Automated Short Answer Grading - a Peer Aware Approach

Gokularajan R

VIT University, School of Computer Science and Engineering, gokularajan.r2022@vitstudent.ac.in

Abstract

Due to the rapid growth of online learning platforms and large-scale assessments, manually grading short answers has become a significant bottleneck. Evaluating such responses requires assessing both correctness and the quality of reasoning, which poses challenges for automated systems. Traditionally, Automated Short Answer Grading (ASAG) systems grade each student's answer independently against a predefined key. This approach ignores collective patterns across responses and overlooks valid variations or keywords not present in the key, often resulting in inconsistent grading and increased sensitivity to outliers. To address these limitations, this paper proposes a novel ASAG method that evaluates student answers in relation to each other by a two pass approach, one to extract a dynamic key after analysing all the answers and the next to grade each answer relative to this key. By incorporating inter-answer dependencies, my system captures semantic and contextual nuances more effectively—much like human graders—leading to improved grading consistency and fairness. Evaluating the systemn on the Mohler dataset introduced in [1], my approach achieves a root mean square error (RMSE) of 0.81, outperforming existing methods that rely on static keys and independent grading. This demonstrates the potential of peer-aware ASAG systems to enhance automated evaluation in educational settings.

Introduction

With the rapid rise of online learning platforms and large-scale digital assessments, evaluating student performance has become increasingly dependent on automated systems. While multiple-choice questions are easily graded, short-answer questions pose a significant challenge due to their open-ended nature and linguistic variability. Short-answer grading requires not only validating factual correctness but also assessing the reasoning and expression behind a student's response. Manual grading of such responses is time-consuming and impractical at scale, especially in massive open online courses (MOOCs) and large classrooms. This has fueled interest in Automated Short Answer Grading (ASAG) systems that can provide consistent, efficient, and scalable evaluation.

Despite significant advances, existing ASAG systems face key limitations. Most approaches rely on static, predefined answer keys, comparing each student response independently against this fixed reference. This rigid framework struggles with the diversity of valid expressions and synonyms, often penalizing correct answers that deviate from the expected phrasing. Furthermore, independent grading makes systems highly sensitive to outliers and unable to detect collective patterns such as common misconceptions. These issues lead to inconsistencies and reduce grading fairness, highlighting the need for systems that are adaptive, context-aware, and capable of leveraging information from the entire set of responses.

Past approaches to extract keywords include supervised learning methods on Support Vector Machines [7] and Naive Bayes Classifiers [8]. [9] describes a keyword as relevant words that provide a rich semantic information about a text. Approaches using TF-IDF weighted vectors have been explored in the past, but they typically grade each response independently without considering inter-answer relationships. We evaluate our method on the dataset introduced by [1], where a peer-independent TF-IDF–based system was used as the baseline. Their approach, which compares each student response to a static reference answer using TF-IDF similarity, achieved a root mean square error (RMSE) of 1.022. Similarly, [3] uses TF-IDF representations to evaluate similarity with a static answer key, followed by training a Support Vector Machine (SVM) to predict scores for their best result of RMSE of 1.15. [2] uses a clustering approach to group similar responses and assign scores using the hamming distance between an answer and a static answer key and compares the errors of the model with the disagreement between teachers. All these papers use the same dataset and similar methods, but they do not leverage the collective patterns in responses or the relationships between answers to improve grading accuracy and consistency.

To address these challenges, I propose an ASAG approach that grades student responses in relation to each other rather than in isolation. My method employs a two-pass strategy: in the first pass, it constructs a dynamic grading key which starts as a bag of words from the key and all the words in all the answers. Now each word from every student answer is now added to this key with a "strictness" parameter that weights the influence of each student answer in the key. In the second pass, the similarity between individual responses are evaluated relative to this dynamic key and their peers. We then model the relation between these similarities and scores from human graders. This simple system recognizes diverse yet valid answer formulations and adapt to context-specific variations. This peer-aware design provides a grading process that more closely resembles human evaluators, resulting in improved consistency, fairness, and robustness against outliers.

The rest of the paper is organized as follows: Section 2 reviews related work, Section 3 presents my proposed method, Section 4 discusses evaluation results, and Section 5 concludes the paper with future scope.

Related Work

Automated Short Answer Grading (ASAG) has been widely explored using traditional machine learning and more recently using LLM based grading methods. [3] explores the use of large language models (LLMs) for automated short answer grading and compares their performance against traditional machine learning approaches. Their findings indicate that LLM-based grading performs worse than conventional methods on their chosen dataset. Specifically, their best-performing model uses TF-IDF representations of student responses to evaluate similarity with a static answer key, followed by training a Support Vector Machine (SVM) to predict scores. However, their approach grades each response independently, without leveraging inter-answer relationships or dynamic key construction. Consequently, their system reports a best RMSE of 1.15, which is notably higher than the 0.81 achieved by my peer-aware method.

[5] presents a multi feature approach for automated answer grading. The system computes alignment scores using a word aligner enriched by lexical and contextual similarities. It calculates cosine similarity between semantic vectors derived from word embeddings. To reduce bias from question words, a demoted version of each feature is also computed after removing these terms. Furthermore, a tf-idf variant is used to weigh terms based on their in-domain frequency, gathered through Wikipedia pages related to the question context. These features are combined with a length ratio metric, and the final feature set is used to train a regression model. This method extracts and uses 4 features and achieved an RMSE of 0.887 on the Mohler dataset. However I only use a single feature of semantic similarity.

[2] introduces a model that predicts marks by measuring the similarity between student responses and a predefined answer key. Similarity is computed based on the overlap of common words using Hamming distance. To assign scores, the system analyzes the relationship between these similarity measures and existing scores, followed by clustering the responses using k-means. Each cluster is then assigned the same score, ensuring consistent grading within groups of similar answers. Additionally, this clustering approach provides teachers with insights into the variety of student responses, enabling targeted feedback for each cluster and helping address diverse answer formulations.

[6] leverages contextual word embeddings from pre-trained transfer learning models to capture semantic meaning. Rather than fine-tuning these models for the target task, they utilize the frozen embeddings obtained from source-task training, assuming that these representations already encapsulate rich semantic and syntactic features. This approach avoids the computational cost of full model adaptation while still benefiting from the deep linguistic knowledge encoded in large-scale pretrained corpora. This method achieves their best RMSE of 0.978 on the Mohler dataset.

Term Frequency - Inverse Document Frequency (TF-IDF) is one of the most widely used techniques to judge the importance of a keyword in language processing tasks and there are several variations in usage. For instance, [9] introduces a semantic sensitive TF-IDF method for scoring word importance that adjusts traditional TF-IDF scores using semantic representations, so that words semantically related to high-TF-IDF terms are also assigned greater importance. These adaptations demonstrate the continued relevance of TF-IDF in capturing meaningful term importance.

[4] presents a system that fine-tunes the RoBERTa-Large model, pretrained on the STS Benchmark dataset, for the task of short answer grading. The system was evaluated on the same Mohler extended dataset. The approach achieved an RMSE of 0.7, surpassing the previously reported state-of-the-art results (RMSE of 0.793).

While [4] achieves strong performance using a large pretrained transformer model, my lightweight method requires only seconds to train and still attains a competitive RMSE of 0.81. This makes it a highly efficient alternative—offering state-of-the-art performance within its computational class. In the following section, I present my proposed approach, which builds upon the insights from prior work while addressing their limitations in scalability and flexibility.

Proposed Method

Motivated by the limitations of existing methods—such as the use of static keys or resource-heavy architectures—we now describe a simple yet effective method that dynamically adapts to the distribution of student responses.

Preprocessing

I start by cleaning the data by converting all to lowercase, removing punctuations and discarding stop words listed in the default R `stopwords()` function. I also stem the words in the answer and key to their basic forms using the `stemDocument()` function from the `tm` package. This ensures that variations of words (e.g., "running" vs. "run") are treated as equivalent.

Vectorization and key construction

Student answers were vectorized using the TF-IDF scheme via the `text2vec` package. Each answer was tokenized, and a document-term matrix was constructed. This was then transformed using TF-IDF resulting in a matrix of importance of each keyword in each answer across all answers. A bag of words and a vector that contains the frequency of each word across the whole corpus is created for key construction. This is the first pass of going through all the answers.
Further, I also demote the keywords in the key by removing the words that are present in the question to ensure students are not graded on repeating the question words.

$$\text{for all terms } t \text{ in corpus } d = ans \bigcup key$$

$$\text{TF}(t, d) = \frac{f_{t,d}}{T_d}$$

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \cdot \log \frac{N}{n_t}$$

where N is the number of documents in the corpus and n_t is the number of documents that contain the term t.

Now for the second pass, each answer and key is represented using the bag of words approach, where each document is a binary vector of the presence or absence of words in the vocabulary. The key too is initialized in the same way but we weight down the keyword frequency vector by a "strictness" parameter. This parameter controls how much influence each student answer has on the key. A higher strictness value means that the key is more influenced by the answers, while a lower value means that the key is more static and less influenced by individual answers. I have used 20 as the strictness parameter as it consistently gave me the best performance. Now to the answer vector, we add the corresponding TF-IDF vector to the binary bag of words vector. This process is done independenly for each question. i.e. such that only the answers to the same question are considered for the key construction.

$$key = key + \frac{\text{TF}}{\alpha}$$

where α is the strictness parameter and TF is the term frequency vector of the answer.

For i in all student answers:

$$ans = ans + \text{TF-IDF}[i, :]$$

Similarity Calculation and Grading

To evaluate the similarity between each student answer and the dynamic key, I use the cosine similarity measure. This metric captures the divergence between two vectors, providing a normalized score that ranges from 0 (no similarity) to 1 (identical). The cosine similarity is computed by taking the dot product of the answer vector and the key vector, and normalising the result. Now that we have similarity scores and the human graded scores, it becomes a supervised learning problem. The similarity and the scores are used to train a polynomial regression of degree 7, which I have found to model the similarity-score relationship well.

$$c = \sum ans \cdot key$$

$$\textbf{similarity} = \frac{c}{\sqrt{\sum l_1 \cdot \sum l_2}}$$

Evaluation

Dataset

The dataset used for evaluation is the Mohler extended dataset introduced in [1]. It consists of 2264 student responses to a short-answer question, along with two human-annotated scores ranging from 0 to 5. Each row is a tuple of `<question, desired_answer, student answer,  score1, score2, score_avg>`.

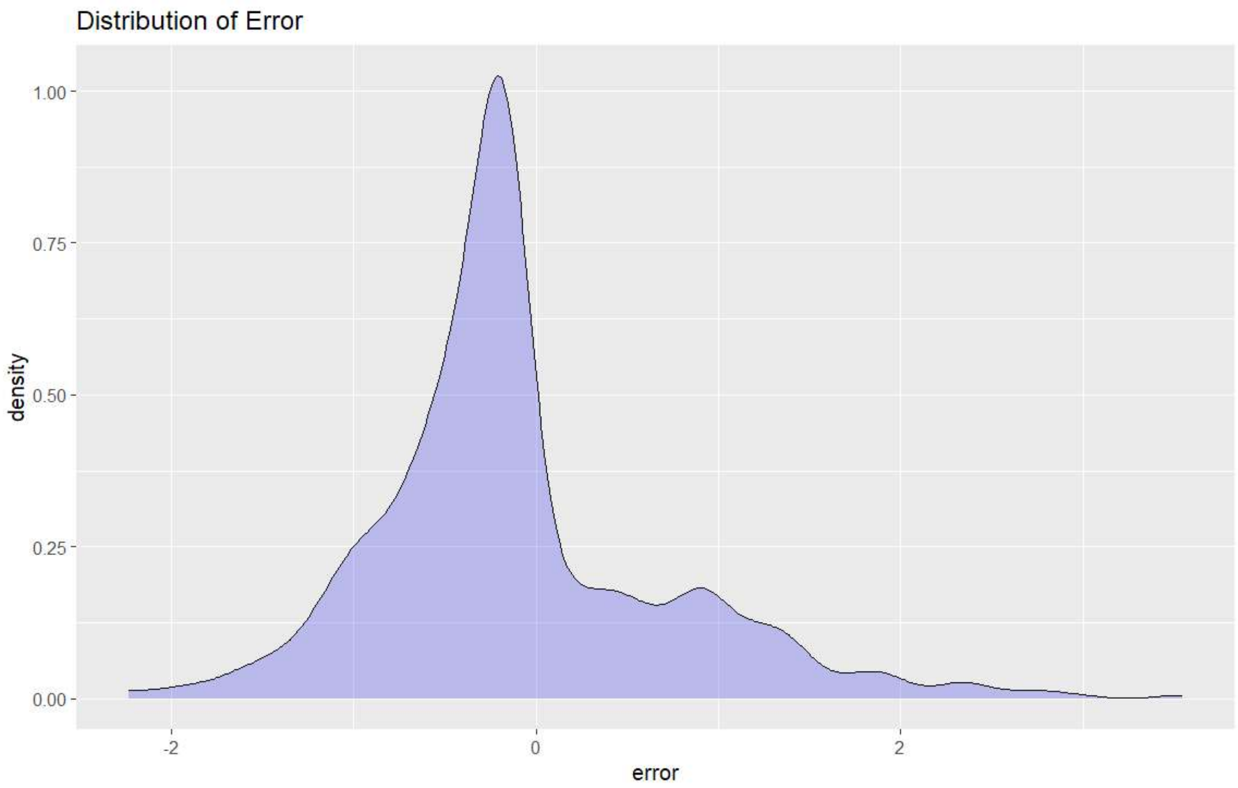
Column	Value
question	What is the role of a prototype program in problem solving?
desired_answer	To simulate the behaviour of portions of the desired software product.
student_answer	High risk problems are address in the prototype program to make sure that the program is feasible. A prototype may also be used to show a company that the software can be possibly programmed.
score1	4
score2	3
score_avg	3.5

Table 1 : Example tuple from the used dataset

I use 80% of the questions in the dataset for training and test on the remaining 20%. The dataset is split such that each question is only present in either the training or test set, ensuring that the model does not see any answers to the same question during training.

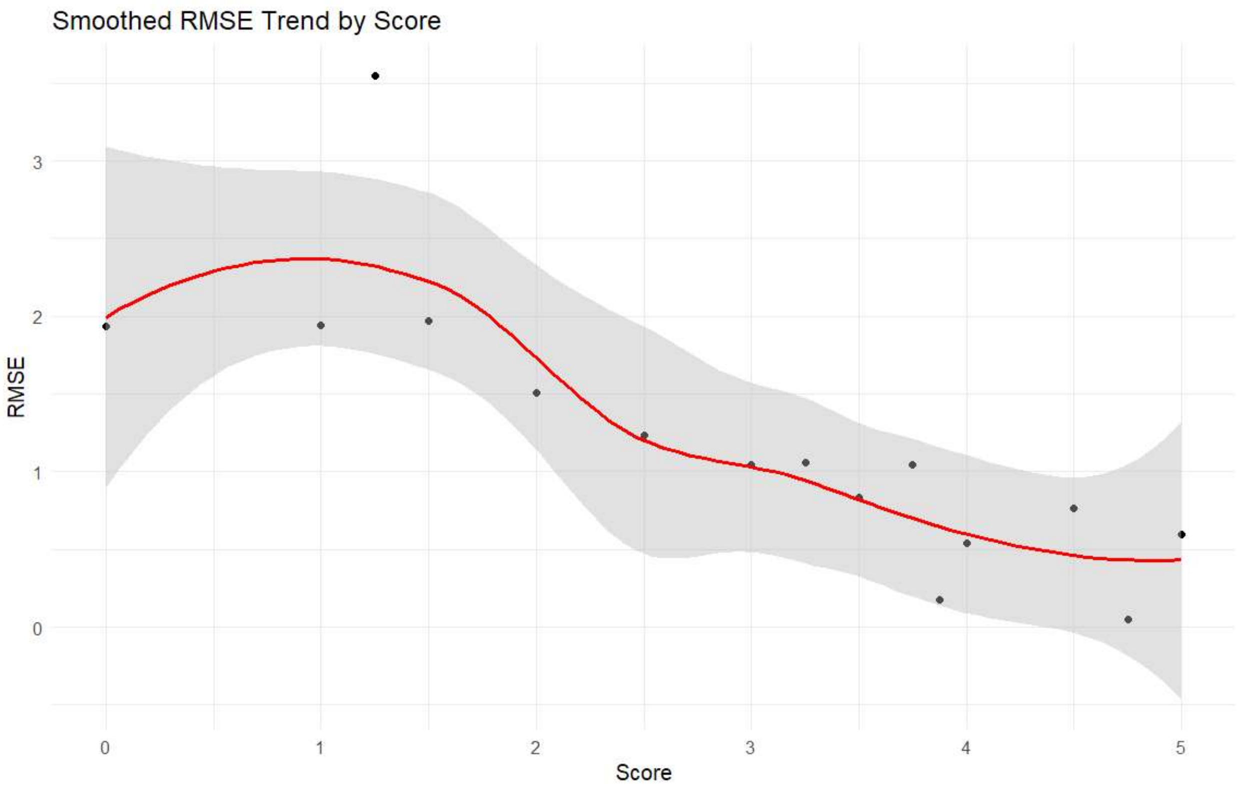
As mentioned before, several prior works have used TF-IDF-based features for short-answer grading. To evaluate our improvements, we compare our model against these baselines under similar setups using the Mohler dataset [1] and report results on RMSE against the average scores provided in the dataset.

Method	RMSE
TF-IDF + SVM [3]	1.15
Roberta-Large fine-tuned [4]	0.7
TF-IDF + SVR [1]	1.022
Bag of Words [1]	0.978
Similarity feature + Ridge Regression (in domain) [5]	0.85
TF-IDF weighted similarity + regression (Mine)	0.81



Future Scope

While the current model demonstrates competitive performance with a lightweight design, several avenues exist for further enhancement. First, the choice of model architecture can be revisited to explore more expressive or specialized models that capture deeper semantic relationships. Additionally, incorporating auxiliary features—such as syntactic roles, answer length, or embedding-based similarity—during supervised training, as demonstrated in [5], could potentially improve the model's discriminative ability. Expanding and diversifying the training data may also contribute to improved generalizability across varied question domains. The dataset used in this work is notably imbalanced, with a majority of answers receiving high scores as shown in the following graph where errors for lower scores significantly increase. Minority oversampling during training may help mitigate this imbalance and enhance the model's generalizability, particularly for underrepresented answer types.



References

[1] : Learning to Grade Short Answer Questions using Semantic Similarity Measures and Dependency Graph Alignments - Michael Mohler et al. (2011)

[2] : AUTOMATIC SHORT ANSWER GRADING AND FEEDBACK USING TEXT MINING METHODS N. SUZEN, A.N. GORBAN et al. (2019)

[3] : Automatic Short Answer Grading in the LLM Era: Does GPT-4 with Prompt Engineering beat Traditional Models? - Rafael Ferreira Mello et al. (2025)

[4] : FINETUNING TRANSFORMER MODELS TO BUILD ASAG SYSTEM - Mithun Thakkar (2021)

[5] : Fast and Easy Short Answer Grading with High Accuracy - Sultan et al. (2016)

[6] : Comparative Evaluation of Pretrained Transfer Learning Models on Automatic Short Answer Grading - Sasi Kiran Gaddipati (2020)

[7] : Keyword extraction using support vector machine - K. Zhang et al. (2006)

[8] : Keyword extraction using naive bayes - Y. Uzun et al. (2005)

[9] : Semantic Sensitive TF-IDF to Determine Word Relevance in Documents Amir Jalilifard et al. (2021)