Solving Math Problem with In-Context Error

Keya Hu
ACM Class 2021
hu_keya@sjtu.edu.cn

Yijin Guo
ACM Class 2021
guoyijin@sjtu.edu.cn

Hang Ruan
ACM Class 2021
zzrh01@sjtu.edu.cn

Abstract

Nowadays, in-context learning (ICL) excels in tasks requiring complex reasoning. This paper investigates how typical ICL factors affect math problem-solving performance and examines the impact of erroneous examples. First, we analyze how varying examples—such as language styles, difficulty levels, and response lengths, affect performance. Our findings indicate that example format significantly impacts performance. Second, we examine the models' tolerance to errors by modifying the MATH dataset, revealing that changes like altering number signs or modifying numbers by more than two digits greatly influence ICL performance. Additionally, we introduce a weak-to-strong learning scenario in which outputs from a weaker model serve as examples for stronger models. We show that this operates within our identified error tolerance, effectively enhancing the performance of stronger models.

1 Introduction

In-context learning (ICL)(Brown et al., 2020), an advanced machine learning technology, has proven effective in various tasks requiring complex reasoning. Some research investigates the mechanisms behind ICL in large language models(LLMs)(Min et al., 2022), examining the effectiveness of demonstrations, the models' reliance on semantic reasoning over symbolic logic, and the impact of structured chain-of-thought prompting on problem-solving(Madaan et al., 2023)(Tang et al., 2023).

However, it remains to be thoroughly investigated whether the typical factors that influence ICL performance are still applicable in solving mathematical problems and how they affect performance. Moreover, since math problems are more prone to errors, it is worth researching the impact of erroneous examples on the model's reasoning capabilities. This paper conducts an in-depth study of both problems. We comprehensively analyze the mechanism of ICL in mathematical problems by changing the forms or making errors in examples and then find out the largest possible changes - which is the model's tolerance of errors - in examples while still maintaining the LLMs' performance.

Firstly, we change the examples to analyze how ICL affects the performance of LLMs in math problems. We explore the impact of various forms of examples on the effectiveness of mathematical reasoning, such as language styles, levels of difficulty, lengths of responses, and so on. Meanwhile, we investigate the effects of correcting errors within these examples, focusing on computational and derivation errors. We modify the examples in the GSM8K(Cobbe et al., 2021) dataset. Experimental results indicate that format has a stronger impact on performance than forms. Among forms that have a strong effect, the appearance of formulas in solutions seems to be important. The length of examples does affect the performance, but it varies among different models, while easier examples lead to better performance consistently.

Secondly, we investigate the methods that result in the most significant changes without affecting model performance, essentially exploring the model's tolerance to the degree of errors in the examples. We modify the examples in the MATH(Hendrycks et al., 2021) dataset and explore the tolerance to variations in templates and the scale of computational errors in the problems. Experimental results indicate that changing the signs of numbers and changing numbers for more than two digits have a high impact on in-context learning performance.

Furthermore, for the tolerance we have summarized above, we find an application scenario called weak-to-strong learning, in which the outputs of a weaker model for some questions act as examples for stronger models in ICL. The examples generated by the weaker model may contain some errors, so our tolerance can be a standard to measure whether a model can act as the weaker model in this scenario to enhance the performance of the stronger models. We choose Llama2 7B as the weaker model and find it's within our tolerance. Experimental results also prove that this model is really helpful for the performance of stronger models.

In summary, our main contributions are:

1. We explore the impact of various features of examples and errors within these examples on the effectiveness of mathematical reasoning.
2. We investigate the model's tolerance to the degree of errors in the examples, exploring the methods that result in the most significant changes without affecting model performance.
3. We further propose a novel application of the tolerance in a weak-to-strong learning scenario and conduct a successful experiment for it.

Our repository is available here: In-context-Reasoning-with-Errors.

2 Related Work

2.1 In-context Learning And CoT

In recent years, in-context learning has emerged as a novel machine learning paradigm in natural language processing and is receiving widespread attention. This learning approach allows models to respond solely based on the observation of input context without explicit task-specific training (Brown et al., 2020). This strategy has been extensively applied in OpenAI's GPT series of models, where massive text pre-training enables the models to understand and generate responses highly relevant to the context provided (Radford et al., 2019). Concurrently, the "Chain of Thought" (CoT) method has been introduced to enhance model comprehension and problem-solving capabilities. CoT guides the model to generate a series of intermediate thought steps to form a final answer to complex problems(Wei et al., 2022). This method is particularly suited for tasks requiring reasoning and extended logical coherence, such as solving mathematical problems or complex reasoning tasks. By integrating In-context Learning and CoT, models can capture the linguistic patterns of the given context and demonstrate how to reach problem resolution through step-by-step reasoning. This significantly enhances the transparency and effectiveness of models in handling complex problems.

2.2 Mechanisms of In-Context Learning

The efficiency of in-context learning (ICL) in adapting large language models (LLMs) to new tasks has been a focal area of recent research, significantly advancing our understanding of how LLMs process and respond to prompts. Key works in this domain include examining the foundational mechanisms that enable effective ICL by analyzing different demonstration methods(Min et al., 2022). Complementary to this, another study explores the nature of reasoning within these models, positing that LLMs primarily leverage semantic associations rather than engaging in symbolic reasoning(Tang et al., 2023).

Further deepening this line of inquiry, research on how structured thought processes, when embedded within prompts, can enhance the model's reasoning capabilities(Madaan et al., 2023), highlights the strategic use of chain-of-thought (CoT) prompting to guide LLMs through complex reasoning tasks more effectively.

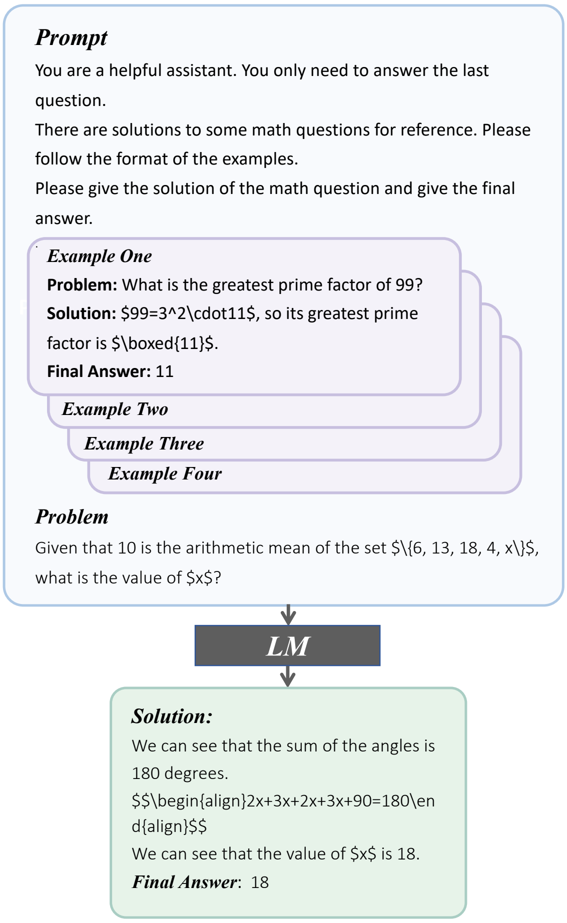
2.3 Weak-to-strong Learning

In recent years, significant progress has been made in Weak-to-strong Learning. This approach primarily focuses on whether models with limited capabilities can effectively guide the development of more advanced, stronger models. Research(Burns et al., 2023) demonstrated the preliminary feasibility of this method in tasks such as classification, chess, and reward model building, proving that basic models can assist in developing more complex algorithms to a certain extent.

However, applying this learning paradigm to more advanced complex reasoning tasks has not yet been extensively researched and verified. These complex reasoning tasks far exceed simple extrapolation or pattern recognition and involve deep logical thinking and decision-making. Studies(Qiao et al., 2023) have highlighted that understanding whether large language models (LLMs) can mimic or surpass human capabilities in these areas is crucial for understanding and advancing artificial intelligence development.

3 Methods

We utilize different methods to obtain some questions with solutions and final answers, which are



Prompt

You are a helpful assistant. You only need to answer the last question.

There are solutions to some math questions for reference. Please follow the format of the examples.

Please give the solution of the math question and give the final answer.

Example One

Problem: What is the greatest prime factor of 99?

Solution: $99=3^2\cdot11$, so its greatest prime factor is $\boxed{11}$.

Final Answer: 11

Example Two

Example Three

Example Four

Problem

Given that 10 is the arithmetic mean of the set $\{6, 13, 18, 4, x\}$, what is the value of x?

Solution:

We can see that the sum of the angles is 180 degrees.

$$\begin{align}2x+3x+2x+3x+90=180\end{align}$$

We can see that the value of x is 18.

Final Answer: 18

Figure 1: The architecture of our method, with four in-context learning examples and one question.

used as examples in ICL. Then we prepare some questions to ask the LLMs. In each method, a prompt is composed with some fixed examples and one question. Our architecture is shown in Figure 1.

In order to explore the mechanism of in-context learning and to identify the tolerance of incorrect examples in mathematical problems, we modify the examples using two types of ideas. One is to modify the examples to conform to certain features, while the other involves introducing errors and disruptions to initially excellent examples.

Table 1 shows one specific example for part of the method.

3.1 Formats of examples

This part is about the formats of examples. The original dataset has a typical format of Chain of thoughts(CoT) with solutions and answers. We do some modifications in the format to reveal the mechanism of ICL and CoT influencing the performance of LLM in math problems.

Typical (with CoT) This is the basic format of CoT initially given in the dataset. Firstly it puts forward the math problems, then gives out the solutions that deal with the problem step by step and finally shows the results. This format is shown in Figure 1. In the following experiments, it is also denoted as "Human-Written Examples".

Direct This is a format that deletes the solutions in the typical format. It only includes the math questions and directly gives the final answers.

Swap This method swaps solutions with final answers and first give out the final answers and then the solutions.

3.2 Features of examples

There are some categories focusing on the features of math examples, which represent their characteristics in expression and difficulty. Considering the logical and rigorous nature of mathematical problems, we need to explore whether the different features of examples in terms of expression and difficulty will have a special impact on the performance of LLMs.

Length of solution We define the length of a solution based on the number of letters it contains. When the solution contains fewer than 100 letters, we classify the length of the solution as short. When the solution contains more than 100 letters but fewer than 1000 letters, we classify the length as middle. When the solution contains more than 1000 letters, we define the length as long. In this method, we compare the performance of samples with different length.

Difficulty We define the difficulty of a problem based on the number of reasoning sentences in the solution. When the solution contains 1 to 3 sentences, we define the problem difficulty as easy. When the solution contains 4 to 6 sentences, we define the problem difficulty as medium. When the solution contains 7 to 10 sentences, we define the problem difficulty as hard. In this method, we compare the performance of samples with different difficulty.

Style We change the examples into different styles of expression.

Storytelling. We transform the original problem description and solution into a storytelling style suitable for kindergarten children, incorporating metaphors and adding some easy-to-understand

Table 1: Some examples using our method in the MATH dataset

Problems: Solve for n, if $8^n \cdot 8^n \cdot 8^n = 64^3$.	
Method type	**Solution**
original	The equation, $8^n \cdot 8^n \cdot 8^n = 64^3$, can be written as $8^{3n} = 64^3$. We also know that $64 = 8^2$, so we can rewrite the equation as $8^{3n} = 8^{2(3)}$. Solving for n gives $n = \mathbf{2}$.
Styles	
formula only	$8^n \cdot 8^n \cdot 8^n = 64^3, 8^{3n} = 64^3.64 = 8^2, 8^{3n} = 8^{2(3)}.n = \mathbf{2}$.
statement only	Both sides of the equation can be rewritten so that the bases on both sides are equal. Then by making the exponents equal, the problem can be solved.
Errors	
abstract symbol	The equation, $a^n \cdot a^n \cdot a^n = b^c$, can be written as $a^{cn} = b^c$. We also know that $b = a^d$, so we can rewrite the equation as $a^{cn} = a^{d(c)}$. Solving for n gives $n = \mathbf{d}$.
wrong -1	The equation, $8^n \cdot 8^n \cdot 8^n = 64^3$, can be written as $8^{3n} = 64^3$. We also know that $64 = 8^{(-3)}$, so we can rewrite the equation as $8^{3n} = 8^{-3(3)}$. Solving for n gives $n = \mathbf{-3}$.
wrong 1	The equation, $8^n \cdot 8^n \cdot 8^n = 64^3$, can be written as $8^{3n} = 64^3$. We also know that $64 = 8^{(9)}$, so we can rewrite the equation as $8^{3n} = 8^{9(3)}$. Solving for n gives $n = \mathbf{9}$.
wrong 2	...
...	...

This table selects a question in MATH dataset. In *abstract symbol*, the question is also modified into "Solve for n, if $a^n \cdot a^n \cdot a^n = b^c$". The bolded numbers in "Solution" are the final answers.

phrases. In this method, we compare the performance of original styles with storytelling style.

Formula only. This method only modifies the solutions. It retains only the mathematical formulas in the solution and removes all the textual descriptions and guidance.

Statement only. This method only modifies the solutions. It retains only the textual descriptions and replaces the irreplaceable math processes - such as formula derivation and calculation - with descriptions of the derivation methods. The language style keeps consistent with the original examples.

3.3 Errors of examples

Whether for large language models or in the process of manually creating datasets, correctly answering mathematical problems is more prone to errors. Consequently, we make some mistakes in the examples with multiple methods to explore LLM's tolerance of errors.

Wrong Calculation This method modifies both the solutions and answers. It makes mistakes during basic operations such as addition, subtraction, multiplication, and division. It is worth mentioning that though there are calculation errors, it will not lead to the repetition errors, that is, in the derivation process, if a wrong answer is used later, it will still use the modified wrong result rather than the original correct one.

Moreover, we hope to identify the tolerance scale of computational errors. Therefore, we design examples with varying degrees of computational errors based on the absolute difference between the incorrect and correct answers. For example, if the absolute value is within 10(e.g. 3+2=8), we call this type as "wrong 1". If the absolute value is within 100 but larger than 10(e.g. 3+2=45), we call this type as "wrong 2". We also take positive and negative numbers into consideration and call this type as "wrong -1"(e.g. 3+2=-8).

Wrong Inference This method mainly modifies the solutions and may influence the answers. This method is a significant disruption to the original examples. In the solution, it may present imprecise, logically reversed, or even completely irrelevant derivation processes. To control variables, there is no computational errors in all the derivation steps, but incorrect derivations may still lead to wrong answers. Here is a example below at Table 2.

Abstract Symbol This method modifies the questions. the solutions and the answers in the examples. It substitute all the numbers with abstract symbols. Similar to *wrong calculation*, we don't contain any

Table 2: A example for wrong inference

Problem: The state of Virginia had 3.79 inches of rain in March, 4.5 inches of rain in April, 3.95 inches of rain in May, 3.09 inches of rain in June and 4.67 inches in July. What is the average rainfall amount, in inches, in Virginia?
Solution: It rained for a total of $3.79 + 4.5 + 3.95 + 3.09 + 4.67 = 20$ inches. The average amount should be 20 inched.
Final Answer: 20

repetition errors in modified examples.

It should be noted that in our dataset, there are some tricky examples whose derivation process and result calculation are related to specific numbers. Directly replacing them with abstract symbols may significantly alter the meaning of the examples. For such cases, we may replace all the numbers in the solution and answer into letters "$a, b, c, ...$", each specific number will correspond to one specific letter.

4 Experimental Setting

4.1 Dataset

We conducted our experiments on two datasets related to mathematical problems, the grade school math (GSM8K)(Cobbe et al., 2021) dataset and the MATH(Hendrycks et al., 2021) dataset.

GSM8K Dataset: The dataset comprises 8.5K high-quality grade school math problems crafted by human problem writers, segmented into 7.5K training problems and 1K test problems. We randomly selected four tasks in training problems as examples of in-context learning. We use the first 500 testing questions as our testing set.

MATH Dataset: The dataset contains seven different types of math problems. The seven types of math problems are algebra, counting and probability, geometry, intermediate algebra, number theory, pre-algebra, and pre-calculus. All these math problems are separated into five difficulties: the larger the number, the greater the difficulty. Considering the performance of our model, we ultimately conducted experiments only on the difficulty$= 1$ subset of algebra, counting and probability, intermediate algebra, and pre-algebra. The size of the algebra subset is 135, the counting and probability subset is 39, the intermediate subset is 52, and the pre-algebra subset is 86.

4.2 The model

We experiment with our method using the weak model to supervise stronger models with the following models.

For stronger models, we utilize two large language models without instruction-tuning as our strong model: Llama 3 and Mistral. Llama 3 is an open-sourced advanced language model developed by Meta AI and released in April 2024. It builds on the strengths of its predecessors, LLama and LLama2, with improvements in both performance and efficiency. We adapt a pre-trained Llama 3 language model with the size of 8 billion parameters. Mistral is an open-sourced advanced language model developed by Mistral AI and released in September 2023. We adapt a pre-trained Mistral language model with a size of 7 billion parameters.

For the weaker model, we utilize one large language model without instruction tuning as our weak model: Llama 2. It is an open-sourced language model developed by Meta AI and released in February 2023. Given its earlier release date and the use of older technology, its performance is significantly lower compared to the other two models. We adapt a pre-trained Llama 2 language model with a size of 7 billion parameters.

4.3 Details and evaluations

Our experiments are divided into 3 parts. In the first part, we try to investigate the main factors for ICL to impact the performance of LLMs in math problems. We conduct some experiments on GSM8K Dataset. For forms of examples, we sample some examples that differ in *style, length, difficulty*(mentioned in Method) from the training set of GSM8K. For templates and errors, we randomly sample examples from the training dataset as our original examples and then modify these examples with the methods of *swap, wrong calculation* and *wrong inference*. In this part, we contain 4 examples into the prompts and ask three models - Llama2, Llama3 and Mistral - totally 500 questions in test set.

In the second part, we aim to find out LLMs' tolerance for modifications and errors in examples in ICL. Secondly, we modify the templates and make errors in examples and do experiments on Math Dataset. We utilize the method of *pattern only* and *statement only* for templates and *wrong calculation* and *abstract symbol* for errors. We also con-

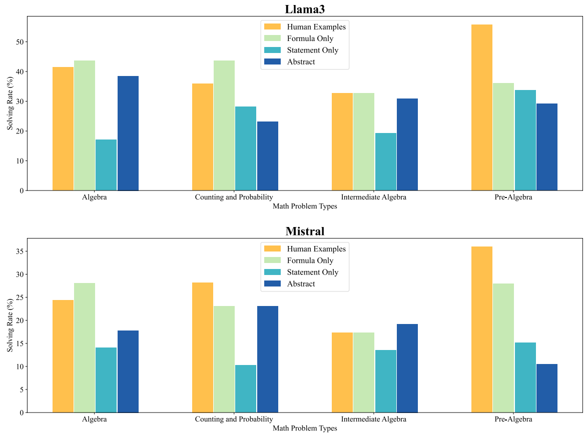


Figure 2: The performance (%) of models with different styles of example solutions on the MATH dataset.

tain 4 examples in the prompts and experiment on Llama3 and Mistral. Considering the performance of these two LLMs, we only choose four types of math problems, which are algebra, counting and probability, intermediate algebra, and pre algebra, with difficulty=1. All the questions in these subsets are asked in our experiments(Detailed numbers are mentioned above in 4.1).

In the first two parts above, we do the experiments as our architecture shown in 1 and extract LLMs' final results from their responses and compare with the correct results. The accuracy is our criteria for measuring LLMs' performance in this method.

In the final part, we do a simple experiment on wrong-to-strong reasoning, which is one of practical application of the tolerance we have found. Llama2 is selected as the smaller model and Llama3, Mistral are the larger models. The dataset is still the four math types in Math. We utilize Llama2 to generate 4 samples for larger models and measure the performance by accuracy.

5 Results

Based on experiments with different examples, we have to some extent identified the tolerance of ICL to errors in examples. Building on this foundation, we have also demonstrated the effectiveness of our tolerance in the subject of "supervising a high-performing model with a low-performing model" through specific experiments.

5.1 Explore possible important aspects

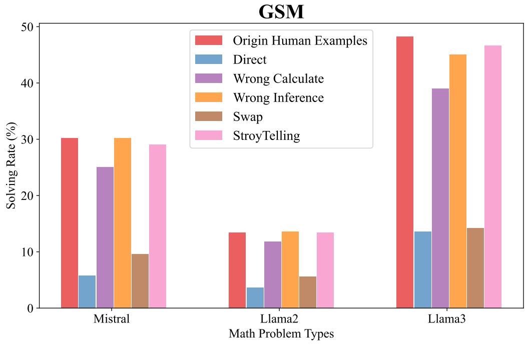


Figure 3: The performance (%) on the GSM8k dataset of three different models on different potential sensitive aspects.

We conduct several experiments to find out the possible aspects to which ICL is sensitive. Our experiment utilizes three models, Llama2, Llama3, and Mistral, and uses the GSM8K dataset to evaluate their performance. We explore two formats: make the examples directly give answers (Direct) and first generate answers and then solutions (Swap), two kinds of errors: calculation errors (Wrong Calculation) and inference errors (Wrong Inference) within solutions, and the kindergarten narrative solution style (StoryTelling). The results are demonstrated in Figure 3.

As Figure 3 shows, the format of examples has a strong impact on the performance of three models. Removing the solutions or making the models first propose answers, then the solutions can reduce the solving accuracy dramatically. Oppositely, whether there are calculation errors or inference errors does not affect the performance much. Also, the narrative style is not important. We are surprised that the errors in examples do not harm its performance; we think that may be because the errors within the examples are not large enough, and the GSM8k dataset is relatively easy. Therefore, we conduct further experiments to find out the error tolerance in the following section.

We also explore the impact of the length of the examples and their difficulties. We manually select three sets of length-related examples by their length and categorize them into long, middle, and short; each set contains four examples. We also select three sets of difficulty-related examples by their difficulty level and categorize them into easy, middle, and hard; each set contains four examples. The results are shown in Table 3 and Table 4.

Table 3 shows that the length effects of in-context learning vary by different models. Mistral and Llama2 benefit from shorter solutions, while Llama3 benefits from longer solutions. We specu-

Table 3: The impact of the length of solutions on the accuracy of three large language models' performance (%) on the GSM8k dataset.

Model	Length		
	Long	Middle	Short
Mistral	21.2	25.8	**26.4**
Llama2	9.6	10.0	**11.2**
Llama3	**44.4**	40.4	38.4

Table 4: The impact of the difficulty level of solutions on the accuracy of three large language models' performance (%) on the GSM8k dataset.

Model	Difficulty Level		
	Easy	Middle	Hard
Mistral	**24.4**	19.0	20.4
Llama2	**15.4**	11.0	11.0
Llama3	**42.4**	38.2	34.2

late that it results from different ways of pertaining, which leads to different preferences of the models.

Table 4 shows that the difficulty levels have a strong and consistent impact on all experimented models. Taking easier questions as examples leads to much better performance compared to more difficult problems. We infer that easier questions are better for understanding the format of the problems.

5.2 Tolerance of example errors.

We conduct experiments on the MATH dataset to find out the tolerance of how the error among the examples affects the results of in-context learning. We get our experimental results in Figure 4.

As Figure 4 shows, Llama3 demonstrates a relatively high tolerance to the error within the examples. In the Algebra, Counting and Probability, and Intermediate Algebra math types, although there are slight fluctuations of performance between different levels of error, the solving ability is relatively stable. For Pre-Algebra math problems, its performance shows a significant decline with the increase in error. Also, changing the signs in the examples has a strong negative impact on Pre-Algebra math problems.

The Mistral model shows a significant decline with more error among examples in the Algebra and Pre-Algebra datasets. In most math types (except Algebra), the performance drops relatively high when the signs in examples change. Most math types undergo a greater performance drop after more than two digits are changed.

To summarize the two models, we find that most of the time, changing the signs in training examples can lead to a large performance drop, possibly because it leads to an inconsistent inference. Llama3 has quite a high tolerance to error, while Mistral is more sensitive to it. For more sensitive models and question types, if the error in the examples is more than two digits, they have a high possibility to harm the performance of the model significantly. It seems that in most cases, the models can tolerate the error within two digits.

5.3 The impact of solution template.

Besides finding the error tolerance among different models, we also explore how the ways of providing solutions affect the performance of in-context learning. We experience three different ways: only give the natural language description of solutions (Statement Only), only give the formula derivation (Formula Only), and transform every number in solutions and answers into letters. The results are demonstrated in Figure 2.

As Figure 2 shows, large language models are relatively insensitive, with only given formulas as solutions or changing the numbers into letters. However, their performance decline among all math types when only given natural language descriptions of the solutions, which indicates that the existence of formulas is of significant importance for in-context learning math problem examples.

5.4 Weak models to stronger models

In the above experiments, we find that the error tolerance of large language models is quite high unless the error is too extreme and counterintuitive. It can perform great just as a human-written example with a few errors. Therefore, we conduct an experiment using a relatively weak model, Llama2, to generate solutions and answers as in-context learning examples for stronger models, Llama3 and Mistral. The results are shown in Figure 5.

We can see that although Llama2 is weaker than the other two models and the answers it generated usually contain errors, it can serve as a good example generator since the generated data are within the tolerance. In some cases, the examples it generated lead to a higher performance than human-written examples. Therefore, it indicates that rather than using humans to make examples for large language models, we can utilize the relatively smaller and

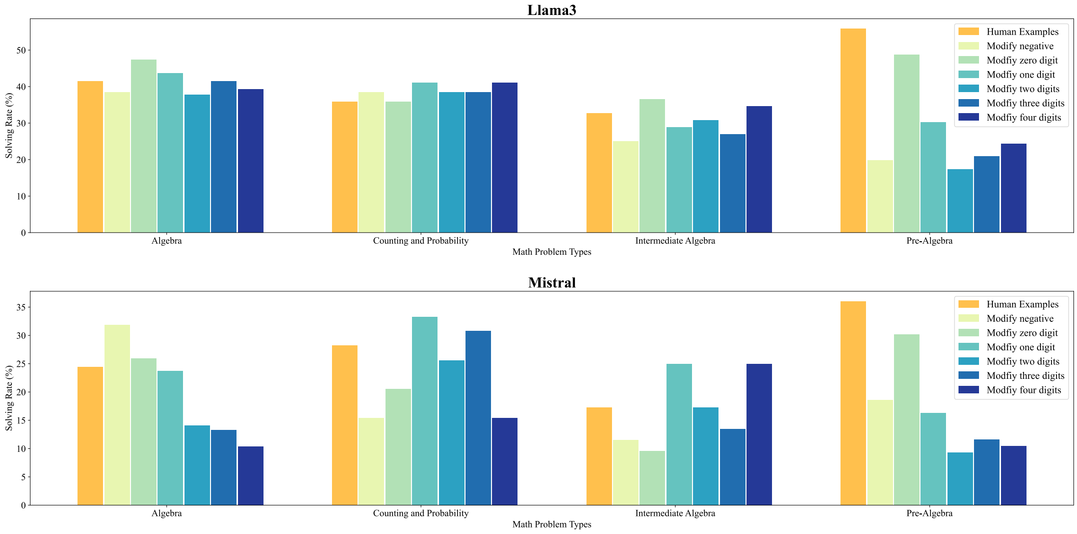


Figure 4: The performance of models (%) on the MATH dataset with different extents of errors in examples, which indicates the tolerance of large language models

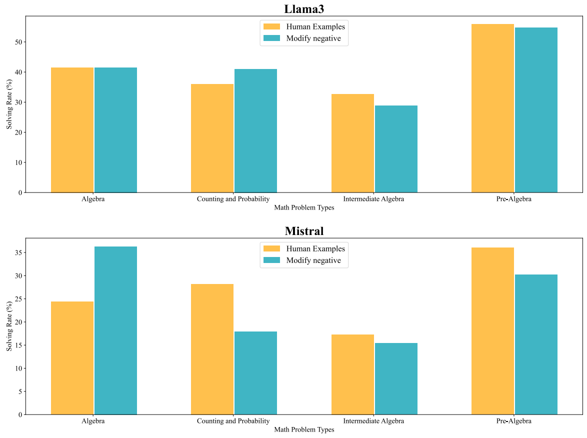


Figure 5: The performance (%) of stronger models using examples generated by the weak model compared to human-written examples. The experiments are conducted on the MATH dataset. The stronger models are Llama3 and Mistral and the weak model is Llama2.

weaker models to synthesize data, which is much cheaper and more efficient.

6 Conclusion

We conduct several experiments to explore the impact of in-context learning examples' forms, formats, and errors. We use Llama2, Llama3, and Mistral as our models and utilize the GSM8k and MATH datasets to evaluate their performance. We find that changing formats significantly affects performance while changing forms is relatively low. Among all format styles, the appearance of formulas seems to be an important attribute of the good performance of math problems in context learning. The length of examples does affect the performance, but it varies among different models, while easier examples lead to better performance consistently.

For the tolerance of errors in the examples, we find that slightly changing the digits of numbers or containing some inference errors does not significantly affect the results. However, if the digit changes are more than two, the performance has a higher possibility of dropping greatly, indicating that two digits can be the tolerance of most questions and models.

Since the tolerance of the large language models is quite high, we attempt to make weaker models to generate example solutions and answers for stronger models. We find that the generated data are within the tolerance. The experiment result shows that the data generated by the weak models can achieve compatible performance as human-written examples, indicating this approach is promising and efficient.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan

Leike, Ilya Sutskever, and Jeff Wu. 2023. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *Preprint*, arXiv:2312.09390.

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. *arXiv preprint arXiv:2110.14168*.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. *Preprint*, arXiv:2210.00720.

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. *NeurIPS*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *Preprint*, arXiv:2104.08786.

Aman Madaan, Katherine Hermann, and Amir Yazdanbakhsh. 2023. What makes chain-of-thought prompting effective? a counterfactual study. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1448–1535, Singapore. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *Preprint*, arXiv:2202.12837.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5368–5393, Toronto, Canada. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. 2023. Large language models are in-context semantic reasoners rather than symbolic reasoners. *Preprint*, arXiv:2305.14825.

Hugo Touvron, Louis Martin, and et al. Kevin Stone. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.