summarized answers have the answer keys well organized and the grading based on that should be more accurate and easier than before, no matter the answers are graded by human or automatic graders. On the other hand, we also discuss the scenario when more than one grader are involved in the grading but providing inconsistent scores. We adopt a majority voting mechanism to overcome such difficulty and produce superior result in average. Overall the proposed methodology has its evaluation done to show the superiority to other state-of-the-art methods. The pre-trained transformer version 3.5 (GPT 3.5) is used to serve the text summarization tool given a well-designed prompt.**

***Index Terms*—Natural Language Processing, Answer Grading, Regression, Text Simplification.**

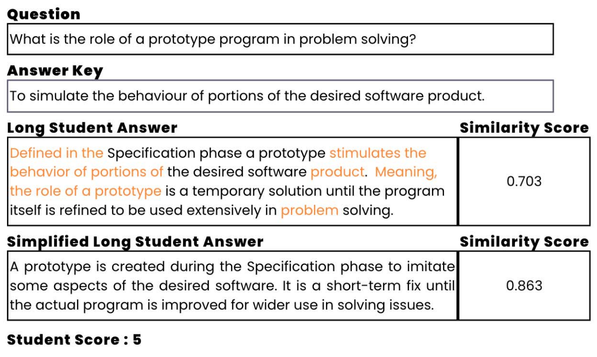
I. Introduction

Online teaching often attracts a huge amount of participating students. One of the bottleneck for offering such courses is usually on the student evaluation for their performance. Evaluating students' performance from assignments and examinations is a paramount task for teachers when deciding students' final grades [1]. By default, teachers also can infer the student's capabilities and comprehension of the course materials through hiring teaching assistants. Apparently, human efforts are not as efficient as automatic agents, so called automated grading systems (AGSs) where tasks can be accomplished with very few delay following a pre-defined grading guidelines. Given the experience of the covid epidemic, the AGSs have attracted course instructors to assist them to achieve the goal of teaching experience with scalable participants. In this work, we propose an automatic grading system given the availability of Large Language Model (LLM) and GPT tools.

The recent progress of LLM and generative models rewrite history of natural language processing since the end of 2022, which includes the plausible breaking of Turing test (e.g. [2]) thanking the tremendous efforts and facility of computation power. The well-known GPT 3.5 and GPT 4-based language models achieve superior performance than top professional members or students [3]. Given the aforementioned progress, we would like to leverage what has been achieved to AGSs. That is, we consider the assistance from LLM or generative models and apply it to the grading tasks for the possibility of building an automatic system. As long as the availability of LLM and GPT tools becomes more and more ubiquitous, we can have them included as part of the AGS system and hope the system based on them can perform superior to other existing approaches.

Given an exam, we usually have questions like multiple-choice (MC), short-answer (SA), and essay (ESS) to answer. Intuitively, MC is more straightforward than other types in terms of grading, yet it may not be perfect for teachers who aim for student assessment [4], [5]. Compared to that, SA and ESS questions provide more informative for teachers to understand students' understanding and progress on certain topics; however, grading SAs and ESSs could be challenging due to the bias of the student answers [6]. Student answers to SA or ESS questions can be with diverse "styles". We focus on finding assistance for SA and ESS grading. Fig. 1 shows a few student answer examples.

There are plenty of writing styles when the exam questions ask for a few sentences for the answers. The answers of students could be wordy or too short is not the main problems. They could be either *incomplete* to cover all key parts, could be *confusing* or *ambiguous*. On the opposite, the answers could be too wordy due to lack of confidence from students. When replying long answers to a question, some irrelevant parts could also be included. Therefore, grading on those long answers takes longer time than grading the normal cases. The third situation is that the sentences contain too many grammatical or other errors so that the readability of the answers is too low for teacher to grasp what the real meaning



Question

What is the role of a prototype program in problem solving?

Answer Key

To simulate the behaviour of portions of the desired software product.

Long Student Answer	**Similarity Score**
Defined in the Specification phase a prototype stimulates the behavior of portions of the desired software product. Meaning, the role of a prototype is a temporary solution until the program itself is refined to be used extensively in problem solving.	0.703

Simplified Long Student Answer	**Similarity Score**
A prototype is created during the Specification phase to imitate some aspects of the desired software. It is a short-term fix until the actual program is improved for wider use in solving issues.	0.863

Student Score : 5

Fig. 1. Student answers and their evaluation result if based on similarity score.

of the answers. For the second and third cases, we can have a summarization tool to assist teachers to pick up the key components from the whole set for teachers to grade, but with efficient efforts. That is the goal which explains why we can use LLM and generative models, served as the summarization tool to conquer the tasks. In this work, we focus on a set of specific questions as the grading task. We attempt to propose an automated SA grading tool in a regression fashion via sentence simplification, called ASAGeR to accomplish the task. We aim to help the teacher (or regressor) comprehend the student's answer easily by an answer simplification procedure, so that the simplified answer is close enough to the answer key and the grading task can be easily completed.

Let us further consider the two AG systems: Automatic Short-Answer Grader (ASAG) for SA and Automated Essay Grader (AEG) for ESS grading, respectively. Two may own different aspects in their evaluation. For instance, the ASAG systems are focused on evaluating factual answers and prioritizing content assessment over writing style [7]. On the other hand, the AEG rather than the ASAG systems pay more attention to the text-related features, such as text length and word counts [8], or grammar, coherence, and diction [9]. These are so-called text-based quantitative features. That is, AEG concentrates on writing styles as well as the writing content. According to Science, Technology, and Mathematics (STEM) learning, the assessment of content covered in ASAG holds greater importance compared to the assessment of grammar in AEG [10]. Following the above rationale, we explore ASAG in this work.

Most students tend to answer questions with long description[1]. Many of them may believe that long answers may lead to high grades. To reply SA questions, the tendency of writing long answers however can introduce nightmare to teachers' evaluation. For instance, teachers may be confused by a quick glance of students' answer if the correct answer key exists in the students' answer, but looks too sparse to be picked up from other part of the answer by teachers in the evaluation. Even if the long answer basically shows its correctness, but still can contain a great redundancy where the evaluation on it also takes longer time than the one on normal cases.

[1]We will discuss how students often give lengthy answers with more details in Section IV.

If on the other hand, an automatic approach is considered for the evaluation, sparseness can lead to underestimation of the answer's correctness, if for instance, a cosine similarity or similar measure is used for the matching-type of evaluation between the student's answer and the answer key. After all, we aim to work on the evaluation of long answers, in particular, for SA questions by the proposed ASAG systems. LLM and generative models shall be adopted for the answer summarization, served as a preprocessing step before any grading suggestions or judgment. To the best of our knowledge, existing ASAG models with LLMs do not consider sentence simplification [11], [12]. In Figure 1, we have a student who answers an SA question by long sentences, which is compared to the key, with only one sentence. A reasonable approach from the cosine similarity outputs 0.703, while we simplify the student's answer, the cosine similarity value is increased. We attempt to adopt the idea of sentence simplification on ASAG for fair grading.

To speak of the techniques that are involved in the system design, we take turn to discuss the two main modules in the automated grader (AG): the module taking care the Natural Language Processing (NLP) and the module dealing with Machine Learning (ML) tasks. The NLP module can help to extract text features and to estimate the correlation between questions and student's responses [13] to name a few; while the ML module offers functionalities like answer grade prediction based on regression models [14].

We further study another common problems often seen in grading tasks. That is the inconsistency problem when two or more than two graders are involved in the same grading task but offer different grades due to alternative aspects on students' answers. To have an ML approach to apply given this context, we shall worry that inconsistent grading result may be learned by ML models which could lead to poor or unreliable judgment on future grading tasks. Traditionally, most ASAG studies compiled average scores on their models [15] to cope with the issue. We have to point out that some grading inconsistency may be created due to both teachers and their teaching assistants are involved in the same grading tasks. Given the different background and different schedule tightness from the teachers and teaching assistants, we can imagine diverse opinions from the two parties. We shall design the ASAG system that is clever enough to choose between different grading results or assign different weights to different results if the inconsistency is observed. To solve the problem, we build the ASAGeR with the major voting mechanism during the final grade prediction session.

We summarize our contributions and novelty in this work as follows:

- We propose an ASAG model for automatic grading but focusing on SA questions in this work.
- The proposed model is constructed via a sentence simplification technique. The technique is realized partially

from an LLM module, also with the help from similarity embedding encoder, and regression model, namely Automated Short Answer Regression (ASAGeR).
- We deal with the inconsistent grading problem via a major voting mechanism.

The rest of this work is organized as follows. In Section II, we discuss past studies on ASAG models and related topics. Continuing that is the detail description of the proposed method in Section III, which is followed by Section IV to show the evaluation performances of the proposed mothod. In the end, we conclude the work, also the main contributions in Section V.

II. Related Work

In recent years, ASAG studies have been improved from Large Language Models. Speaking on NLP studies for ASAG, Erickson, *et al.* [16] used Bag-of-Words (BoW) to compute each word frequency in student answers as the feature to predict the score using machine learning models, such as: XGBoost, and LSTM. Indeed, BoW cannot describe the contextual student answers as only depend on word frequencies. To have better exploration on student answers, bidirectional encoder representation from transformer (BERT) was applied on ASAG [11], [15], [17]. Further, Bonthu, *et al.* [18] incorporated fine-tuning pre-trained sentence transformer for comprehensive analysis on ASAG. Yet, those studies may resulted poor grading estimation when student answering too wordy. To omprove the grading process, we propose to feed ASAG model with sentence simplification concepts. Several studies on text simplification are listed: BERT, GPT-2 [**?**] mBART, T5, and mT5 [19] also the latest version of GPT.3-5 [20].

III. The Proposed Methodology

In this section, we elaborate the automated short answer grading system (ASAG) with all details. The goal of the proposed system is to evaluate exam answers and to facilitate exam graders' evaluation task. As an automatic grading module, we have to address (at least) a few issues: (1) how to evaluate exam answers fair; (2) how to evaluate the answers efficiently, and (3) how to incorporate with human efforts for better evaluation performance in terms of correctness and efficiency. One of the special feature of the proposed system is to leverage the power of LLMs to transform the usually lengthy students' answers to a summarized or normalized form so that the evaluation based on that can be more accurately and more efficiently done than the one based on the original answers. Moreover, given the summarized/normalized answers rather than the original answers, we shall have a superior evaluation result with or without human's efforts. The superiority is based on the comparison to other state-of-the-art approaches that work for the same problem.

With the aforementioned discussion well understood, we design the complete ASAG system which consists of three components: (1) the simplification module, (2) contextual similarity embedding, and (3) score prediction, further discussed as follows.

- The first module aims to simplify the student answers by integrating generative pre-trained transformer version 3.5 (GPT-3.5).
- The second module creates a contextual similarity embedding of students' answers and keys from the *text-embedding-ada-002* (EADA2) model with *cl100k_base* tokenizer and cosine similarity.
- The last module predicts the score using support vector regression (SVR) in a major voting fashion. In this work, GPT-3.5 and EADA2 are utilized[2].

Overall, the rationale is based on that checking the grammatical or semantic structures of the question answers could be harder than matching the answers with the answer key. But we need to confirm that the answers are not too long or too short so that the matching cannot be largely influenced by the length of answers. That is why we have the LLM-based text summarization to remove the bias created by text length. In general, the proposed model for ASAG can be seen in Figure 2. Before proceeding to the proposed model, we describe all necessary notations in the next subsection.

A. Preliminary and Notations

We start the discussion by considering three sets: the question set $\mathcal{Q}$, the answer key set $\mathcal{A}$ and the student answer set $\mathcal{B}$, further elaborated below. Given a question $q_i \in \mathcal{Q}$, we can find its answer key $a_i \in \mathcal{A}$. On the side, we shall have a student to reply with some possible answer $b_i \in \mathcal{B}$. To have a student answer b_i, we can discuss its evaluation. An evaluation function $e(q_i, a_i, b_i) \in \mathbb{R}$ is used to describe the evaluation result of b_i given the question q_i and its associated answer key a_i. Sometimes we have $e(q_i, a_i, b_i) = e(a_i, b_i)$ to emphasize a matching or comparison between a_i and b_i is enough to decide the final evaluation of b_i and the contextual information from q_i is not necessary to include.

When focusing on real cases, we learn the fact that the relationship from the set $\mathcal{B}$ to either the the set $\mathcal{A}$ or the set $\mathcal{Q}$ is a many-to-one relationship. Moreover, we should have $|\mathcal{Q}| \approx |\mathcal{A}| \ll |\mathcal{B}|$, meaning the possible answers from students are more diverse than the answer keys.

B. Automated Short Answer Grading Regressor

We proceed to explain the proposed model, namely Automated Short Answer Grading Regressor (ASAGeR) in detail. The ASAGeR contains three modules, further described in the next few paragraphs.

a) Student answer simplification: The first module of ASAGeR is to shorten or simplify all student answers $b_i \in \mathcal{B}$ based on a generative AI module such as GPT-3.5. We compile the GPT-3.5 turbo by setting a prompt:

```
Simplify the given sentences.
Keep the same meaning,
```

[2] https://openai.com/

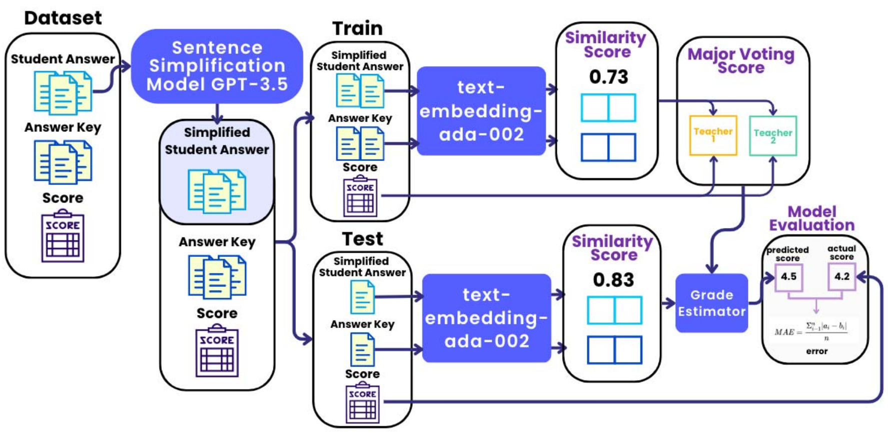


Fig. 2. An overview of the automated short answer grading (ASAG) system via sentence simplification.

`but make them simpler.`

and collect the summarized result afterwards. Consequently, we have a set of simplified student answers $\mathcal{B}_s$. On the other hand, we can also have the proposed ASAGeR to consult SGPT (GPT-Neo) [21] or other LLMs instead of GPT-3.5 to produce some alternative result for possible performance improvement.

b) Contextual similarity embedding: Given the sentence inputs, the convention is to transform sentences into their embeddings, then we can proceed for further computation and produce the grades in the end. To elaborate the complete procedure, we first collect all tokens of student answers $b_i \in \mathcal{B}$ from a tokenization method to produce tokenID vectors $T(b_i)$. Next, we create an embedding matrix $E(b_i)$ for $T(b_i)$. The embedding matrix $E(b_i) \in \mathbb{R}^{L\times d}$ represents semantic meanings of a sentence set with L as the number of all tokens (or words) in $\mathcal{B}$ and d is the dimension of the embedding space–which may stand for the length of the sequence.

The second module is to have each simplified student answer $n(b_j)$ map to its corresponding answer key $a_i \in \mathcal{A}$. That is, we aim to implement the function h to find out the answer of $h(n(b_j))$. Let $b' \in \mathcal{B}_{\text{simple}}$ be the simple student answer. The similarity between each simple student answer b' and all answer keys $\forall a \in \mathcal{A}$ based on a function h. Basically, we compute the similarity of their embeddings. Let $H = \{H_{b'_1}, \ldots, H_{b'_p}\} \in \mathbb{R}^{L\times d}$ and $H_{\mathcal{A}} \in \mathbb{R}^{L\times d}$ be the embedding matrices of simple student answers and answer keys from EADA2. The similarity function h here follows cosine similarity from [22]. Hence, we hold a feature similarity vector $v = (v_1, \ldots, v_p) = (h(b'_1, \mathcal{A}), \ldots, h(b'_p, \mathcal{A}))$. The first and second modules are depicted on Figure 3.

c) Score prediction: This part of task is to predict student's evaluation result, through a trained regression model. To focus on one question q_i with its corresponding answer key a_i, we collect a student answer b_j, normalizing/summarizing it to have $n(b_j)$, then the embedding of $n(b_j)$ is used to decide the matching between the embedding and the key, and the

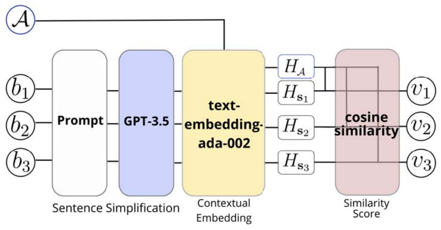


Fig. 3. The proposed ASAGeR based on GPT-3.5.

predicted evaluation is expected.

In the ASAGeR, we focus to predict each v's score based on teacher grading C. Note that the grading may come from more than one teacher due to the excessive student numbers in one classroom. Then, a set C become $\{(c_{11}, c_{12}, \ldots), \ldots, (c_{p1}, c_{p2}, \ldots)\}$. Naïvely, we just take the average, i.e., $\{\text{avg}(c_1), \text{avg}(c_2), \ldots\}$, but it may leads to a large error estimation (or high bias on the estimator). Instead of relying on average scores, we must concentrate on the most "correct" grader by presenting a major voting concept on the estimator f. In this work, we utilize support vector regression (SVR) from [22]. The last module in ASAGeR is to grade each student answer b by predicting the score of v based on function f. By focusing on each grader $\{c_{11}, \ldots, c_{1p}\}, \{c_{21}, \ldots, c_{2p}, \ldots\}$, we can predict the grade of student answers from SVR. Let $\bar{c}_{1j}, \bar{c}_{2j}, \ldots$ be the grade predictions from different teachers with $j = 1, \ldots, p$. We then calculate the error predictions, e.g., $e_{1j} = |c_{1j} - c_{1j}|, \ldots$. The major voting scheme is formulated by:

$$\hat{c}_j = \underset{c_{1j}, c_{2j}, \ldots \in C}{\arg\min} \{e_{1j}, e_{2j}, \ldots\}, \text{ for each } j = 1, \ldots, p; \quad (1)$$

with $\hat{c}$ is the predicted score from the major voting scheme.

IV. Experiment and Discussion

We conduct experiments to evaluate the effectiveness of the proposed methodology on a public dataset that is full

TABLE I
A QUESTION-ANSWERING EXAMPLE OF THE FOCUSED DATASET

$q_i \in \mathcal{Q}$	$a_i \in \mathcal{A}$	$b_j \in \mathcal{B}$	c_t	c_s	$\overline{c}$
What is the role of a prototype program in problem solving?	To simulate the behavior of portions of the desired software product.	A prototype program is used in problem solving to collect data for the problem.	1	2	1.5
		It simulates the behavior portions of the desired software product.	5	5	5

of the ASAG scenarios. Before we move on to show the experiment result, we first demonstrate the focused dataset, which is followed by a few statistics check on the dataset, such as how often students offer lengthy answers, and the lengthy answers lead to biased evaluations from graders. In the experiments, necessary experimental settings shall be shown. Different experiment results that could be linked to various settings should also be discussed.

A. Data

We introduce of the the University of North Texas dataset (or simply called UNT dataset) [1]. The dataset[3] covers ten assignments and two examinations of the Data Structure course held in the university. In the set, we have question ID, question, answer key, student answer, and two scores from either the teaching assistant (denoted by score c_t) or the graduate school student (denoted by score c_s). We can also record the average score of two, denoted by $\overline{c}$. An example of a complete question associated with its answer key and student's answer in the dataset can be found in Table I.

The dataset collects 30 students' answers on 80 questions. However, the collection is in a non-uniform way, such as some students may or may not answer all questions. After all, the total of the data samples is 2273 question-answer pairs. The score range from 0 to 5, with 0 for the incorrect answer and 5 for the perfect one. The average score is used as the ground-truth score to compare with in this case. In general, the sentence length of $\mathcal{Q}$, $\mathcal{A}$, and $\mathcal{B}$ are ranged in $[4, 25]$, $[1, 53]$, and $[1, 157]$, respectively. From that, we assume that students may reply to questions with longer than (the answer key) necessary answers. In the pre-processing step, we performed data cleansing, case folding, stopword removal, and stemming, which are usually done for text mining tasks.

B. Experimental Setup and a Quick Glance of the Experiment Series

We conduct the experiments based on two series of studies as follows. The first series is to test a few straightforward strategies to see how they can contribute to ASAG. Some strategies have ChatGPT, or different pre-processing to work on either then answer key length or the student answer length to see if we can obtain better result or not. Based on the information, we confirm the proposed methodology with the right settings to outperform other state-of-the-art approaches.

[3]The dataset can be found in https://web.eecs.umich.edu/~mihalcea/downloads.html.

TABLE II
HYPERPARAMETER SETTINGS OF THE PROPOSED MODEL

settings	default	set A	set B	set C
temperature	1	0	1	0
max_token	64	64	53*	53*

The notation (*) indicates we follow the maximum length of answer keys

In the first experiment series, we take turn to consider three approaches to see how their performance is. Based on that, we learn how to design the proposed methodology with its best settings.

- *ChatGPT viewpoints*: We set two prompts in this case. The first choice is to let the question to be the prompt. The second one is to continue providing another prompt, to shorten the previous output to enhance the performance.
- *Length of answer keys*: We partition the dataset into two subsets, the short-length (SL) set and the long-length (LL) set, such filtering student answers that are associated with their answer keys no more than five words and the remaining dataset with the key length longer than five (not included) words.
- *Just-right-length of student answers*: we compile two more subsets from the dataset by the two criteria: the student answers with their length does not exceed the keys (just-right-length or JR) and the answers with only one sentence (one-length or OL).

Given the experience of the first series of experiments, we proceed to build up the proposed methodology. There are three key decisions on this part, further discussed below.

- *Sentence simplification*: We compare the proposed model and the SOTA ones w. and w/o the sentence simplification (SS) strategy. The SS strategy is assumed helpful to extract significant information from answers in the computation of the cosine similarity to the answer keys.
- *Similarity measurement*: we take turn to consider various distance metrics on the proposed model to evaluate the contextual similarity appropriately.
- *Hyperparameter settings*: we are also interested in setting different parameters of the proposed model to observe if or not we can boost the model performance even more. Some candidates include temperature and maximum token, which can be viewed as the answer length. The detail of the parameters is given in Table II

This work presents ablation studies on the proposed model by analyzing different LLMs, various similarity measurements, and regression models for all experiment scenarios. For evaluation purposes, we split the datasets in each scenario with a ratio of $70\% : 30\%$. We compile all LLMs with SS by settings parameters: temperature=1, max tokens=64, maximum length=256, top p=1.0, frequency penalty=0.0, and presence penalty=0.0. In this work, we implemented five

TABLE III
ABBREVIATIONS OF SCENARIOS DISCUSSED IN THIS WORK.

	SS	Sentence Simplification	
	MV	Major Voting	
Answer	SL	short-length	$\text{len}(a_i) \leq 5$
Key	LL	long-length	$\text{len}(a_i) > 5$
Student	JR	Just-right-length	$\text{len}(b_i) \leq \text{len}(a_i)$
Answer	OL	one-length	$\text{len}(b_i) \leq 1$

LLMs: Sentence-BERT (SBERT) with siamese and triplet network structures [23], *generative pre-trained transformer* (GPT) [24], GPT version 2 (GPT-2) [25], Sentence-GPT (SGPT) from GPT Neo [21], and GPT-3.5. For the regression models, we employed five conventional machine-learning for regression tasks: Isotonic Regression (IR) [26], Linear Regression (LR) [27], Ridge Regression (RR) with $\alpha = 1$, and solver: singular value decomposition (SVD) [28], and Support Vector Regression (SVR) with radial basis function (RBF) as the kernel, $\gamma =$ scale, and $C = 1$ [29].

On the other hand, we define *manual* grading in a classification manner. According to the average score ($\overline{c}$) and Similarity Value (SV), we grade 0 for low sv. Otherwise, we grade avg for high SV. Given cosine similarity between the student answer and keys, cv $\in [-1, 1]$, we define the manual grading function, as follows:

$$\overline{c}^{\,+} = \begin{cases} 0 & \text{for} \quad \text{cv} \in [-1, 0.1] \\ \overline{c} & \text{for} \quad \text{cv} \in (0.1, 1] \end{cases} \; ; \tag{2}$$

After all, we evaluated the proposed model in terms of the Mean Absolute Error (MAE). Recall that the ground truth of the dataset is the average score denoted by $\overline{c}$. Thus, we calculate the MAE between the estimated and average scores to decide the model effectiveness.

$$\text{MAE} = \frac{1}{\ell} \sum_{i=1}^{\ell} \mid \overline{c_i} - \hat{c}_i \mid; \tag{3}$$

All the scenarios or strategies are again summarized in Table III. We then discuss the proposed model performances from the case study to the real case in the next section.

C. *Naïve strategies*

a) ChatGPT viewpoints: Before going on through with the proposed model discussion, we would like to illustrate how a current hot LLM-based tool ChatGPT[4] responds to the question set in the dataset. Note that we cannot apply it directly as an ASAG model because we do not have the true grade from the ChatGPT outputs. Given a few trials, we have a few observations. First, we found that ChatGPT may not be appropriate to use to answer short essays as it tends to respond to questions with wordy style answer. In Figure 4, we count how many students or ChatGPT perform better than the other. Given only the question as the prompt, ChatGPT's performance is far from perfection, and it (about 70% of them) performs worse than most students if measuring the cosine similarity between an answer and its corresponding answer key to decide the winner/loser.

[4]https://openai.com/blog/chatgpt

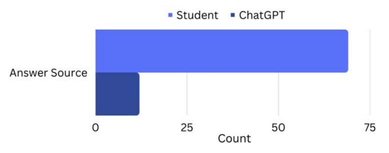
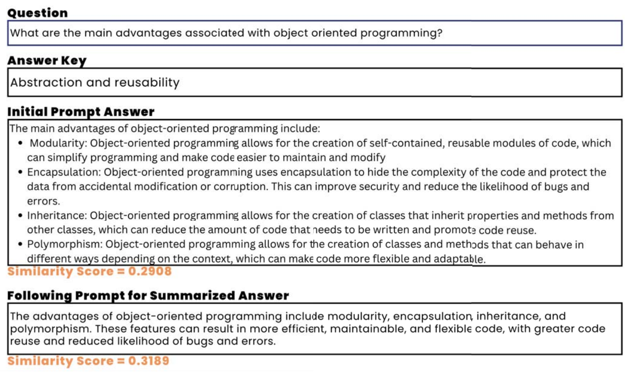


Fig. 4. A comparison between the performance from students and ChatGPT. We show counts of either students or ChatGPT produce higher cosine similarity than the other in short essay examination. Apparently, students react better than ChatGPT in this test.


Question

What are the main advantages associated with object oriented programming?

Answer Key

Abstraction and reusability

Initial Prompt Answer

The main advantages of object-oriented programming include:
- Modularity: Object-oriented programming allows for the creation of self-contained, reusable modules of code, which can simplify programming and make code easier to maintain and modify
- Encapsulation: Object-oriented programming uses encapsulation to hide the complexity of the code and protect the data from accidental modification or corruption. This can improve security and reduce the likelihood of bugs and errors.
- Inheritance: Object-oriented programming allows for the creation of classes that inherit properties and methods from other classes, which can reduce the amount of code that needs to be written and promote code reuse.
- Polymorphism: Object-oriented programming allows for the creation of classes and methods that can behave in different ways depending on the context, which can make code more flexible and adaptable.

Similarity Score = 0.2908

Following Prompt for Summarized Answer

The advantages of object-oriented programming include modularity, encapsulation, inheritance, and polymorphism. These features can result in more efficient, maintainable, and flexible code, with greater code reuse and reduced likelihood of bugs and errors.

Similarity Score = 0.3189

Fig. 5. An example of ChatGPT's response to two-stage prompts.

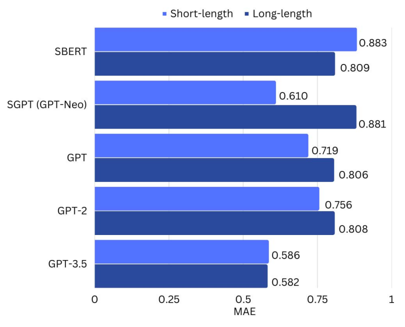
As we have seen, ChatGPT mostly fails to produce the answer of high cosine similarity to the answer key if compared to students. The second attempt from ChatGPT is by requesting another following prompt as:

```
Summarize the first response.
```

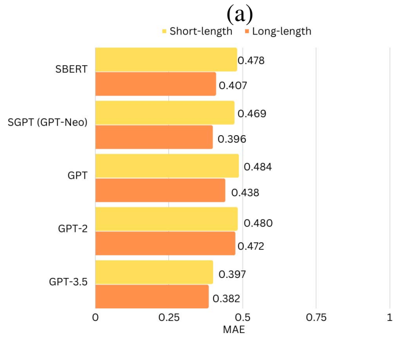
which can be viewed as a set of two-stage prompts. We illustrate a question with a very simple answer key in Figure 5. From the two-stage prompts, the similarity to answer key gets better from ChatGPT, yet the simplification result remains far away from the answer key, or fails to offer the correct answer sometimes. We continue to discuss other scenarios to study the influence of different pre-processing on the student answers regarding to SA questions in the next two paragraphs.

b) Length of Answer Keys: In this scenario, we inspect student answers that are associated with the questions having short answer keys such as $\{b_i \in \mathcal{B} : \text{len}(a_i) \leq 5, a_i \in \mathcal{A}\}$ and the ones having long answer keys $\{b_i \in \mathcal{B} : \text{len}(a_i) > 5, a_i \in \mathcal{A}\}$. Note that we may still encounter long student answers when facing short answer keys.

In this part of experiment, we compare between a naïve setting and an advanced setting following the proposed methodology, given various LLMs. In the naïve setting, we consider the regression model when the average score of two graders is the target to learn, and no sentence simplification (SS) is applied to student answers, shown in Figure 6 (a). We learn that GPT-3.5 grades better and produces more stable result, for both cases, the ones with short and long answer keys, than other LLMs.

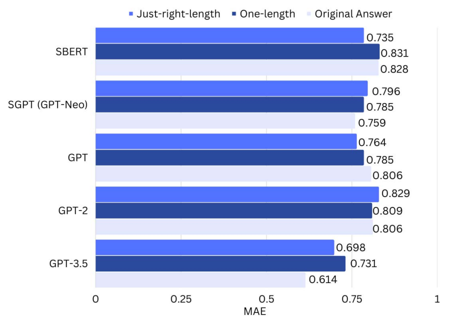


(a)

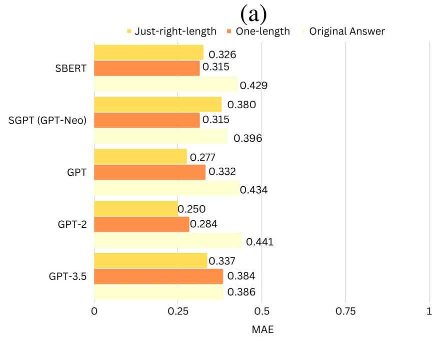


(b)

Fig. 6. A comparison of simple and long answer keys from (a) common ASAG models and (b) the proposed model.



(a)



(b)

Fig. 7. A comparison of various just-right-length student answers from (a) common ASAG models, and (b) the proposed model.

Most LLMs show difficulty to match the student answers with the long answer keys that may contain complex meanings. After that in the advanced setting, we simplified the student answers and the aforementioned major voting (MV) strategy to find appropriate target score to learn is considered. All LLMs' errors are greatly reduced for almost 50% compared to the naïve case, as shown in Figure 6 (b). Moreover, we maintain small error gaps among different LLMs and among different scenarios when student answers are filtered into different subsets. We believe that by adopting the SS+MV scheme, the grading task becomes easier than before.

c) Just-right-length of Student Answers: In this part of experiments, we again test the difference between w. and w/o applying the SS+MV scheme and their result, but on different scenarios. We purposely cut student answers from two view-points, the JR and the OL cutting to see how the SS+MV scheme reacts to the the different scenarios. In Figure 7 (a), we have the result without the help pf SS+MV scheme. As expected, all LLMs predict poorly when with either the JR or OL cutting on student answers. The performance looks not so different from the one w/o any cutting (except the GPT-3.5 case).

Unexpectedly, even with very little information after the cutting, all LLMs with SS+MV scheme can estimate grades ways better than the corresponding LLMs w/o the SS+MV scheme, as shown on Figure 7 (b). On top of that, we observe a better performance from the JR rather than the OL strategy (except the SGPT (GPT-Neo) case). Overall, the best grader is GPT-2, but only with the cutting from either JR or OL. Moreover, the robustness of the proposed model (w. SS+MV) is again confirmed. We are ready to have further study on the proposed methodology.

D. The Proposed Model on the Real-world Case

The proposed model is in fact a system consisting of a few settings which may have influence on the model outcomes. Starting from the data inputs, we can choose to have various LLMs to assist us for student answer pre-processing, such as Sentence Simplification (SS) as well as other sentence selection strategies. Given the processed inputs, we can adopt a regression model, e.g., SVR to predict the scores. The target score may be obtained from average computation or Major Voting (MV). After all, the model that adopts GPT-3.5 for SS, along with the SVR+MV combined strategies outperforms all other SOTA methods with its MAE reaches 0.386, shown in **bold** in Table IV.

Note that we follow the SOTA settings for ASAG from [12], w/o SS, and the average scores are chosen to be the target score to learn (compared to the MV setting of the proposed model). As we anticipated, the proposed model can handle the bias scores from different graders by selecting the score that is most likely from the teacher. In general, the proposed ASAGeR should be favored over other SOTA methods in ASAG. In the next few paragraphs, we show mainly the ablation study of the proposed model by taking turns to consider several alternative choices to find out the influence of them.

a) GPT-3.5 and Various LLMs: We first discuss the ablation study by trying various LLMs including one of the

TABLE IV
An ablation study to show MAE of ASAG on pre-trained model with sentence simplification.

Model	MAE														
	IR			RR			LR			**SVR**			Manual		
	w/o SS	w. SS	w. SS + MV	w/o SS	w. SS	w. SS + MV	w/o SS	w. SS	w. SS + MV	w/o SS	w. SS	**w. SS + MV**	w/o SS	w. SS	w. SS + MV
SBERT	0.907	0.812	0.588	0.944	0.832	0.614	0.943	0.832	0.612	0.828	0.740	0.429	0.839	0.862	0.735
SGPT	0.909	0.817	0.541	0.944	0.788	0.534	0.944	0.789	0.536	0.832	0.720	0.396	0.846	1.752	1.617
GPT	0.931	0.874	0.595	0.906	0.864	0.597	0.909	0.944	0.587	0.806	0.787	0.434	0.874	0.822	0.661
GPT-2	0.917	0.876	0.640	0.896	1.114	0.757	0.897	0.944	0.587	0.811	0.806	0.441	0.929	0.823	0.661
GPT-3.5	0.664	0.732	0.495	0.708	0.763	0.527	0.713	0.786	0.550	0.614	0.695	**0.386**	0.737	0.792	0.676

TABLE V
Hyperparameter settings

Model	MAE									
	IR		LR		RR		**SVR**		Manual	
	w. SS	w. SS + MV	w. SS	w. SS + MV	w. SS	w. SS + MV	w. SS	**w. SS + MV**	w. SS	w. SS + MV
default	0.732	0.495	0.763	0.527	0.786	0.550	0.695	0.386	0.792	0.676
set A	0.763	0.490	0.739	0.508	0.762	0.526	0.670	**0.372**	0.764	0.648
set B	0.742	0.487	0.754	0.504	0.759	0.530	0.670	0.377	0.777	0.663
set C	0.736	0.492	0.779	0.546	0.797	0.556	0.678	0.376	0.791	0.671

most recent GPT-3.5. As shown in Table IV, GPT-3.5 conquers almost all other models. The only exception is from LR with the SS+MV strategies, where SGPT (0.536) beats GPT-3.5 (0.550).

b) Sentence Simplification and Majov Voting: As mentioned several times, the approaches w. SS works better than those w/o SS. After adding the MV strategy, we enjoy even better performance. The above observation is true for almost all LLMs and all regression models.

c) SVR vs. Other Regression Methods: To speak of the choice of regression models, SVR shows a smaller MAE if compared to other regression methods, no matter for almost all the settings (w/o SS, w. SS, w. SS + MV). Note that SVR, with its power to deal with data non-linearity usually works better than the linear approaches. In the next part, we observe different similarity measurements to compute the contextual similarity embeddings.

d) Similarity Measurement: In this part, we discuss a few distance metrics to show which one can bring us the most representative contextual similarity embeddings. In this work, we compared cosine similarity (**??**) with Manhattan, Euclidean, and Minskowski metrics. Manhattan distance is the sum of the absolute differences between documents [30]. Meanwhile, Euclidean distance is the squared differences between the corresponding document. On the other hand, the Minkowski distance is generalized from Euclidean and Manhattan distance metrics [31] to work on q-norm space.

We want to ensure that cosine similarity provides us with the smallest MAE by applying GPT-3.5 and SVR as the model base. The comparison of MAE values between distance metrics is given on Table VI. Here, cosine similarity has the smallest error compared to other metrics since we check the angle between the contextual similarity embedding vector and the score. Meanwhile, other metrics focus on how far the difference is between them.

TABLE VI
A comparison of distance metrics to compute the similarity between student answers and keys.

Metric	Cosine	Euclidean	Manhattan	Minkowski
w/o SS	0.614	0.618	0.671	0.824
w. SS	0.694	0.696	0.694	0.698
w. SS + MV	0.386	0.389	0.386	0.390

e) Hyperparameter Settings based on GPT-3.5.: Before we already discussed the proposed model performances based on default parameters, we now try to present it's results on different parameter settings. As one of this work's contributions on answer lengths, we are eager to investigate two parameters (max_token, temperature) on with various LLMs and regression models, see the detail setting schemes in subsection IV-B. Without considering randomness and creativity, i.e., temperature = 0 (set A), we achieve the best grading results when max_length = 64 (closer to the maximum length of student answer) and SVR. In that case, we may prefer less creativity since we want to check the closeness between students' answers and keys. By allowing creativity–temperature = 1, we have slight gap errors when we set smaller max_token. For different regression models, we have different results as they may capture the randomness on different scales. Yet, we achieve the smallest MAE when (temperature, max_token) = (0, 64) in this work. The detailed results of a few parameter settings are shown in Table II.

V. Conclusions

We proposed an automated short-answer grading system called ASAGeR which can assist teachers in exam evaluation. The proposed method is especially important when students reply with long answers which may lead to inefficient grading from teachers or graders. Moreover, we discussed the case when inconsistent grading exist from more than one graders are involved in a grading task and the grading result may be misused by machine learning modeling to produce unreliable outcomes. A majority voting scheme was adopted to extract the result with possible high agreement from graders to be used to produce trustworthy learning models. Technically, the proposed method incorporated with LLMs, similarity embedding encoding, and regression models to help to re-write exam answers such that the teachers can easily deal with the rewritten, considered a summarized or normalized result for better grading experience than before. The grading result may contain less errors following the new schemes. Specifically, based on the ablation studies, we decided to construct the proposed model from GPT-3.5, cosine similarity, and SVR, for different components in the proposed system. In general, we simplified student answers with GPT-3.5 and language modeling heads. Then, we encoded the simplified answers and keys from EADA2 and computed their cosine similarity into an embedding. At the last, we predicted the embedding score from SVR with majority votes. To speak of the performance evaluation on the proposed method, we conducted experiments to evaluate the proposed method on a well-known public ASAG dataset. We first showed the importance of sentence simplification on ASAG by observing the result before and after the simplification and their corresponding question grades to see how the simplification and the proposed method can help to improve the performance of grading in terms of its effectiveness and efficiency. After all, we demonstrated the superiority of the proposed model to other state-of-the-art approaches, on the topic of SA question evaluation. We believe the proposed model can be applicable to a wide range of similar applications for its great value among school teachers in their evaluation tasks.

ACKNOWLEDGEMENTS

This research is fully funded by Institut Teknologi Sepuluh Nopember, as Research Collaboration Grants, 2023.

REFERENCES

[1] R. M. Michael Mohler, Razvan Bunescu, "Learning to grade short answer question using semantic similarity measures and dependency graph alignments," *ACL:P11-1076*, 2011.

[2] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," 2023.

[3] OpenAI, "Gpt-4 technical report," *ArXiv*, vol. abs/2303.08774, 2023.

[4] J. D. Karpicke and H. L. Roediger III, "The critical importance of retrieval for learning," *science*, vol. 319, no. 5865, pp. 966–968, 2008.

[5] M. Roy and M. T. Chi, "The self-explanation principle in multimedia learning," *The Cambridge handbook of multimedia learning*, pp. 271–286, 2005.

[6] K. Zupanc and Z. Bosnić, "Increasing accuracy of automated essay grading by grouping similar graders," in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, 2018, pp. 1–6.

[7] P. Butcher and S. Jordan, "A comparison of human and computer marking of short free-text student responses," *Computers & Education*, vol. 55, 09 2010.

[8] L. S. Larkey, "Automatic essay grading using text categorization techniques," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 1998, pp. 90–95.

[9] D. Ramesh and S. K. Sanampudi, "An automated essay scoring systems: a systematic literature review," *Artificial Intelligence Review*, pp. 1–33, 2021.

[10] M. Zhang, S. Baral, N. Heffernan, and A. Lan, "Automatic short math answer grading via in-context meta-learning," *arXiv preprint arXiv:2205.15219*, 2022.

[11] C. Sung, T. I. Dhamecha, and N. Mukhi, "Improving short answer grading using transformer-based pre-training," in *International Conference on Artificial Intelligence in Education*. Springer, 2019, pp. 469–481.

[12] S. K. Gaddipati, D. Nair, and P. G. Plöger, "Comparative evaluation of pretrained transfer learning models on automatic short answer grading," 2020.

[13] Z. Wang, H. Huang, L. Cui, J. Chen, J. An, H. Duan, H. Ge, N. Deng *et al.*, "Using natural language processing techniques to provide personalized educational materials for chronic disease patients in china: development and assessment of a knowledge-based health recommender system," *JMIR medical informatics*, vol. 8, no. 4, p. e17642, 2020.

[14] Z. Yuan, "Interactive intelligent teaching and automatic composition scoring system based on linear regression machine learning algorithm," *Journal of Intelligent & Fuzzy Systems*, vol. 40, no. 2, pp. 2069–2081, 2021.

[15] H. Ghavidel, A. Zouaq, and M. Desmarais, "Using BERT and XLNET for the automatic short answer grading task," in *Proceedings of the 12th International Conference on Computer Supported Education - Volume 1: CSEDU,*, INSTICC. SciTePress, 2020, pp. 58–67.

[16] J. Erickson, A. Botelho, S. McAteer, A. Varatharaj, and N. Heffernan, "The automated grading of student open responses in mathematics," 03 2020, pp. 615–624.

[17] A. Condor, "Exploring automatic short answer grading as a tool to assist in human rating," *Artificial Intelligence in Education*, vol. 12164, pp. 74 – 79, 2020.

[18] S. Bonthu, S. Rama Sree, and M. Krishna Prasad, "Improving the performance of automatic short answer grading using transfer learning and augmentation," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106292, 2023.

[19] S. Štajner, K. C. Sheang, and H. Saggion, "Sentence simplification capabilities of transfer-based models," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 11, pp. 12 172–12 180, Jun. 2022.

[20] Y. Feng, J. Qiang, Y. Li, Y. Yuan, and Y. Zhu, "Sentence simplification via large language models," 2023.

[21] N. Muennighoff, "SGPT: GPT sentence embeddings for semantic search," *arXiv preprint arXiv:2202.08904*, 2022.

[22] K. Zhou, K. Ethayarajh, D. Card, and D. Jurafsky, "Problems with cosine as a measure of embedding similarity for high frequency words," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 401–423.

[23] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[24] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[25] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[26] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 625–632.

[27] M. S. Acharya, A. Armaan, and A. S. Antony, "A comparison of regression models for prediction of graduate admissions," in *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, 2019, pp. 1–5.

[28] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," 2007.

[29] A. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[30] A. A. Keller, "Chapter 3 - elements of technical background," in *Mathematical Optimization Terminology*, A. A. Keller, Ed. Academic Press, 2018, pp. 239–298.

[31] V. H. Kamble and M. P. Dale, "Machine learning approach for longitudinal face recognition of children," *Machine Learning for Biometrics*, 2022.