Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 235 (2024) 2178–2186

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering (ICMLDE 2023)

# Fine-Tuned T5 For Auto-Grading Of Quadratic Equation Problems

Roshni M Balakrishnan[a], Peeta Basa Pati[a], Rimjhim Padam Singh[a,*], Santhanalakshmi S[a], Priyanka Kumar[b]

[a]*Department of Computer Science and Engineering, Amrita School of Computing, Bengaluru, Amrita Vishwa Vidhyapeetham, India*
[b]*Department of Computer Science, University of Texas Saint Antonio, USA*

**Abstract**

Assessments constitute a fundamental and inevitable component of any educational journey. Manual effort required for the evaluation of these assessments is very high. Automation of the evaluation process and grading helps in making the review process more efficient, objective, and scalable, thereby reducing the workload of human reviewers. Automating the grading process for multiple-choice and short-answer assessments is relatively straightforward, but it poses significant challenges when applied to the evaluation of formal languages, particularly in the context of mathematical assessments. In this paper a model that automatically evaluates and grades the Quadratic Equation problems is presented. The study is conducted using a manually curated dataset comprising 1200 solutions to various quadratic equation problems. Embeddings of the quadratic solutions are generated using Google's T5 Model. These embeddings are then used to train different traditional and ensembled machine learning models along with complex Deep learning models like LSTM and Bi LSTM. An in-depth analysis of the fine-tuned T5 model's performance, evaluating its effectiveness in comparison with the pretrained T5 model in automatic grading of quadratic equation problems has been explored. Fine-tuning significantly contributes to the reduction of error by 70% and a noticeable increase in the R2 value to 97%.


Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering

*Keywords:* Auto-grading; T5; Fine tuning; Subject matter experts; Formal language; Mathematics; Quadratic Equations;

## 1. Introduction

The most important side of a teaching-learning process is conduction of assessments. It helps in understanding the students' knowledge in a particular subject. Conduction of assessments are time consuming and tedious for the educators, as they have to set the question paper, key to those questions and last but not the least is the more hectic evaluation part. The conventional methods of conducting assessments are also susceptible to human errors. Automating the process of evaluation could be a solution for reducing the effort of the educators to a greater extend. It is easy to automate the evaluation of multiple-choice questions whereas when it comes to automatic grading of short answers,

* Corresponding author
*E-mail address:* m_roshni@blr.amrita.edu

1877-0509 
Peer-review under responsibility of the scientific committee of the International Conference on Machine Learning and Data Engineering
10.1016/j.procs.2024.04.206

it becomes more challenging. Computer-assisted assessment (CAA) represents an emerging research field dedicated to exploring the potential of computer technologies in assessing students' learning progress. The evaluation of free-text answers within the realm of CAA has remained a persistent challenge, drawing the attention of researchers since the 1960s. Despite this lengthy period of investigation, the complexities associated with CAA of free-text responses remain unresolved [12]. In case of short or long answers the educators have to read the whole content and try to analyse the understanding of the student. The educator neither gives full mark nor award zero if the students' answer is matching or not matching with the content of the key. In this regards, certain keywords are searched for and marks are awarded accordingly. But when it comes to the evaluation of formal language answer scripts like programming, mathematics, chemical equations etc., marks are not based on any keywords. In-spite of there being a proper rubrics for evaluation, the educators award marks based on the approach of the student towards a particular question. Especially in case of mathematics, the final answer alone does not help the student in scoring full marks, instead each steps contribute towards the marks obtained. A student who wrote the complete steps and made a minute error in a step due to which the answer becomes wrong is susceptible to score more than a student who got the final answer correct but skipped many of the steps in between. Marks assigned to the same question can vary based on individual teachers. This variation is often attributed to differences in each assessor's grading policies, even when they adhere to a shared rubric or guidelines [4]. This makes the automation of evaluation process quite a challenging task.

The main contribution of the study is:

- A model that emulates the grading process employed by subject matter experts to assign marks.
- A Detail exploration on how a non natural, formal language text can be processed and how a normal transformer model can be used for creating the embeddings of non natural formal language text for a machine to be trained.
- A dataset manually created for the study which contains about 1200 samples of quadratic equation solutions.

The rest of the paper is organized as follows. Section 2 deals with some of the previous works reported in this domain. The Proposed architecture and experimentation is reported in the section 3. Detail exploration of dataset and the result analysis is done in section 4. Section 5 concludes the paper.

This research pursuit is in accordance with the fourth Sustainable Development Goal (SDG-4) set by the United Nations. This goal underscores the importance of striving for high-quality education.

## 2. Related Work

Pacheco et al. [13] proposed an automatic mathematical expression evaluation tool named MathDIP. The system provides immediate feedback after each step so that the student get to know the errors and correct them by comparing with the sequence of steps in the solution set.

Lan et. al. [8] has tried to automatically grade the students' response for a mathematical question. For each of the solution a feature vector has been created. Similarity check was done between the original solution vector and the feature map created. The similar answers has been clustered together and grades were assigned to them.

Kushman et.al. [7] built a model to automatically learn how to solve algebra word problems. It analyses and then constructs equations based on the analysis. After creating the equations the model finds way to solve them returning a final output. The authors created a standard form of representing an equation from a normal statement.

A model that focuses on automatically evaluating and grading the students' responses for the open ended problems has been built and reported by Erickson et al. [4]. Students have to specify the reason for choosing a particular answer in MCQ assessments. Based on the reasoning statement, grading is done. The model converts the answers to a vector, using Count Vectorizer. These vectors are used to train different Deep learning (DL) and machine learning (ML) models and predict the grades.

A good number of study was done with a combination of word embeddings and DL models like RNN, LSTM and different ML models in automatic short answer grading [1],[19]. Use of CodeBERT and T5 transformer in combination with different ML/DL models for the automatic evaluation and grading of C programs has been reported by Narmada and Pati [11] and Muddaluru et. al. [10]. Mounika et.al. [9], reports about an encoder decoder model built for correcting the errors of speech recognized mathematics equations.

From a detailed survey done, it was observed that no study has been done on the use of any transformer embedding models in combination with ML/DL models in automatic grading of mathematics answer scripts.

Prabhakar et al. [14] reported the effect of fine tuning process of T5 in summarizing Legal Documents. Zhuang et al. [22] explored and reported on the fine tuned T5 for text ranking. From the findings of Narmada and Pati[11], powerful language model T5, is effective in handling programming-related tasks.

The comparatively better performance of T5 as explored and reported in different domains is likely due to T5's ability to understand Natural and formal language. Therefore, the proposed model focuses on use of feature vectors generated by Google's state of the art, T5 — Text-to-Text Transfer Transformer Model [15] pre-trained and fine tuned in training a wide range of regressors for predicting the scores.

## 3. Proposed Methodology

The proposed model extracts the embedding from the quadratic equation solutions using T5 and these embeddings are used to train the model for predicting the marks. To perform the analysis both traditional and ensembled ML regressors and more complex DL algorithms were used. Fig 1. shows the process flow of the entire system. The architecture shows that the whole process can be subdivided into 3 phases - Preprocessing, Feature Extraction and Training - Testing.

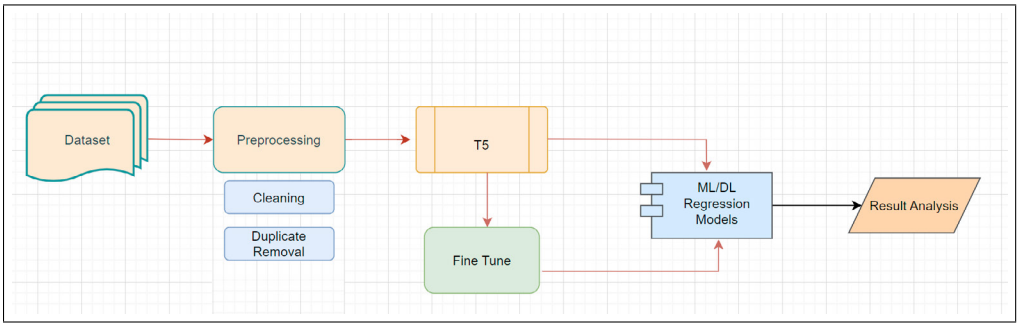


Fig. 1. Process Flow of the proposed system

The model utilized two preprocessing tasks: cleaning and the removal of duplicates. The dataset contained certain solution.

Cleaning phase: The cleaning phase involved the removal of specific responses, such as "I don't know" or "I forgot," from the samples. These types of answers were excluded as they were not relevant to the study's primary objective, which was to investigate the performance of T5 on non-natural language data.

Duplicate Removal: The dataset included a few duplicate entries, which were systematically eliminated through manual intervention. Duplicate entries can distort the quality of the dataset. They provide redundant information that doesn't contribute to the learning process but can lead to bias in models and might lead to the case of overfitting by learning to assign too much importance to these duplicated examples, resulting in poorer generalization to new, unseen data

The T5 model is a Transformer-based architecture. [20] which is a language model that may be applied to a number of NLP applications, including question answering, text summarization, and machine translation. There are two major components: an encoder and a decoder. A hidden state representation of the input text is created by the encoder. The output text is then produced by the decoder using the concealed state representation. The T5 model was initially trained on a vast text and code dataset. This training enables the model to acquire a broad comprehension of language. After then, the model can be fine-tuned for a specific NLP task by training it to provide the desired output for a given input. Fine-tuning is nothing but adapting a pre-trained model on specific datasets, and allowing it to specialize its knowledge for domain-specific or application-specific purposes.

The proposed model initially utilizes the pre-trained version of T5-base for a pilot study. Subsequently, it fine-tunes T5-base using our dataset of quadratic solutions. During the fine-tuning process, the T5 model's encoder-decoder

architecture was employed. In this process, the same dataset was used as both the input and the output for training the model. By utilizing the input data as the target output, the primary aim was to leverage the powerful encoding capabilities of the T5 model to create rich vector representations for the input text data[3] [5]. The model's features were extracted by obtaining the embeddings generated by the encoder. These embeddings served as the key input features for training the model.

In the training and testing phase, the features extracted during second phase were employed to train and evaluate various machine learning and deep learning models. The study aimed to assess the performance of a diverse range of models, for instance traditional ML models like KNN and Decision Tree, ensemble ML models like Random Forest, boosting ensemble models like XGBoost, CatBoost and sequential DL models like BiLSTM (Bidirectional Long Short-Term Memory) and LSTM (Long Short-Term Memory).

## 4. Experimentation

This section provides an in-depth description of the experimental setup, encompassing every aspect from the curation of the dataset to hyperparameter tuning, model training, and testing. Furthermore, it elaborates on the various evaluation metrics employed to assess the performance of the models.

### *4.1. Data Creation*

21 random quadratic equation problems which can be solved in multiple ways were taken. 60 students chosen in random were asked to solve all the 21 questions. 1260 samples thus collected were subjected to different pre processing techniques viz.

- Preprocessing: 1260 samples had many duplicates and vague responses which was manually removed and created 1197 unique solutions.
- Grading by SME: These solutions collected from the students were evaluated and graded by 4 subject matter experts (SME) on a scale of 5. The SME followed a common rubric that is represented in Table 1.

Table 1. Common rubric followed by the Subject Matter Expert (SME) to obtain uniformity in grading the quadratic equation solutions

| **Conditions** | **Range of Marks awarded** |
|---|---|
| Correct answer with proper steps | 5 |
| Final answer wrong with minute mistakes in the last steps | 4 - 4.5 |
| Correct way of solving but partially correct | 2.5 to 4 |
| Use of correct formulas and mistake in the initial steps | 1.5 to 2.5 |
| Basic formula and substituting the values | 0.5 to 1.5 |

- Final Mark Calculation: The marks awarded by the SMEs even after following common rubric had variations and therefore mean mark was calculated from the 4 different marks. This mean mark was considered for the study.
- Train Test Split: The graded 1197 samples were further divided to separate training and testing data. For testing 70 solutions where chosen manually from all the 21 questions with grades ranging from 1 to 5 instead of choosing random test set.

### *4.2. Pilot Study: Variation in results shown by ML and DL models on Pretrained T5*

In order to assess the impact of different ensembled ML Regressors like Random Forest, CatBoost, XGBoost etc and DL models like LSTM and BiDirectional LSTM, the pretrained T5 model t5-base was used to generate the embeddings and these embeddings were trained and tested with each of these models. This approach allowed to evaluate the inherent capabilities and performance of T5 as a standalone language model, providing a benchmark for comparison with other ML and DL models. Upon training different ML and DL models using the embeddings

generated from the pretrained T5 model, the obtained results revealed that the ML models consistently outperformed the DL models across the evaluated metrics. Root Mean Squared Error (RMSE) was used as the metric for evaluating and analyzing the performance of the model. We obtained a RMSE value in the range of 1.2 to 2.1 with different ML algorithms with a least error of 1.24 for Decision Tree and error of 1.69 and 1.58 for BiDirectional LSTM and LSTM, respectively. This is an indication that the size of the data or the number of samples used is comparatively less for training any neural network model. Also, DL models, with their greater complexity and larger number of parameters, can be more prone to overfitting if the dataset is limited. ML models, on the other hand, can generalize better in such cases. The variation in the results by ML and DL models highlight the potential challenge due to the limited sample size. It was also observed that, T5 in its original pretrained form without any additional training or adaptation specifically for the dataset used in the study was able to generate the embeddings. In light of the unsatisfactory results obtained in the pilot study, it was decided to conduct a fine-tuning process on the T5 model to enhance its performance. This fine-tuning step aims to optimize the model's parameters specifically for the grading task, allowing the model to assess the effect of fine-tuning on the data and compare the results obtained with the findings from the pilot study. It is hypothesized that by fine-tuning T5, the model's performance can be improved, and better results can be achieved compared to its original, generic form. Table 2, shows the hyper parameters set for the fine tuning experiment of T5 model.

Table 2. Hyper-parameter setting of T5.

| Hyper-Parameters | Details |
|---|---|
| Model | T5-Base |
| Learning Rate | 0.00001 |
| Batch Size | 8 |
| Number of Epochs | 5 |
| Loss Function | Seq2Seq loss |

After fine-tuning the T5 model, the resulting model was saved, and it was subsequently employed to generate embeddings for the test data, enabling to assess the performance of the fine-tuned T5 model on previously unseen samples. In the learning process, for the various ML regressors, hyperparameter tuning was conducted using RandomizedSearchCV, an optimization technique as demonstrated to be effective in a previous study [2]. RandomizedSearchCV employs a random search approach across a defined hyperparameter space. It selects a random subset of hyperparameters and evaluates them for a specified number of iterations, ultimately identifying the optimal combination of hyperparameters.

### *4.3. Evaluation Metrics*

The fine-tuned T5 model's performance in vectorization, is analysed by Mean Squared Error (MSE) and R-squared ($R^2$) metrics on the marks predicted by different models. MSE quantified the discrepancy between the predicted vector representations and the ground truth, while $R^2$ provided insights into the model's ability to explain the variance in the target vectors, showcasing the effectiveness of the fine-tuning process.

## 5. Result Analysis

This section presents our findings on the research done, addressing two primary questions. Insights into each research question and inferences based on the results obtained from the experiments are provided. The research questions addressed in the study are:

- Q1 What is the comparative impact of fine-tuning on auto-grading performance, as assessed by RMSE values when compared to the results obtained from the pilot study?
- Q2 What is the comparative performance of traditional machine learning and complex deep learning models on embedding generated by fine tuned T5 in automatic grading of quadratic equation?

### 5.1. Dataset Exploration

The dataset used for this study was manually curated. It is an excel file with two columns, one with the solved quadratic problems and the other with their corresponding marks. Fig 2. shows a histogram to explain the distribution of various grades across the training dataset.

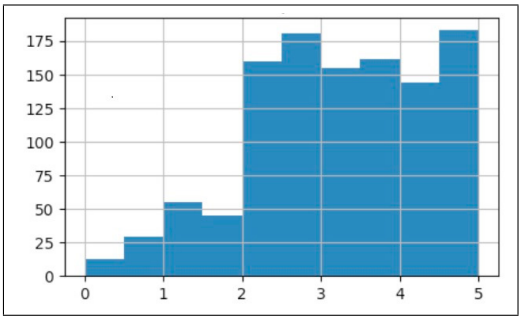


Fig. 2. Distribution of different grades across the dataset

The dataset comprises a sample of n = 1127 quadratic equation graded solutions, where the sample mean of the grades ($\bar{x}$) is 3.14 with a standard deviation of 1.17. The grades in the dataset ranges from a minimum (min) of 0.00 to a maximum (max) of 5.00. Additionally, the dataset exhibited quartiles, with the 25th percentile (Q1) at 2.37, the median (Q2 or 50th percentile) at 3.25, and the 75th percentile (Q3) at 4.00. These sample parameters provide a descriptive summary of the dataset, capturing key characteristics.

*Q1: Impact of Fine-Tuning in Auto-Grading* . To investigate the impact of fine-tuning on the model's performance, the results before and after the fine-tuning process was compared. Upon fine-tuning the model, it was observed that there was a significant improvement in the overall performance. The RMSE value decreased by an average of 77%, indicating that the fine-tuned model achieved higher accuracy in predicting the target variable. Furthermore, the average $R^2$ value increased from 0.13 to 0.89, demonstrating that the fine-tuned model can now better explain the variance in the data. Table 3 shows all the RMSE values obtained by different regressors and sequential recurrent neural network models before and after fine tuning. Fig 3. represents the visualization of the comparison of RMSE values obtained during the pilot study and actual study.

Table 3. RMSE values obtained by different ML and DL models before and after fine tuning.

| Models | T5 Base | T5 Fine-Tuned | Decrease in error |
|---|---|---|---|
| KNN(N=3) [11] | 1.69 | 0.80 | 0.89 |
| Random Forest [10] | 1.74 | 0.49 | 1.25 |
| Decision Tree [11] | 1.24 | 0.22 | 1.02 |
| XGBoost [11] | 2.01 | 0.21 | 1.80 |
| CatBoost [10] | 1.87 | 0.22 | 1.65 |
| Bi Directional LSTM [17] | 1.69 | 1.56 | 0.13 |
| LSTM [6] | 1.58 | 1.55 | 0.03 |

Based on the results obtained, a noticeable decrease in the error before and after fine-tuning is observed, particularly in the CatBoost and XGBoost Regressors. The fine-tuning process has evidently led to significant improvements in the performance of these models, reducing the prediction errors substantially.

For all the models, the RMSE metric demonstrated a substantial reduction in the errors post-fine-tuning. The decrease in error signifies that the fine-tuned models are making more accurate predictions, better approximating the true target values compared to their pre-fine-tuning counterparts.It is worth noting that the observed improvements are consistent with the expectations of fine-tuning, as it is designed to enhance the model's generalization and adaptation to the task of auto grading. The outcomes of these experiments underscore the importance of fine-tuning as a valuable technique in model optimization.

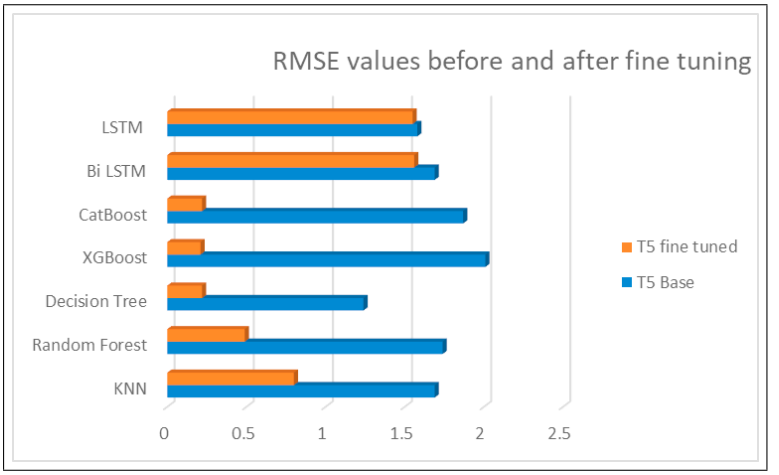


Fig. 3. The RMSE values obtained by different ensembled ML and Sequential Recurrent Neural Networks before and after fine tuning of T5

*Q2: Comparison of results obtained by traditional ML and more complex DL models.* As observed in the pilot study, the results indicate that the ML models outperforms DL models in terms of R-squared ($R^2$) values, with the ML models achieving $R^2$ values close to 1. Table 4 shows different $R^2$ values obtained for training and testing data with fine tuned T5. Fig 4. provides a visualization of this comparison.

Table 4. Training and Testing $R^2$ values of different ML and sequential Recurrent Neural Networks models

| **Models** | **Training $R^2$** | **Testing $R^2$** |
|---|---|---|
| KNN [11] | 0.70 | 0.70 |
| Random Forest [10] | 0.91 | 0.89 |
| Decision Tree [11] | 0.98 | 0.97 |
| XGBoost [11] | 0.98 | 0.98 |
| CatBoost [10] | 0.98 | 0.97 |
| Bi LSTM [17] | 0.15 | 0.13 |
| LSTM [6] | 0.10 | 0.12 |

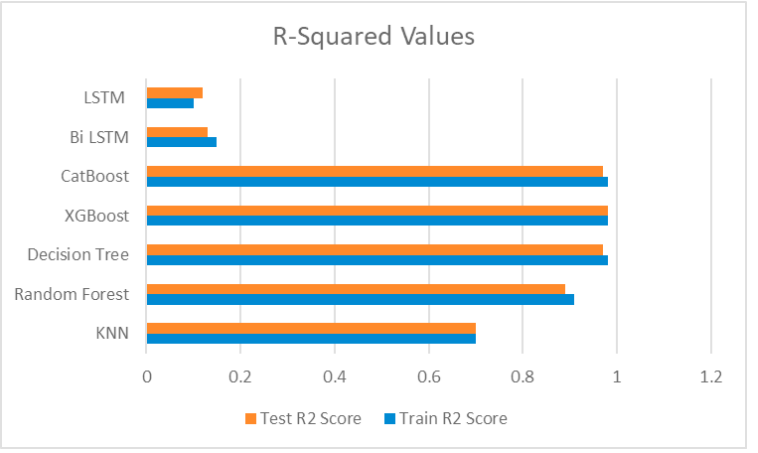


Fig. 4. $R^2$ values obtained with ML and different Sequential Recurrent Neural Networks models

This finding suggests that the ML models fits better to the data and exhibited a high level of predictive accuracy, explaining a large portion of the variance in the embeddings generated. They successfully captured the underlying

patterns and relationships present in the data. ML models often excel in scenarios with limited data, so as seen in Pilot study its obvious that the dataset size is an unavoidable factor which gave this result. On the other hand, the sequence to sequence models tend to require more data and parameter tuning to achieve optimal performance. The model was trying to overfit the data so we implemented early stopping during model training to halt the training process when the model starts overfitting on the validation data. The size of the corpus can be further increased by different augmentation techniques as reported by Venugopalan and Gupta [21] in order to analyse and prove the reason for traditional ML models outperforming the more complex DL algorithms.

As there is no prior work on quadratic equations, the findings reveal that the model is comparable with past equivalent auto grading research done in different domains. Table 5 shows the comparison of best results reported with different performance metrics with automatic grading of programming assignments and short answers. Even though the performance metrics used are different, the results shows that, the automatic evaluation and grading of formal language answer scripts especially mathematics are possible with different learning based algorithms.

Table 5. Comparison of the results obtained in the proposed study with similar previous works

| Model and Domain | Best Results reported |
|---|---|
| Muddaluru et al. [10] in Auto-grading C programming Assignments | R2 = 0.27 |
| Narmada et al. [11] in Autograding of Programming Assignments | RMSE = 1.16 |
| Sanuvala et al. [16] in Automatic evaluation of students' examination paper and [18] | F1 Score of 85.7 |
| Proposed Study on quadratic equation | R2 of 0.98 |

## 6. Conclusion

In this study a new model for automatic grading of quadratic equation was developed using fine tuned T5 for creating the embedding and different traditional and ensemble machine learning models and sequence to sequence deep learning algorithms for training and testing. The model was able to mimic the marks awarded by the subject matter experts. A comparison on the results obtained by using pretrained and fine tuned T5 was explored and reported. From the study conducted it was observed that, fine tuning of the T5 model was able to generate a better embedding thereby enhancing the results by an average of 70%. Similarly the Boosting ensemble models showed a better convergence with r2 value of 0.98 and 0.97 followed closely by Random Forest with r2 value 0.89. The results reveal that even though there was no prior works reported in the domain of automatic grading of mathematics, it is comparable with other formal language scripts which includes programming assignments. In the training and testing phase, it was observed that ML models outperformed the DL models. Increasing the size of the corpus by further parameter tuning and data augmentation might help in producing a better result by the complex deep learning models which becomes the future scope of this study.

## Acknowledgements

The research volunteering team of Amrita Vishwa Vidhyapeetham, Bangalore, provided partial support for this work. We would like to thank the team who helped in generating the corpus used in the study and also the subject matter experts who helped in grading each solution.

## References

[1] Altindiş, Z.T., 2022. Perspective chapter: New approaches to the assessment of domain-specific creativity, in: Brito, S.M., Thomaz, J.P.C.F. (Eds.), Creativity. IntechOpen, Rijeka.

[2] Andonie, R., Florea, A.C., 2020. Weighted random search for cnn hyperparameter optimization. International Journal of Computers Communications& Control .

[3] Einieh, Y., Almansour, A., Jamal, A., 2022. Fine tuning an arat5 transformer for arabic abstractive summarization, in: 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN).

[4] Erickson, J.A., Botelho, A.F., McAteer, S., Varatharaj, A., Heffernan, N.T., 2020. The automated grading of student open responses in mathematics, in: Proceedings of the Tenth International Conference on Learning Analytics & Knowledge, Association for Computing Machinery, New York, NY, USA.
[5] Fukumoto, D., Kashiwa, Y., Hirao, T., Fujiwara, K., Iida, H., 2023. An empirical investigation on the performance of domain adaptation for t5 code completion, in: 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER).
[6] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation .
[7] Kushman, N., Artzi, Y., Zettlemoyer, L., Barzilay, R., 2014. Learning to automatically solve algebra word problems, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Baltimore, Maryland.
[8] Lan, A.S., Vats, D., Waters, A.E., Baraniuk, R.G., 2015. Mathematical language processing: Automatic grading and feedback for open response mathematical questions. arXiv:1501.04346.
[9] Mounika, Y., Tarakaram, Y., Lakshmi Prasanna, Y., Gupta, D., Basa Pati, P., 2022. Automatic correction of speech recognized mathematical equations using encoder-decoder attention model, in: 2022 IEEE 19th India Council International Conference (INDICON).
[10] Muddaluru, R.V., Thoguluva, S.R., Prabha, S., Balakrishnan, R.M., Pati, P.B., 2023. Auto-grading c programming assignments with codebert and random forest regressor, in: 2023 14th International Conference on Computing, Communication and Networking Technologies (ICCCNT).
[11] Narmada, N., Pati, P.B., 2023. Autograding of programming skills, in: 2023 IEEE 8th International Conference for Convergence in Technology (I2CT).
[12] Noorbehbahani, F., Kardan, A., 2011. The automatic assessment of free text answers using a modified bleu algorithm. Computers Education .
[13] Pacheco-Venegas, N.D., López, G., Andrade-Aréchiga, M., . Conceptualization, development and implementation of a web-based system for automatic evaluation of mathematical expressions. Computers Education .
[14] Prabhakar, P., Gupta, D., Pati, P.B., 2022. Abstractive summarization of indian legal judgments, in: 2022 OITS International Conference on Information Technology (OCIT).
[15] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J., 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv:1910.10683.
[16] Sanuvala, G., Fatima, S.S., . A study of automated evaluation of student's examination paper using machine learning techniques, in: 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS).
[17] Schuster, M., Paliwal, K., 1997. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing .
[18] Sung, C., Dhamecha, T.I., Mukhi, N., 2019. Improving short answer grading using transformer-based pre-training, in: Artificial Intelligence in Education: 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part I 20, Springer. pp. 469–481.
[19] Szarvas, G., Zesch, T., Gurevych, I., 2011. Combining heterogeneous knowledge resources for improved distributional semantic models, in: Gelbukh, A.F. (Ed.), Computational Linguistics and Intelligent Text Processing, Springer Berlin Heidelberg, Berlin, Heidelberg.
[20] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Advances in Neural Information Processing Systems .
[21] Venugopalan, M., Gupta, D., 2015. Sentiment classification for hindi tweets in a constrained environment augmented using tweet specific features, in: Prasath, R., Vuppala, A.K., Kathirvalavakumar, T. (Eds.), Mining Intelligence and Knowledge Exploration, Springer International Publishing, Cham.
[22] Zhuang, H., Qin, Z., Jagerman, R., Hui, K., Ma, J., Lu, J., Ni, J., Wang, X., Bendersky, M., 2022. Rankt5: Fine-tuning t5 for text ranking with ranking losses. arXiv:2210.10634.