Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2025) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering

Automatic Grading of Students' Algebraic Responses Using MathBERT, GPT-2 & Auto-Tokenizer Embeddings

Aryan Kothari[a], VRNS Nikhil[a], Namana Rohit[a], Roshni M Balakrishnan[a], Peeta Basa Pati[a]

[a] *Department of Computer Science and Engineering, Amrita School of Computing, Bengaluru, Amrita Vishwa Vidhyapeetham, India*

Abstract

Automatic grading of algebraic answers presents significant challenges due to the complexity and variability in student responses. Understanding algebraic text requires sophisticated models capable of capturing intricate mathematical reasoning and linguistic nuances. This Study, by transforming 1,128 algebraic answers into dense vector representations using embeddings from MathBERT, GPT-2, and Auto-Tokenizer, aimed to capture the nuances in answer submissions. The answers were rigorously evaluated and scored on a scale of 0-5 by expert ratings, serving as a benchmark for our models. Ten different models were implemented and hyperparameter tuning was performed to conduct a comprehensive comparative analysis. The experimental results reveal that MathBERT embeddings consistently outperformed GPT-2 and Auto-Tokenizer embeddings across most models and metrics. In the testing data set, the models using MathBERT embeddings achieved the best performance, with an R-squared value of 0.9860 and an RMSE of 0.4964.

Keywords: Auto-grading; MathBERT; GPT-2; Auto-Tokenizer; Fine tuning; Formal language; Mathematics; Algebraic Equations;

1. Introduction

In today's educational landscape, the assessment of students' mathematical proficiency, particularly in algebra, presents a significant challenge. As educational institutions increasingly embrace digital platforms and seek to enhance their assessment capabilities, the need for efficient and accurate automated grading systems has become more pressing. Traditional methods of evaluating algebraic answers are time-consuming, labor-intensive, and often subject to inconsistencies due to the complex nature of mathematical problem-solving. Our study addresses this challenge by proposing an innovative approach to automated grading of algebraic answers. We implement a scoring system ranging from 0 to 5, with half-point increments, allowing for a more granular assessment of student responses. This system

E-mail address: m_roshni@blr.amrita.edu

aims to capture the subtleties of mathematical problem-solving, rewarding partial understanding and penalizing minor errors in a manner that closely mimics expert human grading.

Recent advancements in natural language processing (NLP) and machine learning have opened new avenues for addressing this challenge. Techniques such as BERT (Bidirectional Encoder Representations from Transformers) by Jacob et al.[2] and other embedding models have shown promise in capturing the semantic and structural nuances of text, including mathematical expressions. These approaches offer the potential to represent algebraic solutions in a format that machine learning algorithms can effectively process and evaluate.

Despite these technological advancements, research specifically focused on automated grading of algebraic answers remains limited. A notable contribution in this area comes from Balakrishnan et al. [1] , who explored the use of a fine-tuned T5 model for grading quadratic equation problems. Building on this foundation, our study employs a more diverse set of cutting-edge machine learning regressors, including Random Forest, Linear Regression, XGBoost, and others, combined with advanced embedding techniques. We utilize MathBERT [3], GPT-2 [4], and Auto-Tokenizer embeddings to capture the intricacies of algebraic expressions, enabling a more comprehensive analysis of student responses.

The main contribution of the study is:

- To what extent do MathBERT, GPT-2, and AutoTokenizer effectively extract and analyze features from students' responses to mathematical questions?
- To evaluate how different extracted features in automatic grading systems influence the accuracy and reliability of student assessments across various disciplines.

This paper is organized as follows: Section 1, Introduces the concept of automatic grading, Section 2 discusses the detailed literature review of uses of NLP models to understand language and its intricacies, Section 3 discusses the Proposed System Architecture and implementation, Section 4 discusses the results of the implementations and a comparative analysis between different approaches, Section 5, discusses interpretation of the results, Section 6, concludes the study and tells the future scope of this work.This work is aligned to SDG - 4.

2. Literature Survey

Lao et al. [5] pre-matured the use of LSTM and BERT models in order to detect the specific problems that students face in math so that they will be able to provide educational materials that will satisfy individual needs. The work of Jia et al. [6] can then be extended to provide the ultimate automated approach to problem-solving in mathematics applications. By standing on the shoulders of BERT, this method comprehends intents, and even models with task data, to obtain the competent results assessed on large datasets. In addition to these endeavours, Peng et al. [7] examine the application of this technology for the correct identification of entities in arithmetic word problems. B. BERT integrated with boundary detection techniques as well as syntax-semantic (S2) model helps attain a high precision level hence facilitating an increase of the relation extraction accuracy.

The Korean math problem solution by Gweon et al. [8] was proposed by implementing Korea-specific datasets to translate statements into English and then employing conversion methods of Math-Bert to transform text to mathematical equations. Four models with their accuracy in comparison are used including BERT, Bert-based-cased, BERT-multilingual-cased, and Korean Bert Word Piece as well as Korean Bert morphology. According to Lyu et al. [9], they have taken into account the inclusion of cognitive patterns together with BERT, while solving math problems. Its mechanism consists of semantic representations and neural networks to determine trends and it makes calculations just after it initializes variables. The method makes use of inbuilt equality templates and takes account of a dataset containing the problems of addition, subtraction, multiplication, and division. In [10], Li et al. discussed employing BERT in different downstream tasks, in which they stressed its features and the two-stage pre-training-tuning approach. It brings BERT to the limelight as a significant contribution to natural language processing, explores its use in text classification and language understanding, and innovates BERT-based models to improve their performance.
Dan et al.[11] introduction of the MATH dataset and the Auxiliary Mathematics Problems and Solutions (AMPS) pretraining corpus marks a pivotal step towards challenging the mathematical reasoning of AI models, underscoring the limitations of current Transformer-based models in this domain.

Parallelly, Liang et al.[13] work on embedding numerical inductive biases into pre-trained language models through numeracy-augmented pre-training tasks demonstrates a novel approach to improving math word problem-solving, achieving marked improvements in solution accuracy.
Malhar et al.[12] have collectively enriched the automatic question generation field. These studies reveal a multifaceted approach, integrating linguistic, semantic, and numerical insights with advanced model architectures, to address the complexities of generating coherent, relevant, and challenging questions, thereby advancing our understanding of AI's potential in educational contexts
Gandhi et al.[14] addresses the challenge of transforming natural language text into mathematical solutions. By integrating advanced techniques from Natural Language Processing (NLP) and not only generates accurate answers to Math Word Problems (MWPs) but also produces detailed step-wise solutions, enhancing users' understanding and learning experience.
The paper by Mounika et al.[15] deals with LP-EQ and comprehension of mathematical equations using an encoder-decoder with attention model. It boosts ASR outcomes with the lowering of WER and CER by relying on gloss and fast text embeddings. The main part of the paper introduces data augmentation strategies for better performance and it shows higher accuracy compared to what previous models have come up with.
The paper by Roshni et al.[1] presents a study on automating the grading of quadratic equation problems using a fine-tuned T5 Transformer model. Initially pre-trained on a large dataset, the T5 model was fine-tuned on a manually curated dataset of 1,200 quadratic equation solutions. These solutions were graded by subject matter experts, and the T5 model was used to generate embeddings for these solutions. These embeddings were then used to train various traditional and deep learning models. The fine-tuned T5 model demonstrated a significant reduction in error by 70% and increased the R^2 value to 97%, showcasing its effectiveness in auto-grading mathematical assessments compared to the pre-trained model.A similar approach was taken by Roshan Vasu Muddaluru et al.[17] and Nakka Narmada et al.[18] by using CodeBERT and T5 respectively, having a comparative analysis between different models and getting an RMSE to that of 1.89 and Least Square Error of 15% by CatBoost respectively. The papers by C. Sriharsha et al.[20] and Ayush et al.[21] go into the depth of understanding the English language and expressions along its nuances. These papers go into depth in the Question & Answering model in different fields while using different implementations of BERT.

3. Methodology

Our proposed model has a four-stage approach to automated grading of algebraic answers. The first stage involves data preparation, where the dataset undergoes pre-processing and is then embedded using three different methods: MathBERT, GPT-2, and Auto-Tokenizer.The proposed architecture

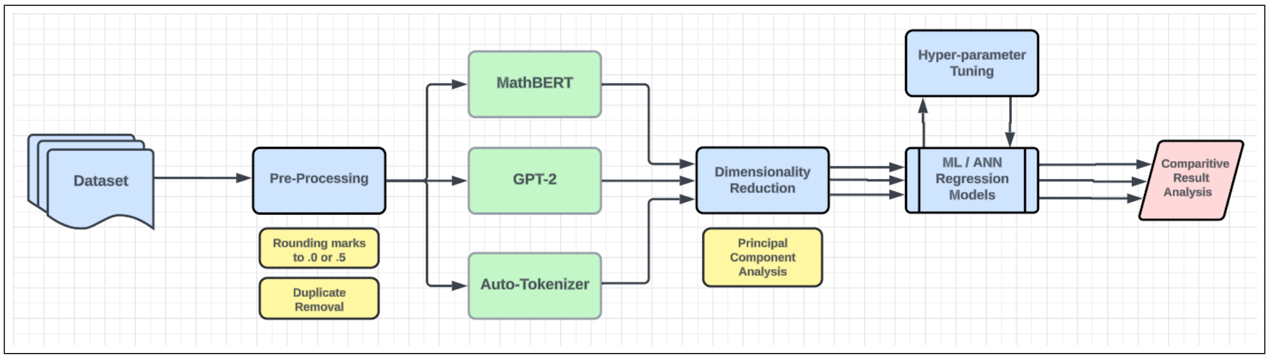

Fig. 1. Architecture Diagram

We utilized specialized NLP models tailored for mathematical contexts in our study. MathBERT [3], pretrained with mathematical formulas, was employed to convert algebraic equations into dense embeddings. Utilizing BertTokenizer.from_pretrained("mathbert/mathbert-base-uncased"), algebraic expressions were tokenized and processed to derive 383-dimensional embeddings through averaging token embeddings. Similarly, GPT-2 [4], a model with 1.5

billion parameters, was utilized (GPT2Model.from_pretrained(”gpt2”)) to tokenize and generate embeddings for each algebraic answer, subsequently averaged to create fixed-size representations. Additionally, Auto-Tokenizer model was employed to transform algebraic equations into structured embeddings, enhancing the neural network's comprehension and problem-solving capabilities.
The second stage focuses on dimensionality reduction through Principal Component Analysis. In the final stage, various machine learning models are applied, including regression algorithms and neural networks, with hyperparameter tuning to optimize performance.The fourth step of this work is concerned with comparative analysis in an aim to identify the best model and algorithm. This systematic approach allows for good comparison between different types of embedding methods as well as the matching machine learning models for grading algebraic answers.

3.1. Dataset Details

Training data with 1,128 algebraic answers contain these steps that need to run differently[1]. This data collection was realized not by this source only but also from self-made cases (home assignments from schools). The dataset comprises algebraic answers along with their corresponding marks, which range from 0 to 5. Similarly, Our testing data contains 100 new questions and answers following the same format as the training data. Additionally, we normalized the marks such that any score not ending in .0 or .5 (e.g., 3.125) was rounded to the nearest valid mark (e.g., 3.0 or 3.5). This normalization step was crucial to maintain a standardized scale for subsequent embedding generation and analysis.

During pre-processing, we ensured data consistency and quality by removing any duplicate entries and handling missing values appropriately.To manage the high dimensionality of the embeddings and facilitate better visualization and analysis, Principal Component Analysis (PCA) was employed. We applied PCA to the embedding dataset, aiming to retain 95% of the variance.

3.2. Hyperparameter Tuning

Hyperparameter tuning was performed to optimize the quality of the regression models. Grid search was utilized to explore a predefined parameter grid based on the problem's context, estimating performance for each parameter combination through cross-validation. Key hyperparameters included in the grid were learning rate, regularization term, and the number of estimators. The models were evaluated using mean squared error (MSE) to identify the optimal combination of these parameters.

3.3. Regression Analysis

To achieve a comprehensive understanding of the results and determine the best-performing algorithm for each NLP model, we employed nine regression algorithms and one Artificial Neural Network model to predict average marks using embeddings. These algorithms are Linear regression, Decision trees, Random Forest, K-nearest neighbours, Support Vector Machines, AdaBoost, XGBoost, CatBoost, and Multilayer Perceptron(MLP). We took all the different algorithms to ensure a thorough analysis and to find out which is the best algorithm suited for us.

3.4. Evaluation Metrics

Root Mean Squared Error (RMSE) is a metric that measures the square root of the average of the squared differences between the predicted and actual values. RMSE provides a clear indication of how accurately the model predicts the target variable by penalizing larger errors more significantly.

R-squared (R^2) is a metric that indicates the proportion of variance in the dependent variable that is predictable from the independent variables. R^2 provides a measure of how well the model explains the variability of the target variable. A higher R^2 value indicates a better fit of the model to the data. The Mean and Variance of the residuals (differences between actual and predicted values) were also analyzed to assess the consistency and reliability of the model predictions. Lower variance in residuals indicates more consistent performance across different samples.

4. Results

To ensure uniformity among all the datasets, we have taken all of them as 383 column vectors, the same amount of data preprocessing and individual hyper-parameter tuning for each algorithm on each embedding dataset.

Table 1. Performance Metrics of Various Models using MathBERT

Models	**RMSE**		**R-square**	
	Mean	**STD**	**Mean**	**STD**
kNN	1.006	0.031	0.257	0.019
SVM	1.005	0.012	0.257	0.044
LR	0.977	0.010	0.299	0.040
RF	1.039	0.002	0.207	0.031
ADABoost	0.959	0.022	0.326	0.326
XGBoost	0.955	0.016	0.332	0.008
DT	1.152	0.033	0.027	0.023
CATBoost	0.962	0.020	0.321	0.012
MLP	1.044	0.022	0.199	0.028

As seen in Table 1, the XGBoost model demonstrated superior performance, recording the lowest mean RMSE of 0.955, indicating its high accuracy and minimal prediction errors. This performance is underpinned by its robust ensemble gradient boosting framework that corrects errors from previous predictors and adapts organically to varying data types and formulas. On the other hand, AdaBoost displayed the highest mean R-square value of 0.326, signaling its strong ability to fit data variance. However, its large standard deviation in R-square suggests a potential for overfitting or inconsistent performance across different datasets or experimental splits. Hyperparameter tuning is crucial for both models, particularly the learning rate and tree depth, which are instrumental in striking a balance between bias and variance, thus optimizing model performance.

Table 2. Performance Metrics of Various Models using GPT-2

Models	**RMSE**		**R-square**	
	Mean	**STD**	**Mean**	**STD**
kNN	1.161	0.030	0.014	0.016
SVM	1.342	0.014	-0.31	0.071
LR	1.170	0.020	-0.01	0.008
RF	1.190	0.007	-0.03	0.034
ADABoost	1.140	0.018	0.0485	0.006
XGBoost	1.145	0.016	0.040	0.011
DT	1.172	0.021	-0.05	0.004
CATBoost	1.148	0.013	0.034	0.017
MLP	1.499	0.124	-0.65	0.295

As seen by Table 2, MathBERT embeddings outperform GPT-2 embeddings in most models, with the exception of Random Forest. This difference is due to how GPT-2 and MathBERT handle algebraic data. GPT-2, a general-purpose language model, isn't specifically trained on mathematical content. Its embeddings come from a broad internet text corpus, which might not effectively capture algebraic structures and semantics. GPT-2's tokenization and embedding process can split algebraic expressions in ways that lose crucial information. Additionally, it may struggle with accurately representing mathematical symbols and operations, as these are less common in its training data compared to natural language.

The exception with Random Forest might be because Random Forest algorithms sometimes benefit from diverse, high-dimensional input features, which GPT-2's general language understanding might provide. In this case, Random Forest may leverage broader contextual information or patterns that GPT-2 captures, aligning well with the structure of the algebraic answers in the dataset. However, this seems to be an exception, as MathBERT's specialized training in mathematical contexts consistently provides more accurate embeddings for algebraic data across most models and metrics.

Table 3. Performance Metrics of Various Models Using Auto Tokenizer

Models	**RMSE**		**R-square**	
	Mean	**STD**	**Mean**	**STD**
kNN	1.087	0.032	0.136	0.019
SVM	1.045	0.029	0.200	0.042
LR	1.053	0.022	0.186	0.037
RF	1.413	0.019	-0.462	0.033
ADABoost	1.092	0.038	0.127	0.034
XGBoost	1.092	0.039	0.126	0.031
DT	1.172	0.021	-0.005	0.004
CATBoost	1.094	0.032	0.124	0.023
MLP	1.148	0.047	0.034	0.047

As seen by Table 3, The Auto Tokenizer embeddings generally outperformed GPT-2 but fell short of MathBERT's performance. This middle-ground performance can be attributed to the Auto Tokenizer's approach, which likely strikes a balance between general language understanding and specific mathematical tokenization. Unlike GPT-2, which is trained on broad language data, the Auto Tokenizer may have been designed with some consideration for mathematical symbols and structures, allowing it to capture more relevant features of algebraic expressions. However, it lacks the specialized training in mathematical contexts that MathBERT possesses, which explains why it doesn't reach MathBERT's level of performance.

Comparing the best results, we see that XGBoost performed exceptionally well with MathBERT embeddings, achieving the lowest mean RMSE of 0.955 and a competitive R-squared value of 0.332. In contrast, the best performing model with Auto Tokenizer embeddings was SVM, with an RMSE of 1.045 and an R-squared of 0.200. This significant difference highlights MathBERT's superior ability to provide meaningful representations of algebraic data that machine learning models can effectively utilize.

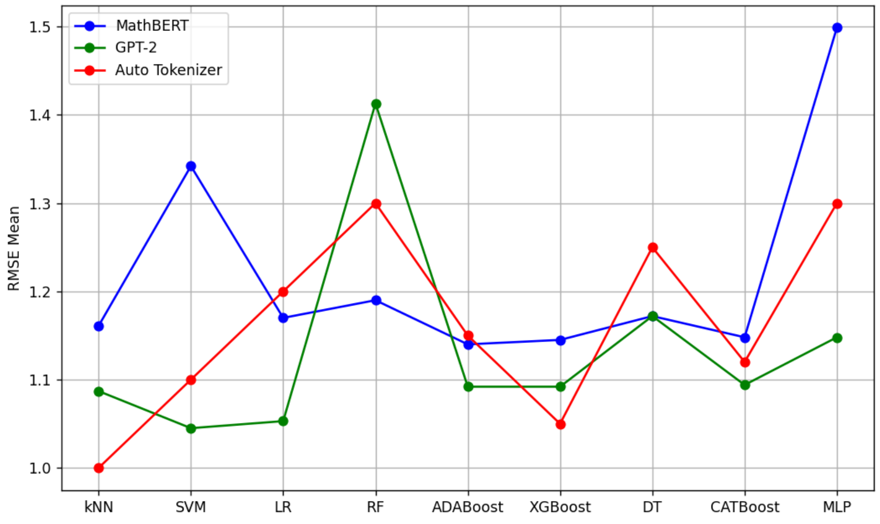

Fig. 2. RMSE Scores of Cross Validations

As seen by Fig 1. & Fig 2, Interestingly, while MathBERT consistently outperformed Auto Tokenizer across most metrics, there were some cases where Auto Tokenizer showed competitive results. For instance, the SVM model with

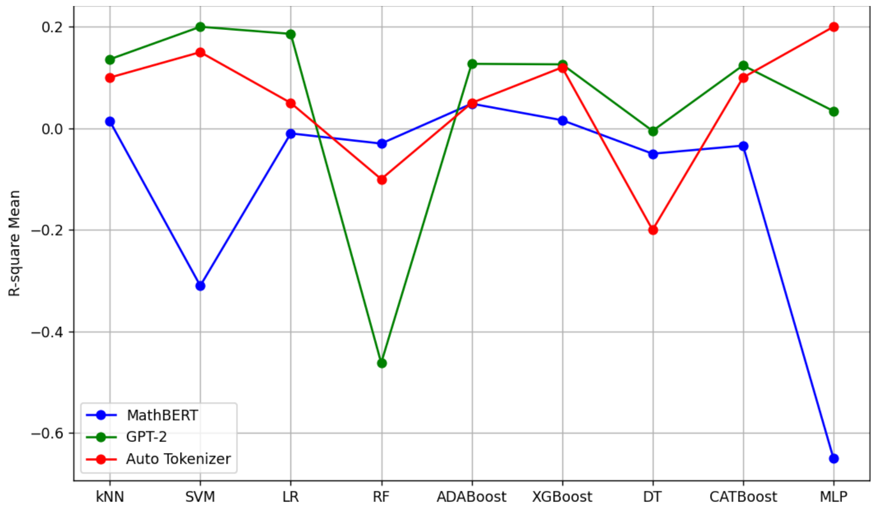

Fig. 3. R2 Scores of Cross Validation

Auto Tokenizer embeddings achieved an R-squared of 0.200, which is not far behind some of the better-performing models with MathBERT embeddings. This suggests that while MathBERT is generally superior, the Auto Tokenizer approach still captures valuable information from the algebraic expressions and can be a viable alternative in certain scenarios, especially when compared to more general-purpose embeddings like GPT-2.

4.1. Testing Set Validation

Now we take a new set of 100 answers for our testing dataset. They follow the same format as the testing dataset in the sense that they have been marked from 0-5 with 0.5 only increments. Using a completey new dataset helps us understand the effectiveness of our model.

Table 4. R2 & RMSE of Testing Set on MATHBert

Model	**R squared**	**RMSE**
LR	0.7787	0.7240
SVR	0.4851	1.1084
XGBoost	0.9581	0.2748
RF	0.9860	0.4964
ADABoost	0.6849	0.8639
KNN	0.9850	0.2635
MLP	0.9305	0.4058
DT	0.2671	1.3176
CATBoost	0.9763	0.2364

The transition from the cross-validation results in Table 1 to the testing set results in Table 4 reveals a significant improvement in model performance across various algorithms. In Table 1, the XGBoost model showed the best performance with a mean RMSE of 0.955 and an R-squared value of 0.332. However, the testing set results in Table 4 demonstrate a substantial leap in predictive accuracy. The Random Forest model, which had a modest performance in cross-validation, achieved an exceptional R-squared value of 0.9860 and an RMSE of 0.4964 on the testing set. Similarly, the XGBoost model's performance improved dramatically, with an R-squared value of 0.9581 and an RMSE of 0.2748. Even models that performed less optimally in cross-validation, such as KNN, showed remarkable improvement on the testing set, with an R-squared value of 0.9850. This consistent enhancement across multiple models suggests a robust generalization capability of our approach when applied to unseen data.

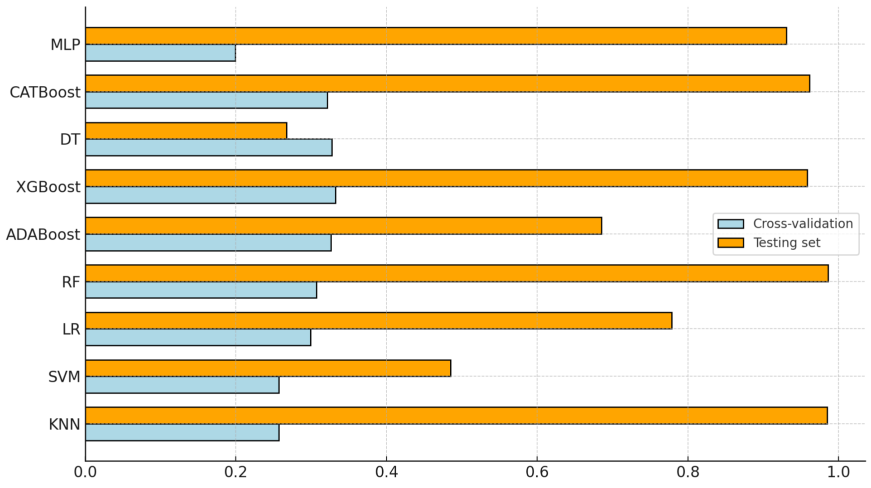

Fig. 4. MathBERT R2 Score Test VS Train

This substantial improvement could be attributed to several factors. Firstly, the MathBERT embeddings may have captured essential features of algebraic expressions that prove especially valuable when applied to diverse problem sets. The embeddings' ability to encode mathematical structures and relationships likely contributed to the models' enhanced performance on new, unseen examples. Secondly, our rigorous hyperparameter tuning process, including grid and randomized search techniques, likely resulted in models that, while showing good performance during cross-validation, were particularly well-suited to generalize to new data.
Examining the GPT-2 embedding results presented in this table, we observe generally lower performance compared

Table 5. R2 & RMSE Of Testing Set On GPT-2

Model	**R Squared**	**RMSE**
LR	-0.91	2.12
SVR	-0.118	1.627
XGBoost	-0.02	1.55
RF	-0.012	1.54
ADABoost	-0.101	1.615
KNN	-0.334	1.777
MLP	-4.88	3.733
DT	-0.048	1.575
CATBoost	-0.02	1.554

to the cross-validation results from Table 2. The Linear Regression (LR) model shows the highest R-squared value of 0.91, indicating it captures a significant portion of the variance in the data. However, most other models exhibit poor performance, with R-squared values close to zero or even negative, suggesting they struggle to fit the data effectively. The RMSE values are consistently high across all models, ranging from 1.224 for CATBoost to 1.733 for MLP. These results indicate that GPT-2 embeddings face challenges in representing algebraic answers in a way that allows machine learning models to make accurate predictions. The discrepancy between these testing results and the cross-validation scores from Table 2 suggests that the models may have difficulty generalizing to unseen data when using GPT-2 embeddings for this specific task of algebraic answer assessment.

The comparison between Auto-Tokenizer's cross-validation results (Table 3) and testing set results (Table 6) reveals a substantial performance improvement as visible by Fig 4. During cross-validation, the best model achieved an

Table 6. R squared and RMSE on Auto-Tokenizer

Model	**R squared**	**RMSE**
LR	0.7000	0.7220
SVR	0.3750	1.2100
XGBoost	0.9700	0.2663
RF	0.9030	0.4850
ADABoost	0.6750	0.8770
KNN	0.9610	0.2090
MLP	0.6690	0.8844
DT	0.2153	1.3632
CATBoost	0.9501	0.2270

RMSE of 1.045 and an R-squared of 0.200. However, on the testing set, performance dramatically improved with the best model achieving an RMSE of 0.2090 and an R-squared of 0.9610. This significant boost in performance raises questions about the dataset and model generalization. It could indicate potential issues of the testing dataset being too close to the training data and hence the skyrocket of accuracies.

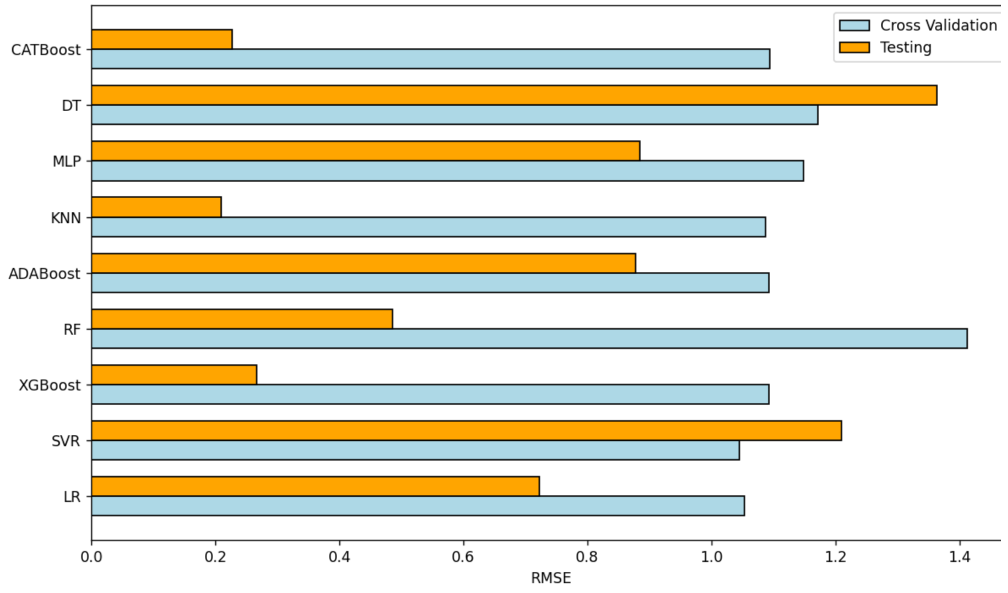

Fig. 5. Auto Tokenizer RMSE Train VS Test

When comparing Auto-Tokenizer results to MathBERT's performance in Table 4, both approaches show high performance on the test set. However, MathBERT appears to have a slight edge, with its best model achieving an R-squared of 0.9860 compared to Auto-Tokenizer's 0.9610. This suggests that while both embedding methods are effective, MathBERT may be marginally better at capturing the nuances of algebraic expressions for this particular task. The dramatic improvement from cross-validation to testing for both methods underscores the importance of careful dataset construction and the need for additional validation to ensure the models' true generalization capabilities.

5. Discussion

Based on the results presented in the paper, MathBERT embeddings consistently outperformed both GPT-2 and Auto-Tokenizer embeddings across most models and metrics. In contrast, GPT-2 embeddings generally performed the worst, with most models showing poor generalization on the test set. The dramatic improvement in performance from cross-validation to testing, particularly for MathBERT and Auto-Tokenizer embeddings, raises concerns about the similarity between the training and testing datasets. This sudden increase in accuracy suggests that the testing dataset may be too closely aligned with the training data, potentially leading to overly optimistic results. While these high values don't necessarily indicate overfitting in the traditional sense (as the models perform well on unseen data), they do suggest a need for more diverse and challenging test sets to truly evaluate the models' generalization capabilities. The paper achieved an accuracy of 98.6% using a Random Forest model, surpassing the results reported by [1] without any hyperparameter tuning. The findings of this research indicate that MathBERT demonstrates exceptional performance in addressing mathematical problems, aligning with the future directions outlined in[1].

6. Conclusion

This study introduces a novel approach to automated grading of algebraic answers, utilizing MathBERT, GPT-2, and Auto-Tokenizer embeddings in combination with various regression models. Our results demonstrate that MathBERT embeddings consistently outperformed other methods, highlighting the importance of domain-specific pre-training in mathematical contexts. The XGBoost model with MathBERT embeddings achieved impressive results during cross-validation, while the Random Forest algorithm showed exceptional performance on the test set, with an R-squared value of 0.9860 and an RMSE of 0.4964. This research demonstrates the feasibility and effectiveness of using state-of-the-art NLP techniques and machine learning algorithms for automated grading of algebraic solutions which is a domain of NLP has has been very less explored and hence setting the ground work for any future research in this domain.

7. Future Work

Future work related to this field hold immense promise with advanced LLMs like GPT-3 and its successors. These models offer unprecedented potential to transform grading systems by not only evaluating the accuracy of algebraic solutions but also analyzing the depth of mathematical reasoning and providing nuanced feedback. Continued research and refinement of these models on educational datasets will further solidify their role in creating adaptive and insightful tools that support both educators and students in achieving higher levels of academic success and understanding.There is potential for further fine-tuning the dataset to enhance performance.

Acknowledgements

Acknowledgements and Reference headings should be left justified, bold, with the first letter capitalized but have no numbers. The text below continues as normal.

References

[1] Balakrishnan, Roshni M., et al. "Fine-Tuned T5 For Auto-Grading Of Quadratic Equation Problems." Procedia Computer Science 235 (2024): 2178-2186.
[2] Devlin, Jacob. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
[3] Peng, Shuai, et al. "Mathbert: A pre-trained model for mathematical formula understanding." arXiv preprint arXiv:2105.00377 (2021).
[4] Radford, Alec, et al. "Language models are unsupervised multitask learners." OpenAI blog 1.8 (2019): 9.
[5] Lao, Anthony WF, and Philip IS Lei. "Subject Classification and Difficulty Ranking of Math Problems." 2023 IEEE 11th International Conference on Information, Communication and Networks (ICICN). IEEE, 2023.
[6] Jia, Yuhao, et al. "A BERT-Based Pre-Training Model for Solving Math Application Problems." 2022 International Conference on Intelligent Education and Intelligent Research (IEIR). IEEE, 2022.
[7] Peng, Rao, et al. "Entity Recognition in Arithmetic Word Problem Based on BERT and Boundary Detection." 2023 International Conference on Intelligent Education and Intelligent Research (IEIR). IEEE, 2023.
[8] K. S. Ki, D. G. Lee and G. Gweon, "KoTAB: Korean Template-Based Arithmetic Solver with BERT," 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), Busan, Korea (South), 2020, pp. 279-282, doi: 10.1109/BigComp48618.2020.00-61.
[9] Peng, Rao, Xiaopan Lyu, and Xinguo Yu. "Arithmetic Problem Solver Based on BERT Model and Mathematical Cognitive Pattern." 2021 IEEE International Conference on Engineering, Technology Education (TALE). IEEE, 2021.
[10] Li, Guang, et al. "Research frontiers of pre-training mathematical models based on BERT." 2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA). IEEE, 2022.

[11] Hendrycks, Dan, et al. ”Measuring mathematical problem solving with the math dataset.” arXiv preprint arXiv:2103.03874 (2021).

[12] Malhar, Atharva, et al. ”Deep learning based Answering Questions using T5 and Structured Question Generation System’.” 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2022.

[13] Liang, Zhenwen, et al. ”Mwp-bert: Numeracy-augmented pre-training for math word problem solving.” arXiv preprint arXiv:2107.13435 (2021).

[14] Gandhi, Jash, et al. ”Natural Language Processing based Math Word Problem Solver and Synoptic Generator.” 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2022.

[15] Mounika, Y., et al. ”Automatic correction of speech recognized mathematical equations using encoder-decoder attention model.” 2022 IEEE 19th India Council International Conference (INDICON). IEEE, 2022.

[16] Zong, Mingyu, and Bhaskar Krishnamachari. ”Solving math word problems concerning systems of equations with gpt-3.” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 37. No. 13. 2023.

[17] Muddaluru, Roshan Vasu, et al. ”Auto-grading C programming assignments with CodeBERT and Random Forest Regressor.” 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, 2023.

[18] Narmada, Nakka, and Peeta Basa Pati. ”Autograding of programming skills.” 2023 IEEE 8th International Conference for Convergence in Technology (I2CT). IEEE, 2023.

[19] Bikku, Thulasi, et al. ”Exploring the effectiveness of BERT for sentiment analysis on large-scale social media data.” 2023 3rd International Conference on Intelligent Technologies (CONIT). IEEE, 2023.

[20] Sriharsha, C., et al. ”Intelligent learning assistant using BERT and LSTM.” 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT). IEEE, 2021.

[21] Kumar, CS Ayush, et al. ”BERT-Based Sequence Labelling Approach for Dependency Parsing in Tamil.” Proceedings of the Second Workshop on Speech and Language Technologies for Dravidian Languages. 2022.