Beyond the Hype: Identifying and Analyzing Math Word Problem-Solving Challenges for Large Language Models

Romina Soledad Albornoz-De Luise[1], **David Arnau**[2], **Pablo Arnau-González**[1], **Miguel Arevalillo-Herráez**[1]
[1]Departament d'Informàtica, Universitat de València,
Avinguda de la Universitat s/n, Burjassot, 46100, València, Spain
[2]Departament de Didàctica de la Matemàtica, Universitat de València,
Avinguda dels Tarongers, 4, València, 46022, Spain
{romina.albornoz, david.arnau, pablo.arnau, miguel.arevalillo}@uv.es

Abstract

Despite not being explicitly trained for this purpose, models like Mistral and LLaMA have demonstrated impressive results across numerous tasks, including generating solutions to Mathematical Word Problems (MWPs). A MWP involves translating a textual description into a mathematical model or equation that solving it. However, these models face challenges in accurately interpreting and utilizing the numerical information present in the MWP statements, which can lead to errors in the generated solutions. To better understand the limitations of LLMs, we analyzed the MWP where models failed to accurately solve problems from the SVAMP dataset. By categorizing these MWPs, we identify specific types of problems where the models are most prone to errors, providing insights into the underlying challenges faced by LLMs in problem-solving scenarios and open new modeling opportunities. By understanding the expected errors, researchers can design strategies to adequately model problems more effectively and choose the most suitable LLM for solving them taking into account each model's strengths and weaknesses.

1 Introduction

LLMs have expanded the boundaries of understanding and generating natural language (Karanikolas et al., 2024). Moreover, recent research has found LLMs to be capable of producing high-quality source code (Rozière et al., 2024). LLMs excel at producing text sequences, but also show reasoning capabilities that have been previously applied to Math Word Problem (MWP) Solving (Kojima et al., 2023) by transforming the MWP in natural language to the mathematical language.

In this context, recent research (Arnau-González et al., 2024) has explored LLMs in the context of education by producing source-code that can be compiled into a solution graph for tutoring and supervision purposes.

This paper aims to investigate the types of MWP statements that LLMs have difficulties solving by analyzing incorrect samples produced in previous studies. To this end, we select the SVAMP dataset and three different models: OpenMath/Mistral-7B from Nvidia, Llama3-8B[1], and CodeLlama 34B[2] (Rozière et al., 2024), which demonstrated high performance in MWP-solving task. By focusing on problem statements where these models failed, we identify patterns in the sources of errors. The provided analysis[3] can direct research towards a better understanding of the reasoning limitations of language models.

2 Background and related work

A MWP model solution can be understood as the result of reducing the initial MWP to a graph of mathematical relationships between quantities.

Consider the MWP where we have bought a car and paid 12 bills of 400 euros each. If the car costs 12 800 euros, we need to determine how much money is left to pay. A possible problem model would establish the following relationships: The total price of the car equals the sum of the money already paid and the money left to pay. Furthermore, the money already paid is calculated by multiplying the value of a single bill by the number of bills paid.

Automatically solving a math problem articulated in natural language presents a significant challenge, necessitating both comprehension and accurate reasoning. This process requires techniques to extract not only the quantities explicitly stated in the MWP but also those implied by terms such

[1]https://llama.meta.com/llama3/
[2]https://llama.meta.com/code-llama/
[3]Analysed dataset available in https://zenodo.org/records/12771266

as “twice”, “half” or “left”. Additionally, solving the MWP demands an understanding of the relationships among these quantities, the identification of the target quantity, and the sequence of operations needed to achieve the final result. In essence, solving a problem from natural language is a task primarily concerned with knowledge extraction and the identification of advanced relationships (Jie et al., 2022a). Recent studies have shown that LLMs show a problem-solving ability similar to that of children, despite differences in the type of MWP they are able to solve best (Arnau-Blasco et al., 2024).

The task of automated MWP solving has been a topic of interest in the literature since the 1960s, inspiring a recent survey (Zhang et al., 2020). Recent efforts in solving Mathematical Word Problems (MWPs) have concentrated on constructing expression (or equation) trees. These methods focus on creating arithmetic expressions by forming equivalent trees. However, due to the exponential growth of the search space as the number of quantities increases, alternatives leveraging reinforcement learning techniques have been explored (Wang et al., 2018).

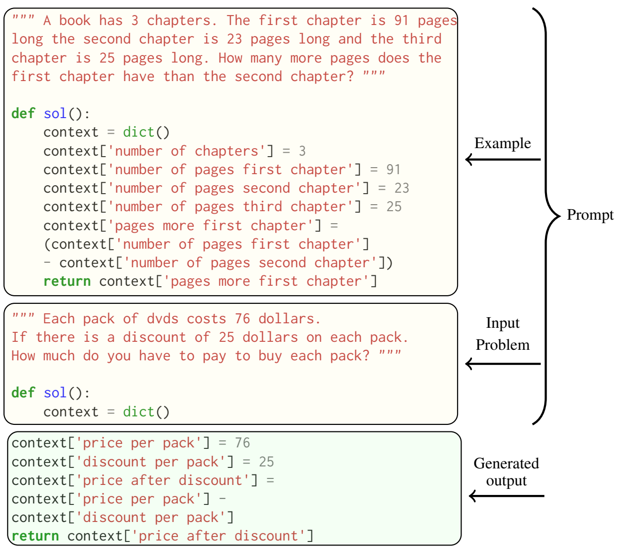
In the last year, prompting techniques have been developed to force reasoning on decoder-only transformers (Kojima et al., 2023). Moreover, the introduction of new-generation LLMs like Llama2 and Mistral has also led to new studies in the field. In this direction, (Arnau-González et al., 2024) proposed a method that incorporates MWP solving, quantity value assignment and naming as well as capabilities for establishing relationships, without the need for fine-tuning the underlying LLM.

3 Dataset

The SVAMP MWP dataset (Patel et al., 2021) consists of 1000 elementary-level arithmetic word problems, each solvable by expressions requiring no more than two operators. This dataset provides annotated solutions for each MWP. SVAMP was selected for evaluating and analyzing the performance of LLMs in solving MWPs, being widely recognized as one of the most challenging datasets for arithmetic MWP solving (Patel et al., 2021; Jie et al., 2022b).

3.1 Studied samples

In a previous study (Arnau-González et al., 2024), the authors have developed a method where a LLM is prompted with an example MWP statement alongside a corresponding correct Python function named sol(), which solves the MWP. Additionally, the model receives the MWP statement to be solved and a partially defined Python function. The model then completes the function by defining quantities and their relationships, and returning the requested result. Figure 1 shows this process with annotations highlighting the different parts of the prompt and the generated output. The accuracy of the solution can be verified by executing the generated Python code and comparing its result to the expected solution.



```
""" A book has 3 chapters. The first chapter is 91 pages
long the second chapter is 23 pages long and the third
chapter is 25 pages long. How many more pages does the
first chapter have than the second chapter? """

def sol():
    context = dict()
    context['number of chapters'] = 3
    context['number of pages first chapter'] = 91
    context['number of pages second chapter'] = 23
    context['number of pages third chapter'] = 25
    context['pages more first chapter'] =
    (context['number of pages first chapter']
    - context['number of pages second chapter'])
    return context['pages more first chapter']
```

```
""" Each pack of dvds costs 76 dollars.
If there is a discount of 25 dollars on each pack.
How much do you have to pay to buy each pack? """

def sol():
    context = dict()
```

```
context['price per pack'] = 76
context['discount per pack'] = 25
context['price after discount'] =
context['price per pack'] -
context['discount per pack']
return context['price after discount']
```

Figure 1: Python code with generated output example: Prompts in yellow sections and an example of LLM generated output in green.

The samples provided in (Arnau-González et al., 2024) are an attempt to automatically solve MWP using Python code.

Originally, the published samples contained Python source code produced by 19 different LLMs, on the problems contained in the SVAMP and GSM-8k datasets. Each model was used to generate 10 independent solutions for each MWP for three different temperature settings $\tau \in \{0.1, 0.3, 0.5\}$. In this study, we focus our analysis on the top-3 top-performing models based on the best accuracy values, computed considering only the first of the 10 solutions generated. These models were OpenMath-Mistral-7b, Llama3-8b, and Codellama-34b, when using $\tau = 0.1$.

4 Analysis

We focused our analysis on the characteristics of MWPs that LLMs tend to solve incorrectly. These

MWPs have been selected by studying the generated samples obtained in previous studies (Arnau-González et al., 2024). Out of the three studied models we have chosen CodeLlama-34b as the worst model, as it provided the worst accuracy on the SVAMP dataset.

We decided to delve into those problems that the worst performing model (CodeLLama-34b) failed to solve accurately, in order to draw some conclusions on the structure of said MWPs.

After an initial visual inspection, we found specific patterns in the MWPs for which models tend to fail and categorized them to better understand the limitations of the models based on the type of MWP they are trying to solve, into at least one of the following three types: MWP with unfeasible solutions (US), MWP with unnecessary quantities (UQ), and MWP involving comparisons (CP).

MWPs that fall into the Unfeasible Solution category are MWPs that, although can be solved analytically, the solution obtained is not practical or possible in the presented scenario. This happens, for instance, when one or more quantities in the solution take an integer or real value where only natural numbers are physically possible. An example of this type of statement is *"A waiter had 12 customers. After some left he still had 14 customers. Then he got 10 new customers. How many customers does he have now?"*. This problem implies that somehow -2 customers left the restaurant.

MWPs with the Unnecessary Quantities category contain one or more quantities that are not required to solve the problem. We have observed that most LLMs have a natural tendency to use all the quantities present in the statement to produce a solution, typically leading to errors in the reasoning. A good example of these MWP is *"Rebecca wants to split a collection of eggs into groups of 6. Rebecca has 18 eggs 72 bananas and 66 marbles. How many groups will be created?"*. In this case, bananas and marbles amounts are unnecessary in determining how many groups of eggs will be created. However, models will still add it to the number of eggs.

What does the Table 2 with Mann-Whitney U rank test show? In the table are the statistics? Finally, MWPs falling in the Involving a Comparison category, have their statements asking or involve a direct comparison between two quantities. This often confuses models as sometimes are not capable of capturing these relationships appropriately. An example of this type of problem is *"There were 3 dollars in Olivia's wallet. She collected 49 more dollars from an ATM. After she visited a supermarket there were 49 dollars left. How much more money did she collect at the ATM than she spent at the supermarket?"*. In this statement, the problem question is a comparison between money collected at the ATM and the amount spent at a supermarket. Table 1 summarizes the types of MWP statements along with brief descriptions of each category.

Other categories have been created by combining the previously identified categories, US$\wedge$UQ, US$\wedge$CP, UQ$\wedge$CP, and US$\wedge$UQ$\wedge$CP. These categories contain the samples that can be tagged in more than one category. Finally, an additional category UNIDENTIFIED has been created, containing the MWPs that have not been labelled in any of the previous categories. This type of problems has a simple structure with no irrelevant comparison elements or quantities.

MWPs that CodeLlama-34B fails to solve correctly have been classified by two independent volunteer annotators. The annotators were asked to tag which of the identified problems were present in each of the selected MWPs. According to both annotators' responses we computed Cohen's Kappa for each of the three separate problems. In all cases, we obtained a $\kappa > 0.6$, indicating substantial agreement was achieved by the annotators. Finally, since annotator #1 had more experience in the field, it was decided to choose samples from that annotator. The analysis of the classification of the selected samples shows that the MWPs can be classified into at least one of the identified categories in over 70% of the MWPs which CodeLlama fails to solve.

CodeLlama fails to solve 45.8% of problems in the UQ category, indicating that these problems present the greatest challenge for the model. 37.8% in the CP category, and 21.8% in the US category. Additionally, 14.5% of the problems fall into both the US and UQ categories, 10.9% fall into both the US and CP categories, 16% fall into both the UQ and CP categories, and 6.9% fall into all three categories: US, UQ, and CP. Finally, 29.1% of the problems do not fall into any of these specified categories. An intriguing observation arises when examining this category of the unsolved problems. These problems typically have clearer statements and generally require a straightforward operation to reach a solution. Despite their apparent simplicity, CodeLlama still faces notable challenges in solving these problems.

In summary, a total of 275 MWPs (those which CodeLlama failed to solve) were selected for anal-

Table 1: Features of MWPs Statements and Brief Description

Category of MWP Statements	**Description**
Unfeasible Solution (US)	MWPs that can be solved analytically, but the solution obtained is not possible in the presented scenario
Unnecessary Quantities (UQ)	MWP statements contain one or more quantities that are not required to solve the problem
Involving a Comparison (CP)	MWP have statements asking or involve a direct comparison between two quantities
UNIDENTIFIED	MWP statements in this category do not exhibit any of the previously mentioned characteristics

Table 2: Category-Wise MWP Statements where Mistral and Llama Models fail Focused on CodeLlama Failures.

	US	**UQ**	**CP**	**US∧UQ**	**US∧CP**	**UQ∧CP**	**US∧UQ∧CP**	**UNIDENTIFIED**
Mistral	43.33	45.24	36.54	40.00	33.33	31.82	26.32	22.50
Llama	61.67	50.79	62.5	50.00	66.67	45.45	47.37	36.25
Total	60	126	104	40	30	44	19	80

The table presents the percentage of problems within each category that are incorrectly resolved by Mistral and Llama models, focusing on problems that CodeLlama initially failed to solve. Additionally, it includes the number of MWPs identified within each category.

Table 3: Mann-Whitney U rank test

	US	**UQ**	**CP**
Mistral	0.0044	0.0004	0.0204
Llama	0.0014	0.0207	0.0002

P-values from the Mann-Whitney U rank test comparing error rates between problems categorized as US, UQ, and CP versus those in the UNIDENTIFIED category. Each sample consists of independent sets of problems, where the identified categories (US, CP, UQ) are compared against the UNIDENTIFIED problems to assess differences in error rates.

ysis in the top two top-performing models. Table 2 displays the error rates for both Mistral and Llama-3 models for the tagged samples, and all the possible tag combinations. A first analysis reveals that the identified categories are indeed problematic also for these models. This is shown by the error rate being higher in all the identified categories and their combinations.

This observation is further supported by a hypothesis test, where the alternative hypothesis (H_a) posits that the distribution of error rates for each identified category (UQ, US, CP) is significantly higher than that of errors in the UNIDENTIFIED category. Specifically, we hypothesize that the error rate in problems identified with UQ (104 problems), US (60 problems), and CP (126 problems) is greater than the error rate in the UNIDENTIFIED category (80 problems). The tested distributions are independent, and since the assumptions of the Students's test are violated (homoscedasticity and normality of the date), we choose a non-parametric alternative. The null hypothesis (H_0) for each comparison is that there is no difference between the error distributions of the identified category and the UNIDENTIFIED category. In other words, H_0 asserts that the two distributions have the same median error rate. The tested distributions represent the proportion of problems solved correctly (is_correct) within each category. These distributions are independent and consist of binary outcomes (True/False). The results, as shown in Table 3, indicate that for all identified categories (UQ, US, CP), there are significantly more errors compared to the UNIDENTIFIED category. This supports our hypothesis that the identified categories represent problem types that are systematically more challenging for the models.

Within the 275 problems incorrectly solved by CodeLlama, we analyzed the errors made by the Mistral and Llama models. Mistral accounted for errors in 173 problems, with 60% of these errors being attributable to the same issues present in the CodeLlama model. Similarly, Llama had 247 problems with erroneous solutions, and 58% of these errors could be explained by the same mistakes made by CodeLlama. This indicates that we were able to identify the characteristics of MWPs of 60% of Mistral's errors and 58% of Llama's errors, respectively. Through this analysis, we can understand a significant portion of the common sources of errors in both Mistral and Llama. In Table 2, we analyze the percentage distribution of incorrectly resolved problems across each category.

5 Conclusion and Future Work

In this work, we have analyzed the performance of three LLMs in solving MWPs on the SVAMP dataset and categorized the sources of errors. The categorization of MWP statement error sources reflects specific patterns in which the models fail to correctly solve these problems. Identifying these patterns provides valuable insights into the limitations of current LLMs.

The provided study shows that, in general, there are three categories of challenging problems for which models tend to generate wrong solutions. Moreover, the results show that statistically, models tend to fail significantly more in problems that fall into one of these categories than in any other type of problems. The fact that these categories can be identified, and the shown difference in performance in these categories shows that LLMs are still weak for certain types of reasoning. The identification of the reasons that cause an incorrect solution in the case of statements that cannot be classified in any of the 3 identified categories is not straightforward. The initial inspection showed that these problems apparently have a clear statement, and there is no reason as to why the models consistently fail to solve the problem. A possible explanation of this issue might be related to the type of relations encoded in the MWPs, as suggested by previous research (Arnau-Blasco et al., 2024).

The presented analysis is a work in progress which examines the characteristics of MWP statements where the selected LLMs fail to provide correct solutions. The initial categorical classification offered as part of this work serves as a preliminary step towards modeling math problems based on categories that reflect the likelihood of being correctly solved by different LLMs. Future work will continue analysing the samples for which the top-performing models fail, in order to gather insights into the reasoning gaps and generate strategies to overcome such failures. We also plan to examine and compare the error rates across different categories made by LLMs with those made by real students.

Acknowledgements

This work has been supported by project PID2023-150960NB-I00, funded by the Spanish Ministry of Science, Innovation and Universities and the European Union; project CIGE/2023/063 and grant CIAPOS/2022/063 funded by Generalitat Valenciana; project TED2021-129485B-C42 funded by MCIN/AEI/10.13039/501100011033 and the European Union "NextGenerationEU"/PRTR; and grant PRE2019-090854, funded by MCIN/AEI/10.13039/ 501100011033 and "ESF Investing in your future".

References

Jaime Arnau-Blasco, Miguel Arevalillo-Herráez, Sergi Solera-Monforte, and Yuyan Wu. 2024. Using large language models to support teaching and learning of word problem solving in tutoring systems. In *Generative Intelligence and Intelligent Tutoring Systems*, pages 3–13, Cham. Springer Nature Switzerland.

Pablo Arnau-González, Stamos Katsigiannis, Ana Serrano-Mamolar, and Miguel Arevalillo-Herráez. 2024. Result outputs for "Automated Math Word Problem solving and quantity identification using Large Language Models for code synthesis".

Zhanming Jie, Jierui Li, and Wei Lu. 2022a. Learning to reason deductively: Math word problem solving as complex relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5944–5955, Dublin, Ireland. Association for Computational Linguistics.

Zhanming Jie, Jierui Li, and Wei Lu. 2022b. Learning to reason deductively: Math word problem solving as complex relation extraction. *Preprint*, arXiv:2203.10316.

Nikitas Karanikolas, Eirini Manga, Nikoletta Samaridi, Eleni Tousidou, and Michael Vassilakopoulos. 2024. Large language models versus natural language understanding and generation. In *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics*, PCI '23, page 278–290, New York, NY, USA. Association for Computing Machinery.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners. *Preprint*, arXiv:2205.11916.

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.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code. *Preprint*, arXiv:2308.12950.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Dongxiang Zhang, Lei Wang, Luming Zhang, Bing Tian Dai, and Heng Tao Shen. 2020. 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.