AASE: AI-Driven Automated Answer Script Evaluation

Mitra Abhi Sura
Shiv Nadar University
Maitreyee Rai
Shiv Nadar University
SONIA KHETARPAUL
sonia.khetarpaul@snu.edu.in

Shiv Nadar University
Saurabh Mishra
BML Munjal University

Article

Keywords: Artificial Intelligence, Education Assessment, Natural Language Processing, Transformer Models

Posted Date: July 17th, 2025

DOI: https://doi.org/10.21203/rs.3.rs-6895375/v1

Additional Declarations: No competing interests reported.

AASE: AI-Driven Automated Answer Script Evaluation

Mitra Abhi Sura[1], Maitreyee Rai[1], Sonia Khetarpaul[1][0000-0001-6058-7235], and Saurabh Mishra[3][0000-0002-1965-7905]

[1] Department of Computer Science and Engineering,
Shiv Nadar Institution of Eminence, Delhi NCR, India
`{ms139, mr644, sonia.khetarpaul}@snu.edu.in`

[2] Department of Computer Science and Engineering, BML Munjal University, India
`{saurabh.mishra}@bmu.edu.in`

Abstract. In today's educational systems, evaluating student answer-scripts are challenging due to the varied grading criteria, different types of questions, and different ways of attempting the same question. Traditional manual grading can often be inconsistent, inefficient, and sometimes prone to bias, making it difficult to ensure fairness in assessments. These challenges are further complicated when dealing with different types of answers, such as written responses in English, mathematical solutions, etc. Each of these requires distinct approaches, increasing the workload and chances of errors in manual grading. To address these challenges, we propose an Automated Answer Script Evaluation (AASE) system. This solution leverages advanced Natural Language Processing (NLP) techniques along with mathematical parsing algorithms to automate the grading process comprehensively. The proposed AASE system is trained on a diverse dataset containing various grading criteria and incorporates multiple techniques for evaluating both English-based and mathematical answers. For English responses, the system employs different approaches like keyword matching and the Word Movers Distance (WMD) algorithm, a BERT model with additional layer and BERT-based model with dropout layers for sequence classification. These models ensure accurate and unbiased evaluations of student answers in English. Additionally, the AASE system integrates optical character recognition (OCR) technology to recognize handwritten mathematical expressions, which are then converted into LaTeX format using an encoder-decoder architecture. The converted expressions undergo evaluation providing flexibility in assessing both direct answers and detailed step-wise solutions. The AASE system is trained and tested on real-world datasets, and has achieved an accuracy of 80.45% for Subjective and 76% for Mathematical answers evaluation.

Keywords: Artificial Intelligence· Education Assessment · Natural Language Processing · Transformer Models

1 Introduction

Answer Script evaluation remained one of the complex, time taking and lethargic task for academicians around the world. Once the theoretical and mathematical answer scripts are evaluated, the justification for the awarded marks and satisfaction of the students with it always result in a debate among them [1]. Another core problem of answer script evaluation is that sometimes different evaluators award different marks to the same answers or give marks with erring hand [2] which results in unnecessary competitiveness among students and discourage them. Some of these issues also question the fairness of the existing examination system when the performance of different students is evaluated on the same pointer scale by different evaluators [3].

Automated answer script evaluation systems represent our attempt to foray into the vast field of educational assessment, offering efficient and objective grading mechanisms for student answers. It is essential that these systems are capable of leveraging NLP techniques to interpret and evaluate responses in various subject domains. With the proliferation of machine learning and NLP technologies, automated grading systems have gained traction in educational settings, promising to streamline the evaluation process and improve assessment consistency. Researchers in recent years discussed various methods for the easing the assessment process in the academic institutes.

The aim of this research is to develop an Automated Answer Script Evaluation (AASE) system to address the challenges associated with manual grading procedures. The main objective is to develop a robust and accurate automated grading system capable of evaluating student responses across various subjects and topics with high precision. This endeavor streamlines the grading process and provides educators with valuable insights into student performance and comprehension. The AASE system makes use of transformer models and algorithmic capabilities to assess student responses across multiple subject areas, enhancing efficiency and fairness in answer script evaluation in academic institutes. The Key contributions of the proposed AASE system are:

1. We used and evaluated three approaches- keyword matching and WMD algorithm, BERT model with an additional layer, and BERT with sequence classifications to evaluate subjective answers written in the English language.
2. We have used DenseNet Encoder and Attention-based decoder to convert to recognize and convert handwritten mathematical expression into LaTeX format and proposed evaluation algorithm to grade the mathematical answers.
3. We evaluated the AASE system on the real-world datasets and achieved an accuracy of 80.45% for subjective and 76% for Mathematical answers evaluation.

The rest of the paper is organized into the following sections: Section 2 reviews the existing work. Section 3 presents our methodology. Section 4 discusses the datasets. Section 5 reports on experimental results. Finally, we conclude in Section 6.

2 Related work

Text analysis remained one of the most discussed topics among the computer and linguistic research in recent years, from the last few decades, mathematicians, engineers, scientists and anthropologists are examining and finding different facts in linguistic tradition and the sociological traditions [9]. With the evolution and expansion of the internet, and generation massive amount of text data, researchers and software conglomerates focused on developing methods and tools to study and analyze these massive datasets and find facts [22]. Researchers [17, 10] were continuously putting effort into developing different algorithms and machine learning models that can generate answers which match or near to the human level of intelligence.

In 2017-18, with the introduction of transformer based neural network [25, 14] which has been proved breakthrough research in the area of text generation and ability of question answering systems. This section elaborates on the different models and methods that can be used for answer evaluation.

Automated short answer scoring(ASAS) [24, 19, 28, 12, 15] and automated essay scoring(AES) [8, 20, 26, 11] has witnessed significant advancements, influenced mainly by developments in neural network-based NLP tasks.

Banshir et al. in [8] proposed a novel approach that utilizes various machine learning, NLP techniques, such as word mover's distance (WMD), cosine similarity, multinomial naive bayes, etc for automatic evaluation of descriptive answers. The authors in [28] developed an automatic semi-open-ended short-answer grading evaluation model that used a LSTM RNN to learn the representation in the classifier. The authors in [20] proposed method contains various steps such as question classification, answer classification and answer evaluation, and used syntactical relation-based feature extraction technique for automatic evaluation of descriptive-type answers. Yad et al. in [26] worked by taking inputs like the user response, expected sentences and expected keywords and works with the help of numerical vectors, natural language processing and deep learning approaches.

The authors in [13] presented a survey of automatic question generation and assessment strategies from textual and pictorial learning resources. Gao et al. in [16] has systematically reviewed the recent educational applications of text-based assessment systems for understanding the latest developments assisting in text-based assessments in higher education.

Sultan et al. [24] utilized short-text similarity with key grading specific contruct for automated short answer grading. Researchers in [19] utilized two-stage deep neural network for prompt independent automated essay scoring, the two stage deep neural network learn from pseudo labels by considering semantic, part of speech and syntactic features of the text. Condor and Litster [12] worked on the grading of short answers using large language models, such as BERT. Funayama et al. [15] used two phase approach (i.e., train the model on existing rubrics and fine tune the model on a given new prompt) for cross prompt training for short answers scoring. Cochran et al. [11] used the augmented method for text data generation and evaluation of student text responses. Zeng et al. [27] utilized framework based on Prototypical Neural Network to determine automatic short answer scoring. Some recent work [7, 21, 18, 23] used large language models and generative AI for answersheet evaluation. But, LLMs are resource-intensive, requiring significant computational power for fine-tuning and deployment, making them costly for large-scale use in answer sheet evaluation.

This paper focused on evaluating both subjective as well as Mathematical answers. We have proposed three approaches- keyword matching and WMD algorithm, BERT model with an additional layer, and BERT with sequence classifications to evaluate subjective answers written in English language. We have also used image processing along with the DenseNet Encoder and Attention-based decoder to evaluate the mathematical answers.

3 Methodology

In this section, we discuss the different approaches used to evaluate subjective answers and mathematical answers.

3.1 Evaluation of Subjective Answers

In this subsection, we present a discussion on three distinct approaches and algorithms employed to evaluate students' subjective answers.

1. **Keyword matching and WMD algorithm:**
   The Custom Similarity Algorithm is applied to compute the similarity between the model answer and the answer to be graded. It starts by preprocessing both the model and the student's answers using the following steps.
   - For each sentence in the model answer, the similarity between each sentence in the student's answer is calculated using Word Mover's Distance (WMD) algorithm (Algorithm 1).
   - WMD measures the semantic similarity between two sentences (it's D_1 and D_2 here) based on the distance between word embeddings.
   - Additionally, the algorithm considers keyword weights to account for the importance of shared keywords between the model answer and the student's answer.
   - It calculates a composite score for each sentence pair, considering semantic similarity (based on WMD) and keyword matching.
   - The algorithm also penalizes for missing keywords in the student's answer compared to the model answer. This penalty is applied to account for essential information or concepts not addressed by the student.

   Finally, it computes an overall score by aggregating the scores for all pairs of sentences, factoring in keyword weights and missing keyword penalties. The output of the custom similarity algorithm is a single score representing the similarity between the model answer and the student's answer. This score reflects semantic similarity and keyword matching, thoroughly evaluating the student's response.

Algorithm 1: WMD Algorithm

```
1: WMD($D_1, D_2$)
2: distance_matrix ← []
3: for each word $w_1$ in $D_1$ do
4:    row ← []
5:    for each word $w_2$ in $D_2$ do
6:       distance ← $comp\_vec\_dist(embedding(w_1), embedding(w_2))$
7:       row.append(distance)
8:    end for
9:    distance_matrix.append(row)
10: end for
11: $T \leftarrow Solve\_Earth\_Mover\_Distance(distance\_matrix, P(D_1), P(D_2))$ {word
    distributions}
12: WMD_distance ← 0
13: for each $i, j$ in $T$ do
14:    WMD_distance $+= T[i][j] \times$ $distance\_matrix[i][j]$
15: end for
16: return WMD_distance
```

Final Score: The algorithm then calculates the final score based on the average scores obtained for each pair of sentences in the model answer and the student's answer. Then, a missing keyword penalty is applied to adjust the score based on the presence or absence of essential keywords in the student's response. Keyword weights and WMD distances contribute to the calculation of each pair-wise score, providing a nuanced assessment of similarity.

2. **BERT with Additional Layer:** The central components of the grading system are the BERT model and tokenizer. The grading model is structured as a neural network module, it incorporates BERT as its backbone, followed by an additional dropout layer and a fully connected layer for grading prediction.

   Dropout Layer Integration: The Dropout layers are essential components in neural network models, facilitating regularization and preventing overfitting by randomly dropping out a proportion of the units during training. This layer helps improve the model's generalization ability by reducing interdependent learning among neurons, thus enhancing robustness to variations in input data.

   This dropout layer is positioned after the BERT backbone to introduce regularization while maintaining generalization capabilities. With a dropout probability of 0.1, this layer randomly sets 10% of the input units to zero during each training iteration, effectively forcing the model to learn more robust features. Consequently, the dropout layer aids in preventing the model from memorizing noise or irrelevant patterns present in the training data, thereby promoting better generalization to unseen data during the evaluation and testing phases.

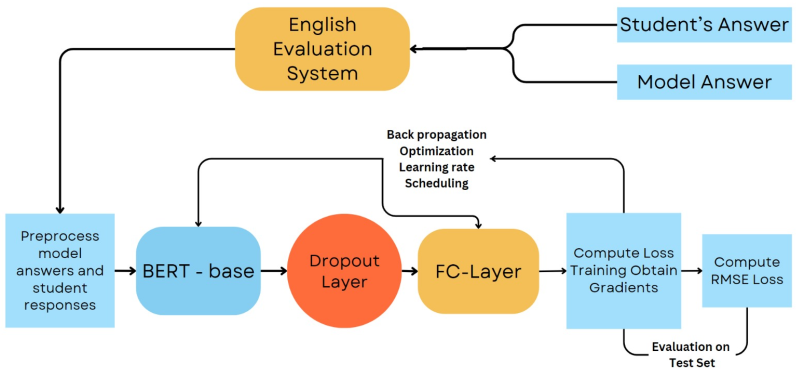


Fig. 1: BERT based Model for Subjective Answer Evaluation

Model Training: The BERT layers are unfrozen to enable parameter updates during training. The training loop iterates over epochs, batches, and data samples, tokenizing input sequences and computing loss against the human-marked grades. Both student and reference answers are passed through BERT, followed by a dropout layer and a fully connected layer with a single class, representing correctness and its probability. This probability is scaled from a range of 0 to 5. The loss is then calculated against the human-marked grades, and the optimizer updates model parameters based on calculated gradients, optimizing performance.

During each epoch, batches of student answers, reference answers, and corresponding grades are processed. Input sequences are tokenized using the BERT tokenizer, and padding or truncation is applied to ensure uniform sequence lengths. The model then predicts grades for each input sequence, and the root mean squared error (RMSE) loss is calculated between predicted and true grades. Back-propagation is performed to update the model's parameters, followed by optimization using the Adam optimizer.

3. **BERT based Sequence Classification:**

The implementation of BERT-based sequence classification for automating the grading process in educational assessments involves several key steps, beginning with the initialization of the model and tokenizer and proceeding through data preprocessing, dataset creation, model training, evaluation, and model persistence. The workflow of the model for evaluation of subjective answers is shown in Figure 1.

Model Initialization and Tokenization: In this stage, the BERT model and tokenizer are initialized. The model is configured for sequence classification with a single output label i.e the grades. The Dataset, comprising student answers and corresponding grades, is loaded.

It is important to note that the architecture for Bert For Sequence Classification is very similar to the architecture of the BERT model as shown in Fig.

1. The key difference is that the fully connected layer is given the number of labels in the particular task instead of finding the probability of correctness of the student's answer.

Model Training and Validation: The training loop iterates over multiple epochs, with each epoch comprising batches of training data. Within each epoch, the model computes predictions for a batch of input data and calculates the corresponding loss using the specified loss function. The optimizer updates the model parameters based on the computed gradients, while a linear learning rate scheduler adjusts the learning rate during training. Training progress, including average training loss and elapsed time, is logged for each epoch.

After each epoch of training, the model's performance is evaluated on the validation set to assess its generalization capabilities. Evaluation metrics such as validation loss are computed and logged to monitor the model's performance.

3.2 Evaluation of Mathematical Answers

In this subsection, we present an approach and an algorithm used to evaluate students' mathematical responses.

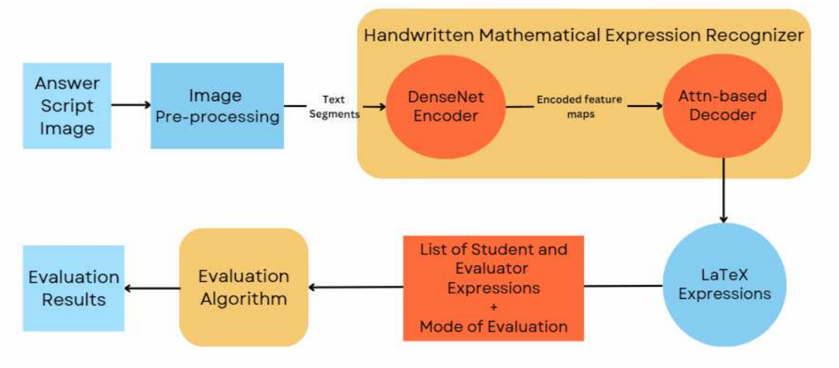


Fig. 2: Model for Mathematical Answer Evaluation

Figure 2 shows the workflow of the model used, and Algorithm 2 describes the complete procedure. The initial step is to recognize handwritten expressions from an image. Additionally, converting the input expression into LaTex (a system for high-quality technical typesetting) was deemed essential for the evaluation algorithm.

Image Pre-processing: Before feeding an image to the model, the image needs to be pre-processed. The input to the model is the student's answer as an image containing all the expressions, written in a step-wise manner one below the other. The model can only detect one expression at a time, therefore, the image is processed as follows:

- First, we read and preprocess the image by converting it to grayscale, detect and remove preexisting lines, and binarize to highlight text regions.
- Identify the rotated rectangle bounding the text regions and ensure the longer side is horizontal. This ensures that the text lines are horizontal.
- Distinguish between space lines and text lines using horizontal projections.
- Compute mean coordinates of white pixel groups to locate text lines accurately and visualize them by drawing green lines on the image.
- Segment text regions based on the detected lines and save each segmented text region as an individual image.

Encoder-Decoder Model: The model takes an image input that contains a mathematical expression and outputs the corresponding LaTeX expression. It has two parts:

- DenseNet Encoder: It is a convolutional neural network architecture known for its dense connectivity pattern, where each layer is connected to every other layer in a feed-forward fashion. It has 121 layers. It takes an input image and extracts high-level features through a series of convolutional layers. The architecture consists of dense blocks, each containing convolutional layers with batch normalization and ReLU activation.
- Attention-based Decoder RNN: It takes the encoded features from the DenseNet encoder as input and generates output sequences token by token. It uses a recurrent neural network (RNN) with gated recurrent units (GRUs) for sequential processing. Additionally, it incorporates an attention mechanism to selectively attend to different parts of the input features at each decoding step, enhancing the model's ability to capture relevant information for generating accurate outputs.

Evaluation Algorithm: After the LaTeX expressions are obtained, they undergo evaluation using an algorithm 2, which parses and simplifies these expressions. Users are presented with two evaluation options:

1. Direct Evaluation: The algorithm solely assesses the final answer provided by the student.
2. Detailed Evaluation: This method evaluates each step provided by the student, assigning a weight of "n%" to the steps and "100-n%" to the final answer. This asks the user to enter the minimum number of steps to ensure a uniform evaluation process for all students. However, this type of evaluation is applicable only to problems such as polynomials, calculus, trigonometry, etc., where step-by-step comparison is feasible. Users can opt for this evaluation to ensure that students have independently solved the problem by following a logical sequence of steps.

The evaluator has the flexibility to choose between referencing their own solution or evaluating the student's answer independently. If the algorithm references the evaluator's answer against the student's answer, the final answers

Algorithm 2: Evaluation of Student's Mathematical Answers

```
1: INPUT:
2: student_solution_list - List of student's LaTeX expressions (steps of the solution)
3: evaluator_solution_list - (Optional) List of evaluator's LaTeX expressions for
   comparison
4: evaluation_mode - "Direct" or "Detailed" evaluation mode
5: min_steps - (Optional) Minimum number of steps required for detailed evaluation
6: OUTPUT:
7: Correctness percentage
8: Feedback on incorrect steps (if detailed evaluation)
9: if evaluation_mode == "Direct" then
10:    student_final_answer ← student_solution_list[-1]   {retrieve student's final
       answer}
11:    evaluator_final_answer ← evaluator_solution_list[-1]   {retrieve evaluator's
       final answer}
12:    if student_final_answer == evaluator_final_answer then
13:       return 100          {Final answer is correct}
14:    else
15:       return 0          {Final answer is incorrect}
16:    end if
17: else if evaluation_mode == "Detailed" then
18:    correctness ← 0, feedback ← []
19:    total_steps ← length(evaluator_solution_list)
20:    min_steps ← length(student_solution_list)
21:    if min_steps > total_steps or min_steps = None then
22:       feedback ← "Minimum steps criteria not met."
23:       return 0, feedback
24:    else
25:       total_steps ← max(total_steps, min_steps)
26:    end if
27:    for each step[i] in student_solution_list do
28:       correct_step ← 0
29:       for each step[j] in evaluator_solution_list do
30:          if step[i] == step[j] then
31:             correctness=correctness+(n/total_steps)   {n% marks for total steps}
32:             correct_step = correct_step + 1
33:          end if
34:       end for
35:       if correct_step has no change then
36:          feedback.append("Step[i] is incorrect")
37:       end if
38:    end for
39:    if student_solution_list[-1] == evaluator_solution_list[-1] then
40:       correctness ← correctness + (100 - n)
41:    end if
42:    return correctness
43: end if
```

from both answer scripts are compared directly. For stepwise evaluation, each step in the student's answer is reduced to its canonical form and then evaluated. Upon evaluation, the algorithm returns the correctness of the answer as a percentage out of 100. In the case of a detailed evaluation, it also provides feedback on the correctness of individual steps, indicating any incorrect step numbers.

4 Datasets and Preprocessing

Multiple datasets were collected and merged to form a comprehensive dataset. The dataset included three fields: Student answer, Reference answer and Grade (treated as a label). In this section we discussed various datasets used for subjective and mathematical answers evaluation:

4.1 Datasets for Subjective Answers

The dataset's attributes provide information about the short answer responses, including the questions and answers.

Mohler Dataset [4] The Mohler dataset, consisted of questions from introductory computer science assignments and responses from undergraduate students at the University of North Texas. With 2273 student answers collected across 80 questions from 31 enrolled students, the dataset offered a diverse array of responses. Each answer was independently graded by two graduate student judges, using a scale of 0 to 5. This gold standard grading, coupled with variations between annotators and a bias towards correct answers, provided valuable insights into the challenges of automated grading.

DigiKlasaur-ASAG Dataset [5] The DigiKlasaur-ASAG dataset, sourced from GitHub, constitutes a rich repository of educational assessment data aimed at facilitating research endeavors in automated grading systems. Comprising a comprehensive array of attributes, this dataset offers invaluable insights into student responses, reference answers, and associated grading metrics. The dataset has a total of 646 question answers. The answers were graded by a single human judge, using an integer scale from 0(completely incorrect), 1(partially correct) and 2(perfect answer).

4.2 Datasets for Mathematical Answers

CROHME [6]: The Competition for the Recognition of Online Handwritten Mathematical Expressions (CROHME) dataset is a cornerstone resource for researchers developing systems that recognize handwritten mathematical expressions. It contains more than 10,000 expressions written by people from various countries, offering a diverse range of handwriting styles. The expressions themselves come from a corpus designed to encompass a variety of mathematical concepts, ensuring the dataset covers a broad spectrum of mathematical notation, from basic arithmetic to more complex functions.

4.3 Data Preprocessing

The initial phase of the implementation of the automated classification system involves data pre-processing. The dataset is loaded from a CSV file, containing student answers, reference answers, and corresponding grades.

Prior to model training, the student answers and reference answers undergo preprocessing steps, including punctuation removal and text conversion to lowercase. For WMD-based algorithm stopword removal and lemmatization were performed, but, for BERT-based models stopwords were retained to provide contextual information, especially for models like BERT, where stopwords contribute to understanding negation words. The preprocessing steps of mathematical questions is expalined in section 3.2.

5 Experimental Results

This section discusses parameters used during training of models and summarizes the results obtained from various approaches to assess subjective and mathematical answers.

5.1 Training of Models:

The training process involved multiple epochs, during which the model learned from the data. Within each epoch, training was performed in batches to prevent memory errors. During the training:

- Loss metrics were calculated at the end of each batch.
- Model weights were adjusted based on these metrics using an optimizer.
- The optimizer step involved updating the model's parameters to minimize the loss function.
- A learning rate scheduler was applied to adjust the learning rate over time, optimizing training efficiency.

After each epoch, the model's performance was evaluated using the validation set, and the loss was recorded for further analysis.

Inferences from Training: Observations and inferences were made based on the logged data during training:

- Data Distribution: The data distribution was heavily skewed towards higher grades, affecting model performance. To address this, adjustments were made to balance the dataset.
- Optimal Epochs: Initial experiments with eight epochs showed stagnant or increasing validation loss, indicating overfitting. Upon analysis of training and validation set losses, the number of epochs was reduced to four to prevent overfitting and improve generalization.

5.2 Results: Keyword Matching and WMD Algorithm:

This approach utilized a combination of semantic similarity through Word Mover's Distance (WMD) and keyword matching to evaluate English-based answers. While providing a comprehensive assessment metric, it exhibited limitations in accurately capturing semantic nuances. The model achieved a hit rate of only 25.57% within a margin of 0.5 grades and 48.66% within a margin of 1 grade point.

5.3 Results: BERT with additional layers

Despite its lower computational complexity, this method yielded inferior results compared to the BERT-based sequence classification model. The model achieved a hit rate of only 24% within a margin of 0.5 grades and 50% within a margin of 1 grade point. Consequently, it was replaced by the sequence classification model in the final iteration.

5.4 Results: BERT based Sequence Classification

Upon completion of training, the trained model undergoes further evaluation on a separate test set to evaluate its effectiveness in automating the grading process. Various plots and metrics are utilized to assess the model's effectiveness in automating the grading process. These include:

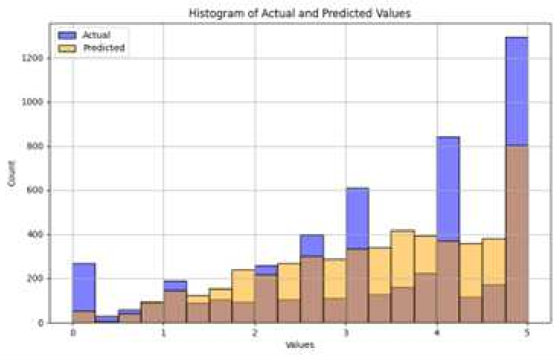
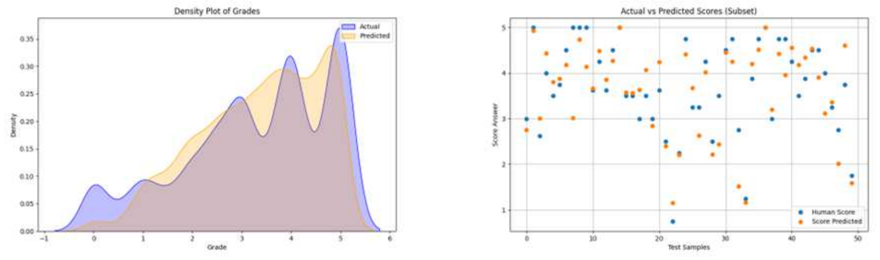


Fig. 3: Histogram of Actual vs. Predicted Values

Figure 3 represents histogram of actual vs. predicted values, it provides a visual comparison between the distribution of actual grades and the grades predicted by the model. It helps identify any discrepancies or biases in the model's predictions.

Figure 4(a) represents Density Plot of Grades, both actual and predicted grades are overlaid on this plot, allowing for a comparative analysis of their distributions. This plot aids in assessing the model's ability to accurately capture the grade distribution in the dataset. Figure 4(b) represents the actual and predicted scores assigned for the given test samples.



(a) Density Plot of Grades (b) Subset of Predicted vs Actual Grades

Fig. 4: Predicted vs Actual Grades

The post-training analysis revealed notable findings regarding the model's performance. Specifically, the model demonstrated an approximate 80% hit rate within a margin of one grade point and a 54% hit rate within a margin of 0.5 grade points across a dataset comprising 7000 answer pairs. This performance exceeded that of alternative models.

5.5 Results: Mathematical Answers

The model was trained and tested on the CROHME 2016 dataset. The model achieves a Word Error Rate (WER) loss of 17.160%, implying that around 17.160% of the words in its predicted output sequence differ from those in the ground truth. Additionally, it exhibits an Expansion Rate of 38.595%, suggesting that, on average, its output sequences are approximately 38.595% longer than the ground truth sequences. While the model demonstrates reasonable accuracy, there's room for improvement in optimizing the length of its output sequences to better align with the ground truth, thus enhancing its overall performance. Table 1 presents sample results of converting handwritten mathematical expressions (input) into LaTeX expressions (output) generated by the model.

Table 2 presents sample results illustrating the scoring of mathematical answers through both direct evaluation and detailed evaluation methods. Using direct evaluation, even though the value of x is correct, the score is 0 because the value of y is incorrect. Whereas using a detailed evaluation, since the minimum number of correct steps (i.e. 5 in this case) is met, marks are awarded to the student.

The mathematical evaluation process yielded promising results in two key metrics.

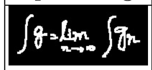
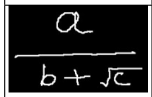
Input Image	Latex Output
	`\int{g} = \lim\_{n\rightarrow\infty}\int{g\_{n}}`
	`\frac{a}{b+\sqrt{c}}`

Table 1: Results: Expression input and the LaTeX output

- Conversion of handwritten expressions to LaTeX: The model showcased an accuracy of approximately 83%.
- Correctness of the evaluation algorithm: Given that the converted expression is correct, the algorithm evaluated all direct answers and 78% of the detailed answers correctly. Instances of incorrect evaluation in detailed answers were primarily attributed to either inaccuracies within the SymPy library or the algorithm's challenge in discerning repeated steps within a solution, where students might receive full credit for redundant steps. Overall, out of 50 full test cases, the system gave 38 correct evaluations which is 76% as shown in Figure 5.

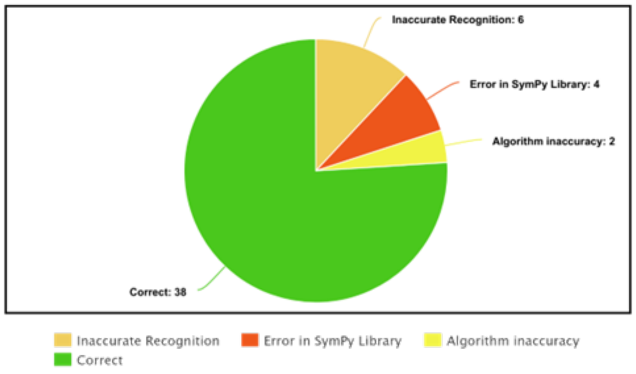


Fig. 5: Performance Results of Mathematical Evaluation

5.6 Concluding Outcomes

The evaluation of English-based answers utilizing the BERT-based sequence models demonstrates promising results as clearly seen in Table 3 and Fig. 6, giving accurate evaluations far better than the Keyword matching approach. Similarly, the recognition and evaluation of mathematical expressions using OCR

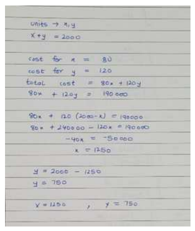
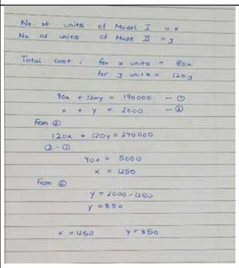
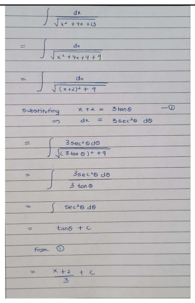
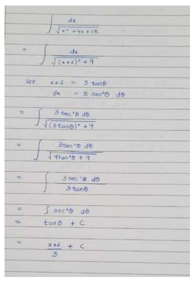
Evaluator's Answer	**Student's Answer**	**Score Assigned**
 units → x, y $x + y = 2000$ cost for x = 80 cost for y = 120 total cost = $80x + 120y$ $80x + 120y = 190000$ $80x + 120(2000 - x) = 190000$ $80x + 240000 - 120x = 190000$ $-40x = -50000$ $x = 1250$ $y = 2000 - 1250$ $y = 750$ $x = 1250$, $y = 750$	 No. of units of Model I = x No. of units of Mode II = y Total cost : for x units = $80x$ for y units = $120y$ $80x + 120y = 190000$ — ① $x + y = 2000$ — ② from ② $120x + 120y = 240000$ ② - ① $40x = 5000$ $x = 1250$ from ② $y = 2000 - 1250$ $y = 850$ $x = 1250$ $y = 850$	0 (Direct Evaluation)
 $\int \frac{dx}{\sqrt{x^2+4x+13}}$ $= \int \frac{dx}{\sqrt{x^2+4x+4+9}}$ $= \int \frac{dx}{\sqrt{(x+2)^2+9}}$ Substituting $x + 2 = 3\tan\theta$ —① $\Rightarrow dx = 3\sec^2\theta \, d\theta$ $= \int \frac{3\sec^2\theta \, d\theta}{\sqrt{(3\tan\theta)^2+9}}$ $= \int \frac{3\sec^2\theta \, d\theta}{3\tan\theta}$ $= \int \sec^2\theta \, d\theta$ $= \tan\theta + c$ From ① $= \frac{x+2}{3} + c$	 $\int \frac{dx}{\sqrt{x^2+4x+13}}$ $= \int \frac{dx}{\sqrt{(x+2)^2+9}}$ let $x+2 = 3\tan\theta$ $dx = 3\sec^2\theta \, d\theta$ $= \int \frac{3\sec^2\theta \, d\theta}{\sqrt{(3\tan\theta)^2+9}}$ $= \int \frac{3\sec^2\theta \, d\theta}{\sqrt{9\tan^2\theta+9}}$ $= \int \frac{3\sec^2\theta \, d\theta}{3\tan\theta}$ $= \int \sec^2\theta \, d\theta$ $= \tan\theta + C$ $= \frac{x+2}{3} + c$	100 (Detailed Evaluation)

Table 2: Results: Sample Scoring of Mathematical Answers

technology and the SymPy library offer a reliable method for assessing mathematical answers as described in subsection 5.5.

Additionally, evaluation metrics such as Root Mean Square Error (RMSE), Mean Absolute Error (MAE), R-squared, Hit rate, and linear weighted Kappa score are computed to quantify the model's performance:

– RMSE (Root Mean Square Error): Measures the average deviation of predicted grades from actual grades, providing a quantitative assessment of prediction accuracy.
– MAE (Mean Absolute Error): Represents the average absolute difference between predicted and actual grades, offering insights into the magnitude of prediction errors.
– R-squared (Coefficient of Determination): Indicates the proportion of variance in grades that is predictable from the model's predictions, serving as a measure of model fit to the data.
– Hit Rate: Reflects the proportion of correctly predicted grades within a specified margin of error, indicating the model's accuracy in grade prediction.
– Linear Weighted Kappa: Assesses how closely the predicted grades from a model match the actual (or true) grades, with a focus on penalizing larger discrepancies more than smaller ones.

Upon completion of training, the trained model undergoes further evaluation on a separate test set to evaluate its effectiveness in automating the grading process.

The evaluation metrics obtained include Root Mean Square Error (RMSE) of 0.87, Mean Absolute Error (MAE) of 0.62, R-squared value of 0.65, Hit Rate of 0.8, and Linear Weighted Kappa score of 0.659, which indicates substantial agreement between the actual and predicted scores.

Overall, these evaluation metrics provide comprehensive insights into the model's performance and effectiveness in automating the grading process.

Metric	**Keyword Matching**	**BERT based Sequence Classification**
RMSE	1.45	0.87
MAE	1.16	0.62
Hit Rate ± 0.5	25.57 %	53.84%
Hit Rate ± 1	48.66 %	80.45%
R-squared	0.018	0.804
Linear weighted kappa	0.082	0.659

Table 3: English Evaluation Results

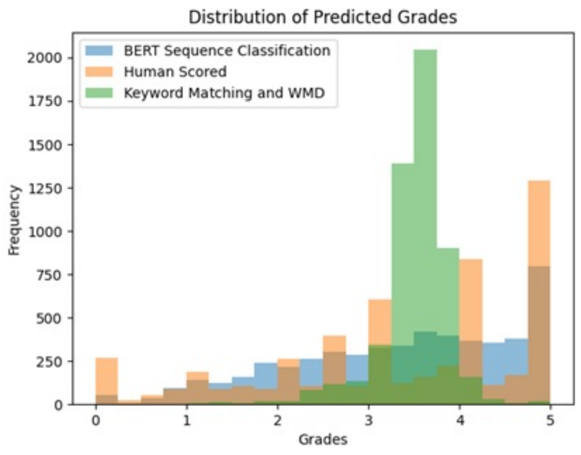


Fig. 6: Grades Density Distribution

6 Conclusion and Future Directions

In conclusion, we have developed the AASE system that is capable of evaluating both subjective as well mathematical answers and it is a significant step towards automating the grading process in education. Through the integration of an advanced BERT-based sequence classification for subjective answers, and the evaluation algorithm for mathematical answers, the system offers a reliable solution to the challenges associated with manual grading procedures. We evaluated and tested the AASE system on multiple real-world datasets, assessed it for different evaluation metrics, and achieved the accuracy of 80.45% for subjective answer evaluation and 76% for mathematical answers.

Moving forward, further optimization and refinement of the AASE system could enhance its performance and expand its applicability across different educational domains. Additionally, ongoing research and development efforts in the field of automated grading systems will continue to drive innovation and improve the efficiency and fairness of educational evaluation processes.

This AASE has scope for improvement. They are as follows:

- Improving the pre-processing part for mathematical evaluation and modifying the model in a way that it can detect the expressions without image segmentation and streamline the overall process.
- Implementing a more strict evaluation algorithm for the detailed evaluation of mathematical answers. This could involve refining the criteria for assessing student responses, perhaps by detecting and penalizing redundant steps more effectively. A more rigorous algorithm would lead to more precise evaluations and better feedback for students.
- During generalization training for English-based assessment, we faced challenges with dataset density, notably spikes at grade points 4 and 5. Conversely, limited data at lower grade points led to lenient grading. Solutions include using uniformly distributed datasets or tailoring data distribution to suit user needs, aiding the model in adapting its training.

References

1. `https://www.hindustantimes.com/education/news/evaluation-of-answer-sheets-unhappy-aktu-students-take-to-twitter\-101624103848937.html`, [Accessed 28-12-2024]
2. Driving India's growth story through the brightest young minds - Times of India — timesofindia.indiatimes.com. `https://timesofindia.indiatimes.com/business/india-business/driving-indias-growth-story-through-the-brightest-young-minds/articleshow/113060795.cms`, [Accessed 28-12-2024]
3. Exam result puzzle: Right answer, wrong evaluation — bangaloremirror.indiatimes.com. `https://bangaloremirror.indiatimes.com/bangalore/others/exam-result-puzzle-right-answer-wrong-evaluation/articleshow/109074577.cms`, [Accessed 28-12-2024]
4. (2024), `https://www.kaggle.com/datasets/abdokamr/mohler`
5. (2024), `https://github.com/DigiKlausur/ASAG-Dataset`
6. (2024), `https://www.isical.ac.in/~crohme/CROHME_data.html`
7. Askarbekuly, N., Aničić, N.: Llm examiner: automating assessment in informal self-directed e-learning using chatgpt. Knowledge and Information Systems pp. 1–18 (2024)

8. Bashir, M.F., Arshad, H., Javed, A.R., Kryvinska, N., Band, S.S.: Subjective answers evaluation using machine learning and natural language processing. IEEE Access **9**, 158972–158983 (2021)
9. Bernard, H.R., Ryan, G.: Text analysis. Handbook of methods in cultural anthropology **613** (1998)
10. Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.t., Choi, Y., Liang, P., Zettlemoyer, L.: Quac: Question answering in context. arXiv preprint arXiv:1808.07036 (2018)
11. Cochran, K., Cohn, C., Rouet, J.F., Hastings, P.: Improving automated evaluation of student text responses using gpt-3.5 for text data augmentation. In: International Conference on Artificial Intelligence in Education. pp. 217–228. Springer (2023)
12. Condor, A., Litster, M., Pardos, Z.: Automatic short answer grading with sbert on out-of-sample questions. International Educational Data Mining Society (2021)
13. Das, B., Majumder, M., Phadikar, S., Sekh, A.A.: Automatic question generation and answer assessment: a survey. Research and Practice in Technology Enhanced Learning **16**(1), 5 (2021)
14. Devlin, J.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
15. Funayama, H., Asazuma, Y., Matsubayashi, Y., Mizumoto, T., Inui, K.: Reducing the cost: Cross-prompt pre-finetuning for short answer scoring. In: International conference on artificial intelligence in education. pp. 78–89. Springer (2023)
16. Gao, R., Merzdorf, H.E., Anwar, S., Hipwell, M.C., Srinivasa, A.: Automatic assessment of text-based responses in post-secondary education: A systematic review. Computers and Education: Artificial Intelligence p. 100206 (2024)
17. Gupta, P., Gupta, V.: A survey of text question answering techniques. International Journal of Computer Applications **53**(4) (2012)
18. Himi, S.T., Monalisa, N.T., Sultana, S., Afrin, A., Hasib, K.M.: Automated exam script checking using zero-shot llm and adaptive generative ai. In: 2024 IEEE International Conference on Computing, Applications and Systems (COMPAS). pp. 1–6. IEEE (2024)
19. Jin, C., He, B., Hui, K., Sun, L.: Tdnn: a two-stage deep neural network for prompt-independent automated essay scoring. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1088–1097 (2018)
20. Nandini, V., Uma Maheswari, P.: Automatic assessment of descriptive answers in online examination system using semantic relational features. The Journal of Supercomputing **76**(6), 4430–4448 (2020)
21. Patil, R., Kulkarni, A.A., Ghatage, R., Endait, S., Kale, G., Joshi, R.: Automated assessment of multimodal answer sheets in the stem domain. arXiv preprint arXiv:2409.15749 (2024)
22. Rockwell, G.: What is text analysis, really? Literary and linguistic computing **18**(2), 209–219 (2003)
23. Senanayake, C., Asanka, D.: Rubric based automated short answer scoring using large language models (llms). In: 2024 International Research Conference on Smart Computing and Systems Engineering (SCSE). vol. 7, pp. 1–6. IEEE (2024)
24. Sultan, M.A., Salazar, C., Sumner, T.: Fast and easy short answer grading with high accuracy. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1070–1075 (2016)
25. Vaswani, A.: Attention is all you need. Advances in Neural Information Processing Systems (2017)

26. Yasin Sharif, A., Ravindhar, N.: Improved evaluator for subjective answers using natural language processing. In: International Conference on Computational Intelligence in Data Science. pp. 98–109. Springer (2024)
27. Zeng, Z., Li, L., Guan, Q., Gašević, D., Chen, G.: Generalizable automatic short answer scoring via prototypical neural network. In: International Conference on Artificial Intelligence in Education. pp. 438–449. Springer (2023)
28. Zhang, L., Huang, Y., Yang, X., Yu, S., Zhuang, F.: An automatic short-answer grading model for semi-open-ended questions. Interactive learning environments **30**(1), 177–190 (2022)

Author contributions

M.S. and M.R. conducted the research and experiments under the supervision of S.K. S.K. and S.M. wrote the manuscript. All authors reviewed and approved the final version of the manuscript.

Funding

Open access fundingis expected to be supported and facilitated by Shiv Nadar University, Dadri, Uttar Pradesh, India.

Data Availability

All datasets used in this study are publicly available. Mohler Dataset is available at `https://www.kaggle.com/datasets/abdokamr/mohler`. DigiKlasaur-ASAG Dataset is available at `https://github.com/DigiKlausur/ASAG-Dataset`; and CROHME dataset is available at `https://www.isical.ac.in/~crohme/CROHME_data.html`.