Math Problem Solving: Enhancing Large Language Models with Semantically Rich Symbolic Variables

Ali Emre Narin
Kabatas Erkek High School
Istanbul, Turkey
aliemre2024@gmail.com

Abstract

The advent of Large Language Models (LLMs) based on the Transformer architecture has led to remarkable advancements in various domains, including reasoning tasks. However, accurately assessing the performance of Large Language Models, particularly in the reasoning domain, remains a challenge. In this paper, we propose the Semantically Rich Variable Substitution Method (SemRiVas) as an enhancement to existing symbolic methodologies for evaluating LLMs on Mathematical Word Problems (MWPs). Unlike previous approaches that utilize generic symbols for variable substitution, SemRiVas employs descriptive variable names, aiming to improve the problem-solving abilities of LLMs. Our method aims to eliminate the need for LLMs to possess programming proficiency and perform arithmetic operations, to be universally applicable. Our experimental results demonstrate the superior accuracy of SemRiVas compared to prior symbolic methods, particularly in resolving longer and more complex MWP questions. However, LLMs' performance with SemRiVas and symbolic methods that utilize one-character variables still falls short compared to notable techniques like CoT and PaL.

Keywords: Math Word Problems, Large Language Models

1. Introduction

The Transformer architecture (Vaswani et al., 2023) has facilitated the development of Large Language Models (LLMs) capable of achieving exceptional performance. Models like PaLM 540B (Chowdhery et al., 2023) and GPT-4 (Achiam et al., 2023), with billions of parameters, have undergone training on vast amounts of textual data using the Transformer architecture. These models exhibit outstanding capabilities across various domains, including reasoning, coding, common sense, translation, and planning, often reaching or surpassing human-level performance (Achiam et al., 2023). The remarkable performance of these models has sparked significant interest among researchers, leading to the creation of numerous LLMs such as Llama (Touvron et al., 2023) and Phi (Li et al., 2023).

The proliferation of models has motivated researchers to devise various methodologies for assessing model performance and to curate datasets for evaluating these methodologies. Typically, researchers evaluate the responses of LLMs against questions in datasets, assessing the accuracy of these responses compared to ground truth answers (Mialon et al., 2023; Cobbe et al., 2021).

One domain where such evaluations are conducted is reasoning. In this domain, models are tasked with detecting and executing various operations based on provided text. Achieving satisfactory results in the reasoning domain has proven challenging for LLMs, with many models exhibiting notably low accuracy rates in this area (Cobbe et al., 2021).

To measure accuracy in the reasoning domain, many researchers use Mathematical Word Problems (MWP) (Li et al., 2023; Chowdhery et al., 2023). These MWPs present mathematical challenges using everyday language and numerical data. Successfully solving the problems demands models to possess a strong proficiency in Natural Language Understanding (NLU), as they must identify both the problem scenario and the route to the solution beforehand. This dual requirement of comprehending the context and deducing the problem's intent closely aligns with human reasoning capabilities, making MWPs a suitable evaluation method in the reasoning domain.

Various datasets and methods are employed by researchers to gauge skills using MWPs. Examples of widely used datasets in the literature include GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and SVAMP (Patel et al., 2021).

In addition to datasets, various methods have been developed to assist Large Language Models (LLMs) in accurately answering questions, particularly for Mathematical Word Problems (MWPs). Predominant among these are Chain-of-Thought Prompting (CoT) (Wei et al., 2023) and Program Aided Language Models (PaL) (Gao et al., 2023). CoT employs an intuitive methodology, guiding AI models through progressive problem-solving steps, while PaL necessitates models to generate program code for solution computation.

These two methods have been observed to significantly increase the accuracy rates of responses to questions in datasets used for measuring reasoning in LLMs. However, both approaches have spe-

MathNLP: The 2nd Workshop on Mathematical Natural Language Processing @ LREC-COLING 2024, pages 19–24

Question
Dan plants 3 rose bushes. Each rose bush has 25 roses. Each rose has 8 thorns. How many thorns are there total?
Answer
Dan plants 3 rose bushes. Each rose bush has 25 roses. Each rose has 8 thorns. So **3 x 25 x 8 = 300**. The answer is 300.

Figure 1: Examples of Errors Made by Large Language Models in Arithmetic (Wei et al., 2023)

cific problems. In the CoT approach, LLMs are expected to perform arithmetic operations. However, LLMs are not models highly-competent in performing arithmetic operations (Nogueira et al., 2021; Lu et al., 2023; Frieder et al., 2023; Stolfo et al., 2023; Meadows et al., 2023). The primary function of LLMs in solving MWPs should be to determine which operations to perform among numbers based on the given scenario. As shown in Figure 1, models correctly define operations, but inaccuracies in performing operations have been observed, leading to decreased accuracy rates. Therefore, measurements made with these approaches only partially reflect reality. The PaL method is not effected by errors that can occur due to arithmetic operations, since a third party code interpreter runs the calculations, however it is limited for models that are proficient in using programming languages. Therefore, PaL is limited in terms of the LLMs it can be applied to.

New methods have emerged to address these challenges, leveraging symbolic variables to delegate arithmetic computations to an external calculator (He-Yueya et al., 2023). Additionally, these methodologies does not require any programming proficiency. One particular method, replaced variables in a question with “w,x,y,z” variables, and then asked LLM to solve the question using self-prompting strategies (Gaur and Saunshi, 2023). We refer to this kinds of approaches in our paper as "One-Character Substitution Method", as they exchange numbers with one-character variables.

This strategy effectively tackles issues related to arithmetic errors and programming expertise limitations. In our study, we investigate the potential refinement of this technique by replacing these generic placeholders with semantically rich counterparts. Instead of employing generic symbols, we advocate for descriptive variable names such as "number-of-books-James-has" as exemplified in Figure 2. We hypothesize that this modification will aid LLMs in problem-solving tasks, as managing numerous one-character variables, especially in contexts involving multiple numerical values, poses a significant challenge even for humans.

We introduce the Semantically Rich Variable Substitution Method (SemRiVas) as an improvement of prior work that leverages symbolic variables to aid LLMs in solving MWPs. Our research contributes to the literature in the following ways:

1. Our approach eliminates the need for LLMs to possess programming proficiency and perform arithmetic operations, making it universally applicable as an evaluation method.

2. We commit to releasing all code and datasets associated with our method, facilitating its adoption for evaluating LLM performance on MWPs.

3. Our method demonstrates superior accuracy compared to previously employed One-Character Substitution Methods.

2. Background

2.1. Large Language Models

Large Language Models(LLMs) are advanced natural language processing systems that utilize different machine learning techniques to comprehend and generate text in human language. With the development of Transformer architecture (Vaswani et al., 2023), massive models with billions of parameters have been proposed. Some of the notable examples include ChatGPT (OpenAI, 2022) , Phi1.5 (Li et al., 2023), and Llama2-7B (Touvron et al., 2023). These models demonstrated remarkable performances in several natural language processing tasks, including machine translation, common sense reasoning, summarizing, planning, reasoning, and coding.

2.2. Math Word Problems

Math Word Problems (MWPs) are mathematical problem-solving scenarios within a contextualized linguistic framework. Solving these problems necessitates the translation of the verbal description into mathematical expressions (Meadows and Freitas, 2023), such as equations or inequalities, which requires analytical thinking and reasoning skills. Recent works have prevalently utilized math word problems to evaluate LLM’s reasoning capabilities (Li et al., 2023; Chowdhery et al., 2023).

2.3. Chain of Thought Prompting

Chain of Thought Prompting (CoT) (Wei et al., 2023) is a strategy that aims to decompose a problem into several intermediary operations to reduce the problem’s complexity, resulting in better performances. Specifically, when employing CoT, we prompt the

SemRiVas

Question
James has number-of-books-James-has books. If he buys number-of-books-James-buys books, how many books he has?
Answer
James will have number-of-books-James-has + number-of-books-James-buys books.

One-Character Substitution

Question
James has w books. If he buys 2x books, how many books he has?
Answer
James will have w + 2x books.

Figure 2: Comparison of Methods

LLM with instructions such as "Let's think step by step," directing the LLM to follow a divide-and-conquer approach. Researchers have shown that this intuitive and easy-to-implement approach exhibits remarkable accuracies when tested with math word problems.

2.4. Program-Aided Language Models

When solving math word problems, LLMs have been shown to encounter difficulties in performing arithmetic. With Program-Aided Language Models (PaL)(Gao et al., 2023), researchers aimed to utilize LLMs' coding abilities to address math word problems indirectly. Instead of directly providing the answer, the approach involves making the model generate a Python code that would, in turn, produce the correct result. In this method, once the Python code is generated, it is given to a third-party Python interpreter to execute the operations. By employing this strategy, they aimed to minimize the errors stemming from LLMs' inability to perform simple math. They demonstrated that PaL achieved high accuracies in their tests with LLMs capable of coding.

2.5. Solving Math Word Problem's Symbolically

Symbolic methods (Ferreira et al., 2022) represent quantities and relationships using symbols or variables, allowing for systematic manipulation and solution. Symbolic methods make sense because they reduce the likelihood of arithmetic errors and enable the application of mathematical reasoning and algorithms to complex scenarios.

The most notable work concerning our domain in this research has used (w, x, y, z) (Gaur and Saunshi, 2023) variables to substitute numerical values in the SVAMP (Patel et al., 2021) Dataset, then tested LLM's ability on this new augmented dataset. We will cover more about that method in the Baselines subsection.

3. Methods

In this study, we introduce SemRiVas: Semantically Rich Variable Substitution method. The SemRiVas method primarily aims to replace numbers in Math Word Problems (MWPs) with self-explanatory variables. This substitution facilitates differentiation between variables, particularly in lengthy questions, and delegates calculations to a third-party calculator to reduce arithmetic errors. We utilize the few-shot prompting technique for consistent and accurate responses.

3.1. Semantically Rich Symbolic Variables

Using symbolic variables instead of numerical values can enhance model performance in solving MWPs. Models often commit arithmetic errors while maintaining correct reasoning, as illustrated in Figure 1. Employing a simple third-party calculator can rectify these errors efficiently. Hence, substituting variables in questions and later solving them appears logical to improve accuracy. Previous methods have utilized one-character variables such as [w, x, y, z], [p, q, r, s], however, we posit that one-character variables might confuse models, especially when there are many of them (Gaur and Saunshi, 2023).

Instead of replacing numbers with one-character variables, we advocate substituting variables that describe the number's purpose. For instance, as depicted in Figure 2, we replace numbers with variables like "number-of-books-James-has." We hypothesize that in a question with multiple numbers, it would be challenging to track one-character variables and recall each variable's significance. Conversely, using self-explanatory variables could aid model comprehension.

3.2. Few-Shot Prompting

Few-Shot Prompting is a strategy where question-answer pairs are compiled in the desired format of model responses. Subsequently, the model is

prompted with these question-answer examples, followed by the original query. This approach guides the model to adhere to specified structural properties and provides concrete examples. It is particularly valuable for evaluation, facilitating the generation of extractable and quantifiable examples.

We have opted for 8-shot prompting in all our evaluations. These prompts were drawn from GSM-8K (Cobbe et al., 2021) and customized for each method addressed in this study. All prompts will be made available as open-source resources to facilitate the adoption of our method.

4. Results and Discussion

4.1. Baselines

In this study, we opted for the GSM8K benchmark (Cobbe et al., 2021) for evaluations, given its widespread use in the literature. The GSM8K dataset consists of high/middle-school level Mathematical Word Problems (MWP) of high quality. Another reason for selecting GSM8K was to examine the impacts of variable-substitution methods on both short and long-context questions. Previous studies utilized the SVAMP (Patel et al., 2021) dataset, which is constrained to a maximum of 4 numbers, whereas GSM-8K encompasses numbers ranging up to 8.

We chose GPT-3.5, specifically "gpt-3.5-turbo," as our base LLM due to its strong coding skills, enabling accurate PaL (Gao et al., 2023) calculation and prompt-following abilities for smoother processes.

To evaluate our method against previous approaches, we assessed PaL, CoT, and a One-Character Substitution Method similar to prior work that utilized [w,x,y,z].

4.2. Evaluation Strategy

We evaluated all four methods on 200 randomly selected QA pairs from GSM-8K. (Due to financial constraints, only 200 were selected). The process of number replacement to generate the dataset for the SemRiVas method was initially assisted by an LLM and later supervised by humans to address any errors. We used a simple Python program to generate the dataset for the One-Character Substitution method, as it didn't require specific variable names for each number.

The models were prompted with similar prompts differing only in response format, not in questions.

For the PaL method, we offloaded the generated responses into a python interpreter for evaluation. For the One-Character Substitution, and SemRiVas method, we simply exchanged variables with the original numbers, and evaluated from there using a simple python calculator. CoT didn't require any additional steps to evaluate, we simply took the final answer it wrote.

We maintained consistent hyperparameters for all methods: temperature parameter at 0.7 and Top-P parameter at 1. We observed in a smaller subset of data that these parameter values have proven to be highly robust compared to higher or lower settings.

Table 1: Accuracy Rates of Each Method (8-Shot)

PaL	CoT	SemRiVas	One-Character S.
48%	84%	44%	34.5%

4.3. Results

According to our research results (Table 1), the GPT-3.5 model prompted with the SemRiVas method achieved a 44% success rate across 200 questions. PaL achieved 48% accuracy, CoT achieved 84%, and the One-Character Variable Substitution method achieved 34.5%.

We have also analyzed the success rates of SemRiVas and the One-Character variable method on the 50 longest/shortest questions to see whether our hypothesis that using semantically rich variables enhances performance in long-context settings. We found that SemRiVas was 38% more successful than the One-Character Variable Substitution method on the longest questions, while it was marginally less successful (7%) than the one-character variable substitution method on the shortest questions.

4.4. Discussion

Our method, designed to measure LLMs' logical and reasoning abilities, achieved higher accuracy than the one-character variable substitution method, demonstrating the effectiveness of semantically rich variables. This increase in accuracy underscores the effectiveness of our approach for solving MWPs symbolically using large language models.

However, we can see that LLMs seem to be struggling to solve MWPs using symbolic variables, as both symbolic methods didn't surpass the accuracy rate of CoT or PaL.

That said, due to budget and manpower limitations, our experiments were confined to the GPT-3.5 model and a smaller subset of the GSM-8K (Cobbe et al., 2021) benchmark. Further testing on the entire GSM-8K dataset and with different models could provide a clearer picture of our approach's success rate.

5. Conclusion

In conclusion, our research introduces the Semantically Rich Variable Substitution Method (SemRiVas) as a novel approach to evaluate Large Language Models (LLMs) on Mathematical Word Problems (MWPs). By utilizing descriptive variable names instead of generic symbols, SemRiVas aims to improve LLMs' problem-solving abilities, particularly in long-context settings. Our experimental results demonstrate the superior accuracy of SemRiVas compared to previous symbolic methods, highlighting its effectiveness in enhancing LLM reasoning capabilities. However, our findings also suggest that LLMs still face challenges when solving MWPs using symbolic variables, as they did not surpass the accuracy rates achieved by methods such as CoT or PaL. Further research is needed to understand and address these challenges comprehensively.

6. Acknowledgements

The author would like to thank Clayton Greenberg for his valuable mentorship.

7. Bibliographical References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, and ... Barret Zoph. 2023. Gpt-4 technical report.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and ... Noah Fiedel. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems.

Deborah Ferreira, Mokanarangan Thayaparan, Marco Valentino, Julia Rozanova, and Andre Freitas. 2022. To be or not to be an integer? encoding variables for mathematical text. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 938–948, Dublin, Ireland. Association for Computational Linguistics.

Simon Frieder, Luca Pinchetti, , Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Petersen, and Julius Berner. 2023. Mathematical capabilities of chatgpt. In *Advances in Neural Information Processing Systems*, volume 36, pages 27699–27744. Curran Associates, Inc.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Vedant Gaur and Nikunj Saunshi. 2023. Reasoning in large language models through symbolic math word problems.

Joy He-Yueya, Gabriel Poesia, Rose E. Wang, and Noah D. Goodman. 2023. Solving math word problems by combining language models with symbolic solvers.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1. Curran.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2023. A survey of deep learning for mathematical reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14605–14631, Toronto, Canada. Association for Computational Linguistics.

Jordan Meadows and André Freitas. 2023. Introduction to Mathematical Language Processing: Informal Proofs, Word Problems, and Supporting Tasks. *Transactions of the Association for Computational Linguistics*, 11:1162–1184.

Jordan Meadows, Marco Valentino, Damien Teney, and Andre Freitas. 2023. A symbolic framework for systematic evaluation of mathematical reasoning with transformers.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2021. Investigating the limitations of transformers with simple arithmetic tasks.

OpenAI. 2022. Introducing chatgpt.

Tim Orchard and Leszek Tasiemski. 2023. The rise of generative ai and possible effects on the economy. *Economics and Business Review*, 9:9.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Alessandro Stolfo, Zhijing Jin, Kumar Shridhar, Bernhard Schölkopf, and Mrinmaya Sachan. 2023. A causal framework to quantify the robustness of mathematical reasoning with language models. pages 545–561.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and ... Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. 2023. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment.