Math Word Problem Solver using NLP

Pallavi N[1], Darshana A. Naik[2],Pallavi T P [3] ,Maddali Kaushik[4], Aaryan Manakame[5], Ashwin Reddy[6], Goutham Suresh[7]

Pallavi.n@msrit.edu[1], darshananaik@msrit.edu[2] ,pallavitp@msrit.edu[3]

Department of CSE, M S Ramaiah Institute of Technology, Bangalore[1,2,3,4,5,6,7]

***Abstract* - In recent times, there has been notable focus on math word problem solvers, primarily because of their capacity to aid learners in tackling intricate mathematical problems. Nonetheless, a requirement exists for math word problem solvers capable of addressing queries in various languages through the application of natural language processing (NLP) methods. This study introduces a math word problem solver designed to handle inquiries in diverse languages using NLP. The approach outlined in this research employs a machine learning strategy to convert the problem into a target language, followed by its resolution. Through the integration of both rule-based and deep learning techniques, the system strives to attain a commendable level of accuracy in both translation and problem-solving endeavors.**

Keywords: Solvers for mathematical word problems, utilization of natural language processing, application of machine learning techniques, incorporation of deep learning methodologies, machine translation procedures, utilization of rule-based translation, language recognition procedures, handling of multiple languages.

I. Introduction

Mathematical word problems serve as a fundamental conduit to a spectrum of quantitative reasoning abilities necessary for an AI agent to exhibit intelligent behavior. This intelligence encompasses tasks like comprehending financial news, calculating alterations in everyday transactions, and scrutinizing organizational productivity. However, solving such problems persists as a challenge due to the intricate nature of mapping real-world objectives, entities, and quantities into mathematical formalizations.

This project's objective is to formulate a system capable of resolving mathematical word problems articulated in various languages using natural language processing techniques. The system's primary target audience comprises self-directed learners seeking to grasp mathematics autonomously. It seeks to empower them with a tool that can automatically decipher math problems presented in a natural language format. The system's mechanism involves processing a math word problem through NLP to dissect it into a mathematical equation, subsequently amenable to solution through mathematical algorithms.

In our current trajectory toward a digital society, virtually all challenges are addressed digitally, spanning from basic information storage to intricate computations. Many arithmetic computations are conducted digitally to mitigate human errors and inaccuracies. However, the majority of digital math calculators necessitate the problem's representation in an arithmetic format. In reality, most arithmetic quandaries manifest in the guise of worded problem statements, demanding manual conversion into arithmetic expressions by humans. To obviate this need for manual conversion, our objective is to construct software proficient in recognizing and directly resolving word problems expressed in natural language. Additionally, we endeavor to rectify the shortcomings observed in existing word problem solver software.

II. Literature Survey

The primary aim of this project is to facilitate users in resolving mathematical problems through the utilization of natural language. It leverages transformer-based techniques from Natural Language Processing (NLP) to comprehend user queries and generate appropriate responses. The integration of transformers has brought about a revolutionary shift in the realm of NLP, showcasing exceptional performance across diverse tasks such as machine translation, text generation, and question answering.

This review of the literature delves into recent studies that explore the utilization of transformers and analogous models like LSTM and basic RNNs for the resolution of mathematical problems. It examines their potential application in the development of solvers for math word problems. Notably, the research focuses on architecture and training strategies that have been investigated globally, outlining their relevance. Ultimately, the study presents our own implementation of MathBot, demonstrating its efficacy in tackling a wide array of mathematical challenges.

In contrast to preceding statistical learning methods, the authors adopt a direct translation approach wherein math word problems are transformed into equation templates via a recurrent neural network (RNN) model, bypassing the need for intricate feature engineering. Furthermore, they propose a hybrid model that amalgamates the RNN model with a similarity-based retrieval model to enhance performance. Comprehensive experiments conducted on a substantial dataset establish the superiority of both the RNN model and the hybrid model over prevailing statistical learning techniques for math word problem solving [1].

The authors introduce the inaugural open-source toolkit for solving Math Word Problems (MWPs) known as MWPToolkit, providing a unified, comprehensive, and extensible framework tailored for research endeavors. The toolkit incorporates 17 deep learning-based MWP solvers and 6 MWP datasets, encompassing diverse models spanning Seq2seq, Seq2Tree, Graph2Tree, and Pre-trained Language Models. These solvers, representing advanced MWP resolution approaches, are matched with widely recognized benchmark datasets. The toolkit's modular and reusable components expedite researchers in their pursuit of model development [2].

In this particular investigation, a sequence-to-sequence model is harnessed to unravel math word problems. By taking problem descriptions as input and generating equations as output, this approach demonstrates remarkable potential. Recognizing two shortcomings in the model—generation of spurious numbers and misplacement of numbers—the researchers propose an innovative augmentation named the copy and alignment mechanism, denoted as CASS. They employ reinforcement learning to train the model, a strategy designed to enhance solution accuracy while overcoming the "train-test discrepancy" linked to maximum likelihood estimation. The findings indicate that the copy and alignment mechanism effectively ameliorates the identified issues, with reinforcement learning exhibiting superior performance in comparison to maximum likelihood estimation. Moreover, the model's output is integrated into a feature-based model, resulting in significant enhancements over state-of-the-art results [3].

The application of Sequence-to-sequence (SEQ2SEQ) models to automatic math word problem solving has proven successful. Yet, a limitation persists—multiple equations can accurately resolve a single math word problem, leading to non-deterministic transduction that adversely affects maximum likelihood estimation. By emphasizing the uniqueness of the expression tree, the authors introduce an equation normalization technique aimed at rectifying the problem of duplicated equations. Additionally, they scrutinize the performance of three popular SEQ2SEQ models in this context and propose an ensemble model to leverage their individual strengths. Empirical evaluations on the Math 23K dataset validate the effectiveness of the ensemble model coupled with equation normalization, surpassing previous state-of-the-art methods [4].

Efficiently solving math word problems stands as a pivotal task for assessing machine intelligence in the broader domain of general AI. This endeavor is inherently complex, necessitating adeptness in both natural language comprehension and mathematical expression inference. While existing solutions tend to rely on sequence-to-sequence models for expression generation, these models often fall short in understanding problems akin to human cognition, leading to erroneous outcomes. This paper introduces a groundbreaking solution named Hierarchical Math Solver (HMS), which capitalizes on deep comprehension and utilization of problems. HMS introduces a hierarchical word-clause-problem encoder, mirroring human reading habits in problem comprehension. It adopts a dependency-based module for clause understanding and employs a novel tree-based decoder to generate mathematical expressions for answers. This decoder leverages hierarchical attention mechanisms and a pointer-generator network to enhance semantic context and guide the inference process. Extensive experimental results underscore the superiority of HMS in generating not only more accurate answers but also more logical inferences [5].

The present study unveils a substantial dataset of math word problems and an interpretable neural solver that learns to map problems to operation programs. Overcoming annotation challenges, the dataset establishes precise operational annotations across diverse problem types. Introducing a new representation language to model operation programs, the authors create the MathQA dataset, enhancing the AQuA dataset with fully-specified operational programs. A neural sequence-to-program model is enhanced through automatic problem categorization. Experimental outcomes validate improvements over competitive baselines in both the MathQA and AQuA datasets. Nevertheless, human performance still significantly outpaces the results, implying new challenges for future research [6].

Efficiently solving diverse textual math word problems (MWPs) poses a substantial challenge, particularly since existing works tend to focus on single-unknown linear MWPs. This study introduces the Universal Expression Tree (UET) method, which endeavors to represent equations for various MWPs uniformly. Building upon this, the semantically-aligned universal tree-structured solver (SAU-Solver) is introduced, capitalizing on an encoder-decoder framework. SAU-Solver generates a universal expression tree by making symbol decisions based on semantic meanings, mimicking human problem-solving approaches. A novel subtree-level semantically aligned regularization is also incorporated to enhance the model's rationality and adherence to semantic constraints. The efficacy of this approach is demonstrated on a challenging Hybrid Math Word Problems dataset (HMWP), encompassing various MWP types [7].

The Academia Sinica Diverse MWP Dataset (ASDiv) is introduced as a comprehensive English math word problem (MWP) corpus, aimed at evaluating diverse MWP solvers. Unlike preceding corpora, ASDiv addresses a broader spectrum of language patterns and problem types. This dataset, comprising 2,305 MWPs, covers a comprehensive array of problem types taught at elementary school levels. Each MWP is annotated with its problem type and grade level to denote difficulty. The metric proposed to gauge lexicon usage diversity demonstrates the heightened diversity of ASDiv compared to existing corpora. Experiments validate the authentic evaluation capabilities of MWP solvers, offering a more faithful representation of their capabilities [8].

III. System Architecture

he objective of the System Design is to offer a comprehensive outline of the system's architecture. Its purpose is to foster effective communication and collaboration among project team members, stakeholders, and other relevant parties. This design blueprint will function as a guide, steering the course for the system's implementation and testing phases. It will also play a pivotal role in shaping the development of software requirements, design specifications, and additional system elements.

A. Architecture Design

Similar to preceding seq2seq models, the original Transformer architecture followed an encoder-decoder structure. The encoder is composed of
encoding layers that progressively process the input, with each layer building upon the previous one. Conversely, the decoder comprises decoding layers that mirror this process with the encoder's output. The central role of each encoder layer is to create encodings that capture the relationships between different elements within the input. These encodings are then passed on as inputs to the subsequent encoder layer. In contrast, each decoder layer employs these encodings to generate an output sequence, utilizing the contextual information they encapsulate.

To achieve optimal outcomes:

1. Each encoder and decoder layer integrates an attention mechanism.
2. The attention mechanism assigns weights to the relevance of various input components, drawing from them to craft the output.
3. In addition to this, every decoder layer features an extra attention mechanism. This mechanism draws from the outputs of preceding decoders before obtaining information from the encodings.

4. Both the encoder and decoder layers are equipped with an additional feed-forward neural network for further processing of the outputs. These layers incorporate residual connections and include steps for layer normalization.

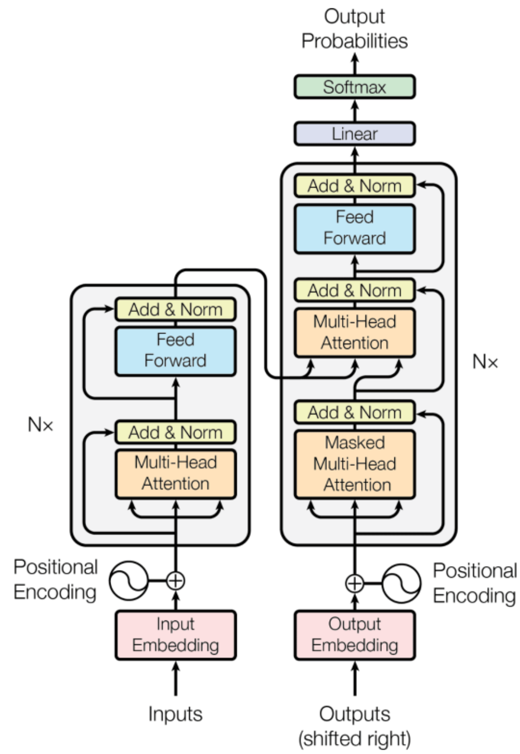
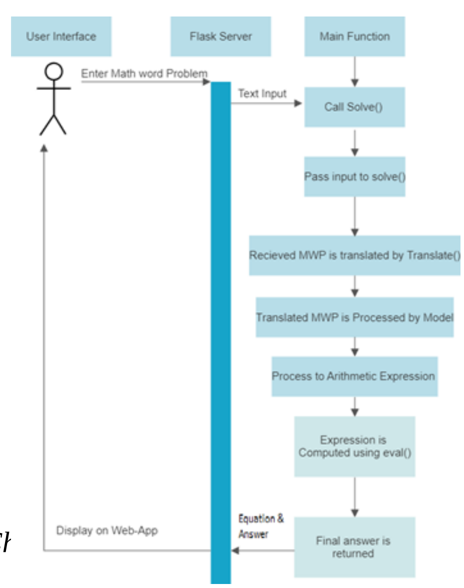


Fig 1. Architectural design

B. *Use Case Diagram*

The actors involved are the application users and the transformer based MWP solver software. Application users initially choose a math word problem that they wish to compute and enter it in its text format into the user interface screen. Having entered the desired arithmetic word problem into the text box and clicking "solve" the software pushes the text input into a function called "Solve()". Correspondingly the input text is further moved to "solve()" where the translate function translates the MWP across languages as per user's input. The translated Math Word Problem is then processed by the model. For the translated Math Word Problem, a corresponding arithmetic expression is generated which is to be computed to generate a final result. The computation of this arithmetic expression is carried out by the "eval()" function. The final computed answer to the input MWP is then returned and displayed to the user on the web application.



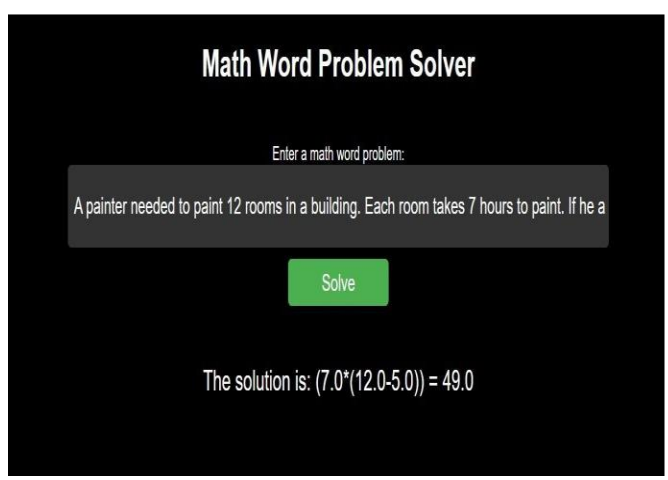


Fig 2. Use Case Diagram for the Math Word Problem Solver

IV. Implementation

A. Graphical User Interface

Figure 3, shows the User Interface Mockup designs for the client-side web application. The user interface for the Math Word Problem solver should be intuitive and easy to use, to enable users to quickly and easily get answers for their questions. The solve function should allow users to get the necessary correct output for the given inputted question in the text field.

Once the output has been calculated it is displayed on the screen in another text field along with the corresponding equation which produces the answer. Overall, the user interface should be designed with the user's needs and expectations in mind, to provide a seamless and enjoyable experience for all users of the system.

B. Data Flow Diagram

The dataflow for this math problem solving system begins with the user inputting a math problem. The 'Solve' function then sends the input to the Google Translate API, for translation to English when needed. The resulting text is sent to an ML model that computes an arithmetic expression using a combination of neural machine translation and rule-based translation techniques. The computed result is then displayed back to the user through a web interface or another type of user interface.

Fig 3. Web User Interface Design for the MWP

The dataflow includes multiple steps, such as input, translation, computation, and output, and utilizes NLP techniques and ML models to provide an accurate solution to the math problem. The final result is presented back to the user, completing the dataflow for the system.

C. User Interface

Our Application will enable users to interact with the software through a text based user interface in which they will be able to type in the MWP (Math Word Problem) in order to obtain the corresponding result. The user interface will provide the user a space to type in the desired text and a button to send the entered text for evaluation of the arithmetic word problem. The corresponding result will be displayed to the user on screen as and when it has been computed. The User Interface will be simple and easy to understand and use, even for users who do not have much knowledge of the underlying model.

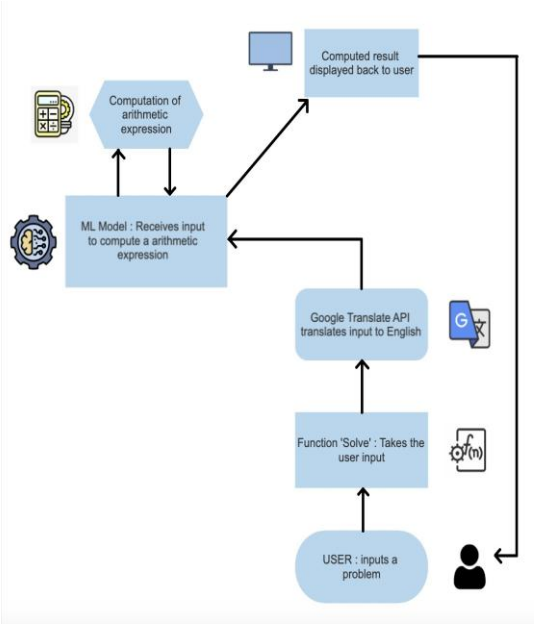


Fig 4. Process Flow for the MWP

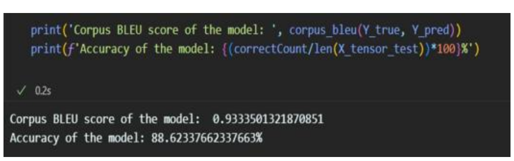
D. Hardware and Software Interfaces

Hardware required to process the data and train the model is not specific. A CPU having at least 4 cores is required for optimal performance. We utilize text-based input for the math word problem, which must be typed using the given keyboard. NVIDIA GPUs that are CUDA-enabled will help us speed up the time required for training by parallel processing high dimensional data .
Python will be the main language used for building the model which will be typed out on an IDE such as VScode or Jupyter.

PyTorch is a machine learning framework based on the Torch library, used for building natural language processing applications. Traditional neural networks suffer from short-term memory. LSTMs efficiently improve performance by memorizing the relevant information that is important and finding the pattern. The UI will mainly be designed using HTML, CSS, Javascript, React and Node.js . This will be used to input the data for the model to process and print the output.

V. Results

The implementation of the Math word problem solver using NLP is expected to have significant positive results. It's built using a transformer model that is trained on a huge number of datapoints and gives high accuracy for math word problems that does not require multiple steps to solve. We received an accuracy of 88% after fine tuning the model with different parameters and adding more data points to the dataset. Due to the training process' high system demands and the amount of the dataset, an NVIDIA GPU is required.


```
print('Corpus BLEU score of the model: ', corpus_bleu(Y_true, Y_pred))
print(f'Accuracy of the model: {(correctCount/len(X_tensor_test))*100}%')

✓ 0.2s

Corpus BLEU score of the model:  0.9333501321870851
Accuracy of the model: 88.62337662337663%
```

Fig 5. Accuracy of model after training

This application may help a lot of people who are attempting to solve arithmetic word problems to do so with great accuracy and dependability, and it also saves time because you don't need another person to check your answers to see whether they're right or wrong.

Some of the other advantages include understanding natural language which enables the model to process word problems and get arithmetic equations to solve it and provides better accessibility as the user can use the application from anywhere as long as they have a PC without the need for an internet connection. Also, the model supports multilingual inputs which enables users who are not familiar with English to also use the MWP solver.

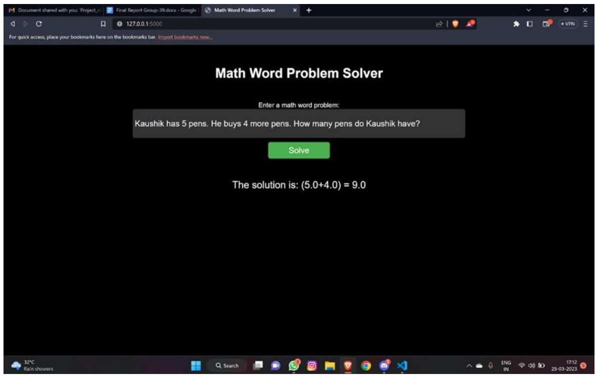


Fig 6. Front-End result for MWP in English

The math word problem solver is accessed by users via the front-end web application, which is linked to the backend Solve() function, which receives the math word problem in any language, translates and processes the question. The response is returned to the front end once it has been processed and solved by the transformer model. The approximate time necessary to solve any question and obtain the answer is 5 to 10 seconds.

The front-end is built utilizing Python's flask framework, as well as HTML and CSS. The model has been tested on several languages, including Kannada, Hindi, and French, and the accuracy for MWPs in other languages is nearly the same, with the exception of circumstances when there is a grammatical or spelling issue when being translated.

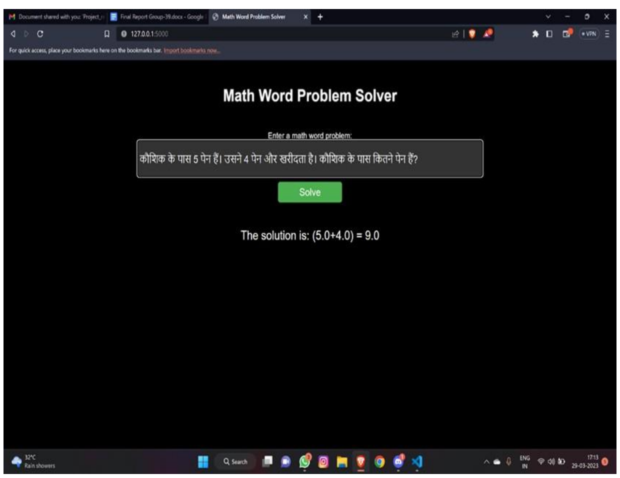


Fig 7. Front-End result for MWP in hindi

As a result, a math word problem solver may be a useful tool for anybody who wants assistance with arithmetic word problems. The solver can save time and minimize stress by giving precise and efficient answers to arithmetic problems. Additionally, using a math word problem solver can increase confidence in math abilities and ultimately lead to improved math skills. Hence the model performs well and gives accurate and desired results.

VI. Conclusion and Future Scope

Math Word Problem Solving using NLP and sequence-to-sequence (seq2seq) models is a promising research area that aims to automate the process of solving math problems presented in natural language format. Seq2seq models have been used to translate one sequence of text into another, and they have been adapted to math word problem solving by treating the problem as a translation task, where the problem statement is translated into an equation or a sequence of equations that can be solved. One of the major challenges in this area is to ensure that the model can understand the semantics of the problem and correctly represent the information in a way that can be used to generate the correct solutions.

Overall, while there is still much work to be done in this field, the combination of NLP and seq2seq models shows promise in automating math problem solving and has the potential to improve education and accessibility for students who struggle with math.

Future Scope-

1. Ability to solve math word problems having a higher degree of difficulty like problems with multiple operations.
2. Improve the model in terms of being able to compute problems with higher correlation and multiple entities.
3. Ease of access by converting the current web-based application to a mobile application.
4. Improve accuracy of currently trained models to give more accurate and consistent results.
5. Capability to integrate with existing and upcoming technologies such AR & VR to create a more immersive and interactive learning experience.

Acknowledgment

We would like to express our heartfelt gratitude to our mentor for guiding us through the process of making this application a reality.

References

[1] Wang, Y., Liu, X., & Shi, S. (2017, September). Deep neural solver for math word problems. In Proceedings of the 2017 conference on empirical methods in natural language processing (pp. 845-854).

[2] Koncel-Kedziorski, R., Roy, S., Amini, A., Kushman, N., & Hajishirzi, H. (2016, June). MAWPS: A math word problem repository. In Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies (pp. 1152-1157).

[3] Ughade, S., & Kumbhar, S. (2019, April). Survey on mathematical word problem solving using natural language processing. In 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT) (pp. 1-5). IEEE.

[4] Huang, D., Yao, J. G., Lin, C. Y., Zhou, Q., & Yin, J. (2018, July). Using intermediate representations to solve math word problems. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 419-428).

[5] Wang, L., Wang, Y., Cai, D., Zhang, D., & Liu, X. (2018). Translating a math word problem to an expression tree. arXiv preprint arXiv:1811.05632.

[6] Zhang, D., Wang, L., Zhang, L., Dai, B. T., & Shen, H. T. (2019). The gap of semantic parsing: A survey on automatic math word problem solvers. IEEE transactions on pattern analysis and machine intelligence, 42(9), 2287-2305.

[7] Brissiaud, R., & Sander, E. (2010). Arithmetic word problem solving: A situation strategy first framework. Developmental Science, 13(1), 92-107.

[8] Sundaram, S. S., & Khemani, D. (2015, December). Natural language processing for solving simple word problems. In Proceedings of the 12th international conference on natural language processing (pp. 394-402).

[9] Huang, D., Liu, J., Lin, C. Y., & Yin, J. (2018, August). Neural math word problem solver with reinforcement learning. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 213-223).

[10] Zong, M., & Krishnamachari, B. (2022). Solving math word problems concerning systems of equations with gpt-3. In Proceedings of the Thirteenth AAAI Symposium on Educational Advances in Artificial Intelligence.