AI-Assisted Generation of Difficult Math Questions

Vedant Shah [1,2] **Dingli Yu** [3] **Kaifeng Lyu** [3] **Simon Park** [3] **Nan Rosemary Ke** [1] **Michael Mozer** [4]
Yoshua Bengio [1,2] **Sanjeev Arora** [3] **Anirudh Goyal** [1]

Abstract

Current LLM training positions mathematical reasoning as a core capability. With publicly available sources fully tapped, there is unmet demand for diverse and challenging mathematics questions. Relying solely on human experts is both time-consuming and costly, while LLM-generated questions often lack the requisite diversity and difficulty. We present a design framework that combines the strengths of LLMs with a human-in-the-loop approach to generate a diverse array of challenging math questions. Initially, leveraging LLM metacognition skills (Didolkar et al., 2024), a strong LLM is used to extract core "skills" from existing math datasets. These skills serve as the basis for generating novel and difficult questions by prompting the LLM with random pairs of core skills that must be utilized in the question. This "out of distribution" task is challenging for both LLMs and humans. Our pipeline employs LLMs to iteratively generate and refine questions and solutions through multi-turn prompting. Human annotators then verify and further refine the questions, with their efficiency enhanced through further LLM interactions. Applying this pipeline on skills extracted from MATH dataset (Hendrycks et al., 2021) resulted in a dataset of complex math questions, while improving expert productivity. Despite using skills from the MATH dataset, our approach of combining random skill pairs in questions resulted in noticeably higher quality, as evidenced by: (a) Lower performance of all models on our questions than on MATH (with open models being the most affected). (b) Higher performance on MATH when using our questions as in-context examples. Although focused on mathematics, our methodology seems applicable to other domains requiring structured reasoning. It can be seen as a method for *scalable oversight,* where human experts evaluate highly capable AI models by also using AI-assistance.

1. Introduction

Significant improvement in the capabilities of LLMs (Chowdhery et al., 2023; Anil et al., 2023; Team, 2023; Team et al., 2023; Abdin et al., 2024; Achiam et al., 2023; Touvron et al., 2023) to understand and generate complex mathematical content has been achieved by leveraging all the public data and a fair bit of private data. Sources of high-quality, varied, and difficult mathematical questions are drying up. Even LLM evaluation has become tricky since even newly-released human exams are somewhat similar to past exams, which are potentially present in the LLMs' training datasets. Hence, there is a pressing need for innovative methods to create new, diverse, and challenging questions.

Expert mathematicians and educators possess the deep understanding required to create questions that not only test a wide range of skills but also push the boundaries of what the learners, and by extension the models, can handle. However, relying solely on human experts is not scalable. Generating synthetic questions using LLMs is feasible at scale (Li et al., 2024; Gunasekar et al., 2023; Patel et al., 2024; Toshniwal et al., 2024; Gupta et al., 2023; Lu et al., 2024; Honovich et al., 2022), but often fall short in terms of the necessary diversity and difficulty. This dichotomy between the quality of human-generated questions and the scalability of LLM-generated questions presents a significant challenge (Yu et al., 2024).

1.1. Evaluation Saturation Phenomenon

LLM evaluations getting saturated is a well-known issue. Some of the saturation is driven by across-the-board improvements arising from better training and more extensive/better datasets. But a lot has to do with evaluation-specific enhancements that optimize model performance on standard evaluations through techniques like supervised fine-tuning (SFT) on synthetic question-answer pairs. These

[*]Equal contribution [1]Mila – Quebec AI Institute [2]Université de Montréal [3]Princeton University [4]University of Colorado, Boulder. Correspondence to: Vedant Shah <vedantshah2012@gmail.com>, Anirudh Goyal <anirudhgoyal9119@gmail.com>.

The first AI for MATH Workshop at the 41st International Conference on Machine Learning, Vienna, Austria.

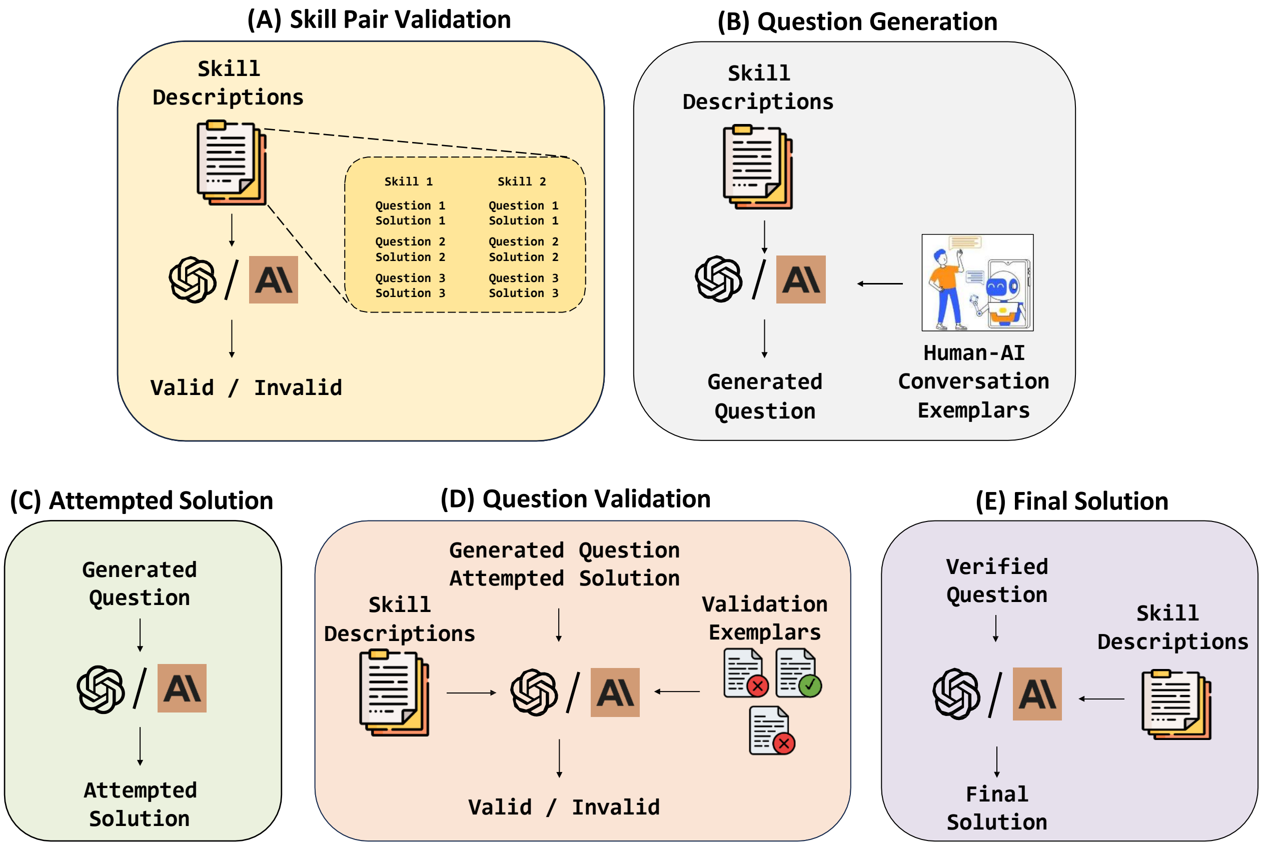


Figure 1. **AI assisted question generation**: This figure illustrates a five-step AI-assisted pipeline for generating high-quality questions. **(a) Skill Pair Validation** - The model first validates that the given pair of skills are distinct and not too similar. **(b) Question Generation** - Using the validated skill pair, the model generates a question that necessitates the application of both skills for its solution. **(c) Attempted Solution** - Given the generated question, the model is asked to attempt a solution to the question while taking a *defeatist* approach. **(d) Question Validation** - The model validates the generated question based on the attempted solution, checking for correctness, skill rigor, clarity, and other quality criteria. **(e) Final Solution** - Valid questions are re-solved by the model using advanced techniques like in-context prompting and majority voting to enhance the accuracy of the final solution.

synthetic pairs can be generated by leading proprietary models when provided with a few examples from the dataset or by filtering the model's own responses (Yue et al., 2023; Yu et al., 2023b). Such methods can dramatically boost performance; for example, just 1 million synthetic examples can elevate Llama2 7B's performance on the MATH dataset to levels comparable to GPT-4 (Li et al., 2024).

The distinction between general and evaluation-specific improvements is crucial. The latter may lead to overfitting to particular evaluations rather than a genuine acquisition of mathematical skills. This issue was highlighted when a new version of the GSM8K dataset revealed performance drops in many models, indicating overfitting to the previous dataset version (Zhang et al., 2024). Similarly, leading LLMs performed significantly worse on newer versions of the Chinese GaoKao exam compared to older exams, raising fundamental questions about the depth of their mathematical understanding.

1.2. Proposed Framework: AI-assisted Generation of Difficult Math Questions

At first glance, it may seem counterintuitive to use an AI model to generate and correct novel questions that it is itself unable to solve. However, recent research (Arora & Goyal, 2023; Didolkar et al., 2024) demonstrated that top LLMs possess a robust understanding of mathematical skills, including capability to identify the skills required to solve given questions (Reid et al., 2024; Achiam et al., 2023). This naturally raises the question: *can LLMs operate in the reverse direction, i.e., generate math problems when given a list of skills that have to be tested?* Our initial attempts yielded mixed results. While leading models could produce creative math questions when provided with a list of skills, the majority of these questions exhibited one or more of the following shortcomings: too similar to existing questions in datasets; have errors or nonsensical elements; are too tedious or mechanical to be engaging for human annotators. (See Section 4.) Moreover, they often conflate "difficulty" with tedious calculations, which actually would play to the strength of machines to leverage external tools such as calculators or Python interpreters.

Nevertheless, there were promising instances where LLMs generated interesting and correct questions which they were unable to solve, due to incomplete or incorrect reasoning. This observation led us to the concept of *AI-assisted creation of evaluation datasets*. Our process may also be of interest for human pedagogy, since it begins with the extraction of core "skills" from existing math datasets, which serve as the foundational elements of mathematical questions. The current paper focuses on the MATH dataset (Hendrycks et al., 2021), a mainstay of LLM evaluation in recent years.

Starting with a list of mathematical skills extracted from the MATH dataset using recently discovered methods (Didolkar et al., 2024), we focused on creating questions that involve one skill from pre-algebra and algebra portions of MATH dataset (this is the portion on which leading LLMs are approaching 100% accuracy) and one other skill randomly sampled from different sections of MATH. Introducing an element of randomness via such combinations ends up enhancing both the diversity and the difficulty of the questions. It requires "out of distributon" thinking that challenged even human raters. Our generation pipeline used carefully crafted prompts and multi-turn interactions with leading models to significantly improve the generation of high-quality questions and candidate answers.

In our AI-assisted process, human experts played a crucial role. Using the (question, answer) pairs generated by LLMs and leveraging API access to leading models, experts identified promising questions—-often those incorrectly answered by the LLMs but containing many correct ideas. Experts then refined these questions to enhance their engagement value and provided gold-standard answers. The AI-assisted process not only boosted human productivity but also resulted in high-quality, novel questions distinct from those in existing datasets. Many of these questions were not just new but also represented new types, combining different skills in innovative ways, such as linking area-and-perimeter calculations with prime number knowledge. We believe this methodology and the resulting dataset could introduce fresh perspectives into math instruction for both AI and human learners.

Connection to Scalable Oversight: This notion (Bowman & etal, 2022) looks ahead to how humans might supervise and check AI systems that potentially outperform humans in many relevant skills. While typically discussed in the context of alignment and safety, the concept is pertinent here. How can human experts reliably evaluate LLMs' understanding of high-school or freshman-level math when these models have already been trained on all available exams and textbooks? Could human-AI collaboration lead to more novel evaluations?

Paper organization and main results: Sections 2 describes our design methodology and pipeline used for generating difficult math questions. Our process yields a dataset of 180 questions. These questions use at least two skills, one of which is extracted from the pre-algebra and algebra portions of MATH (Hendrycks et al., 2021), which are known to be easiest. Section 3 discusses the performance of various open source and proprietary models when evaluated on the generated dataset versus when evaluated on MATH (Hendrycks et al., 2021). It also discusses the usefulness of the generated questions as in-context exemplars. Section 4 discusses some interesting behaviors and failure modes of the models observed during the question generation process.

Figure 2 gives a plot of the scores of how various models do on MATH vs our evaluation. Larger deviations correspond to lack of proper understanding of the relevant math skills, in the sense that the model struggles when a single question involves multiple skills. Table 1 provides the exact numbers plotted in Figure 2.

2. Pipeline for AI Assisted Question Generation

We present a structured approach to generating challenging mathematics questions by combining the capabilities of large language models (LLMs) and human expertise. Given below is a high-level overview of the process before delving into the details of each step.

We begin our pipeline with **skill extraction** - identifying and cataloging distinct mathematical skills from a dataset, as described in (Didolkar et al., 2024). This step creates a repository of skills linked to specific questions. The motivation behind this is to systematically generate and analyze questions that require specific skills, ensuring a comprehensive evaluation framework.

Next, we focus on **generating questions that combine pairs of distinct skills** to increase their difficulty. By using advanced models like GPT-4 and Claude, and incorporating in-context examples of multi-way interactions between AI and humans, we enhance the models' performance in generating complex questions. This step aims to produce challenging questions that robustly assess problem-solving abilities.

The final step involves **screening and validation** to filter out

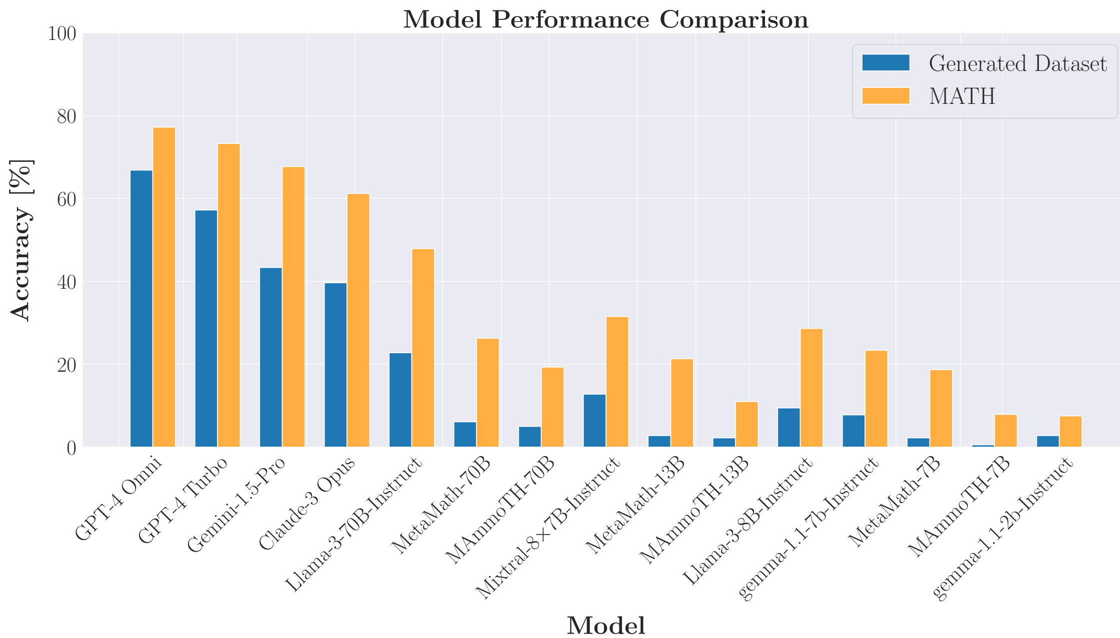


Figure 2. **Comparison of Zero-Shot Performance of Various Models on MATH and Our Generated Dataset** - This figure illustrates the zero-shot Chain of Thought (CoT) performance of both open-source and proprietary models on two different datasets: MATH and our generated dataset. Across the board, models demonstrate a lower performance on the generated dataset compared to MATH. Proprietary models exhibit the smallest decrease in performance, while smaller models within the same family experience more significant performance drops. Detailed numerical values related to this comparison are available in Table 1.

invalid or flawed questions. This rigorous process includes evaluating and solving the questions to identify hidden flaws, such as computational intractability or logical inconsistencies. Advanced techniques like in-context exemplars and self-consistency further ensure the accuracy and quality of the solutions. This step is crucial for maintaining the integrity and reliability of the generated questions and their solutions. Overall, each step in the pipeline is designed to systematically enhance the quality and difficulty of questions, providing a robust and comprehensive evaluation of mathematical skills.

Motivated by these challenges, we employ a five-step approach to generate difficult math questions using advanced models. For each round of generation, we randomly sample a pair of skills and three sample question-solution pairs corresponding to each skill from the skill repository. These reference examples are sourced from the MATH dataset.

Step 1: Skill Pair Validation. We begin by asking the LLM (GPT-4 or Claude) to validate the skill pair by assessing the qualitative similarity of the two skills. Reference examples are provided in-context to enrich the model's understanding of the skills. If the model deems the skills too similar, they are flagged and excluded from question generation, as similar skills might lead to simpler questions.

Step 2: Question Generation. Next, we prompt the LLM to generate a question and a brief solution requiring the application of both skills in the sampled pair. We specify two conditions to ensure high-quality questions: the question should either require an exact answer or specify that an approximate answer is acceptable, and it should ask for only a single final result. In-context, we provide two multi-turn conversations between a human and an AI assistant. These conversations demonstrate the human providing feedback on the AI-generated questions, which the AI then refines. This helps the model anticipate and avoid practical issues, such as insufficient involvement of skills or logical inconsistencies.

Step 3: Solution Attempt. The model then attempts a solution to the generated question, adopting an adversarial approach to identify flaws such as insufficient information, ambiguity, self-contradiction, or excessive computation. If any issues are found, the model stops solving and clearly states the problems. Otherwise, it completes the solution. During this step, the model does not receive the skill names or reference examples to ensure unbiased problem-solving.

Step 4: Question Validation. We give LLM the generated question and its solution for validation against a fixed rubric consisting of seven criteria:

- Single Answer Requirement: The question should ask for only one final answer.
- Exact Answer Requirement: There should be only

one exact answer, unless approximations are explicitly stated.

- Dual Answer Requirement: The question must necessarily and sufficiently involve the application of both skills, with difficulty comparable to or greater than the reference examples.
- Clarity and Completeness: The question should be clear and contain all necessary information.
- Computational Tractability: The question should not require overly complex computations.
- Realism and Logic: The scenario should be realistic and logically consistent.
- Syntax and Grammar: The question should be grammatically correct and clearly written.

The model uses reference examples and validation exemplars - model generated examples of validating questions, to facilitate this step. We employ majority voting (maj @ 4) to enhance robustness.

Step 5: Final Solution and Re-validation. For questions classified as valid, we ask the LLM to re-solve the question to obtain a final solution. Reference examples are provided in-context to improve the model's understanding. We use majority voting (maj @ 4) to ensure consistency. If all the answers obtained in this step are unique, indicating potential ambiguity, the question is discarded.

The questions obtained from the above pipeline are further screened by humans. This structured approach not only generates challenging and novel math questions but also ensures their quality through rigorous validation, effectively combining the strengths of AI and human oversight. For detailed examples of prompts used at each step, refer to Appendix A.5.

3. Experiments and Findings

Through our experiments, we demonstrate the difficulty and quality of the generated set of questions while also analysing the behavior of different models on this task of *compositional generalization*. Firstly, evaluate a wide range of models spanning a large range of parameter counts on the generated dataset and compare against their performance on MATH (Hendrycks et al., 2021) which is the base dataset used for extracting skills, showing that the generated set of questions is necessarily harder than MATH. Next, we further demonstrate the difficulty and quality of these questions by showing that they are better in-context exemplars as compared to standardly used exemplars. We describe the experimental setup below.

3.1. Experimental Setup

We follow the pipeline proposed in (Didolkar et al., 2024) to extract skills from the MATH dataset (Hendrycks et al., 2021). The MATH dataset encompasses seven high-level topics, allowing us to identify and extract finer-grained skills within each topic and label each question accordingly. At the end of the skill-extraction process, we identify a set of 114 skills. We then remove a few simple skills, such as `basic_arithmetic` and `arithmetic_operations`, before using the remaining set to generate questions using the proposed approach. We generate and verify 180 difficult questions to create the new dataset.

Table 1. **Comparison of Zero-Shot CoT Performance (Accuracy) on the Generated Dataset vs. MATH Test Set**: This table compares the zero-shot Chain of Thought (CoT) performance accuracy of various models on our generated dataset and the MATH test set. The generated questions consistently present a higher level of difficulty across all evaluated models. Proprietary models exhibit the smallest, yet still significant, performance drops relative to MATH, while open-source models experience the largest declines. GPT-4 Omni demonstrates the least relative drop in performance at 13.42%, whereas MAmmoTH-7B shows the highest relative drop at 92.91%. Additionally, smaller models within the same family tend to show larger relative performance decreases. Models specifically designed for MATH tasks display the greatest performance degradation when compared to their performance on the MATH test set.

Model	Our Dataset	MATH
GPT-4 Omni	66.85%	77.21%
GPT-4 Turbo	57.22%	73.27%
Gemini-1.5-Pro	43.34%	67.70%
Claude 3 Opus	39.66%	61.20 %
Llama-3-70B-Instruct	22.77%	47.89%
MetaMath-70B	6.11%	26.27%
MAmmoTH-70B	5.00%	19.31%
Mixtral-8×7B-Instruct	12.78%	31.52%
MetaMath-13B	2.79%	21.32%
MAmmoTH-13B	2.23%	10.99%
Llama-3-8B-Instruct	9.45%	28.62%
Gemma-1.1-7B-Instruct	7.78%	23.36%
MetaMath-7B	2.23%	18.69%
MAmmoTH-7B	0.56%	7.90%
Gemma-1.1-2B-Instruct	2.78%	7.52%

We evaluate the generated set of questions on a variety of language models, both small and large. Specifically, we assess the MetaMath (Yu et al., 2023b), MAmmoTH (Yue et al., 2023), Gemmma (Team et al., 2024), and Llama-3 series, as well as one Mixture-of-Experts model Mixtral-8×7B-Instruct. Additionally, we include evaluations of larger proprietary models such as GPT-4o, GPT-4 Turbo[1] (OpenAI, 2023), and Claude 3 Opus[2]. We compare the performances of these models on our generated questions against their performance on the MATH dataset (Hendrycks et al., 2021).

For generating responses, we use the MAmmoTH (Yue et al., 2023) evaluation suite. The responses are graded using a GPT-4 grader, where GPT-4 Omni checks the correctness of a solution response against the ground truth solution. During response generation, we set the temperature to 0 and top_p to 1 for all models. All necessary compute details are discussed in Appendix A.4

Generated Dataset is More Difficult than MATH. Table 1 compares the performance of various models, spanning a range of parameter counts, on the generated dataset versus the MATH dataset. We observe a performance drop across all models on the generated dataset. Even frontier AI models experience significant declines. Specifically, GPT-4 Omni, GPT-4 Turbo, Gemini-1.5-Pro and Claude 3 Opus see relative drop (in terms of percentage of original success rate) of 13.42%, 21.90%, 35.98% and 35.19% respectively. The largest drop is **92.91%** for MAmmoTH-7B. This decrease in performance is likely due to the fact that while MATH questions typically require the application of a single major skill, our questions were generated to require two skills. Among smaller models, the Gemma (Team et al., 2024) series demonstrates better resilience in their size class.

Higher performance degradation on smaller models. Figure 3(b) shows that smaller models generally experience larger relative performance declines compared to larger models in the same family, with the only exception of the Gemma family of models where the 7B parameter shows a 4% larger degradation as compared to the 2B parameter model. This suggests that larger models are better at composing skills or concepts (a phenomenon known as "compositional generalization") than smaller models. This finding aligns with conclusions drawn in (Arora & Goyal, 2023; Yu et al., 2023a).

Specialist Models Show Worse Drops. Math specialist models, such as MetaMath and MAmmoTH, exhibit larger performance drops compared to general-purpose models of similar sizes: drops of 88.07% and 92.91%, respectively for MetaMath-7B and MAmmoTH-7B versus 66.98% for LLama3 8B. Specialist models are extensively trained on synthetic datasets, such as MetaMathQA (Yu et al., 2023b) and MathInstruct (Yue et al., 2023), which might lead to overfitting to the dataset, whereas our dataset is “out of distribution.”

Generated Questions as Effective In-Context Examples for MATH. A possible test for the quality of a Q&A pair on similar topics as MATH dataset is whether performance on MATH improves when using these as in-context exemplar.

We test as follows. Recall that MATH has 7 sections. Exemplars for a section are chosen from the section area. However, by design, our new questions cross section boundaries. We implemented a new procedure to retrieve in-context exemplars based on the skill requirements of the current question.

Since the generated dataset is limited in size, it does not cover all the skills extracted during the skill extraction process, containing 97 out of 114 skills. Figure 3(a) shows the distribution of different skills in the dataset. We filtered the MATH test set to remove examples requiring skills not present in the generated dataset, resulting in the removal of 809 test examples. During evaluation on the filtered MATH test set, for each question Q labeled with skill a ($a \in \mathcal{S}$, where $\mathcal{S}$ is the set of extracted skills), we retrieved in-context exemplars from the generated dataset, ensuring each exemplar involved skill a. We used four such exemplars per question (i.e., 4-shot CoT (Wei et al., 2022)). For skills represented by fewer than four examples in the dataset, we supplemented the remaining exemplars from the MAmmoTH evaluation suite. We compared the performance of models using this targeted prompting strategy against their performance with vanilla 4-shot CoT on MATH, where all in-context exemplars came from the MAmmoTH evaluation suite.

Table 2 presents the results of this comparison. The proposed prompting strategy clearly outperforms vanilla few-shot prompting. We conclude that the generated questions, due to their difficulty and skill relevance, serve as effective in-context exemplars. Performance gains would likely be more significant with larger datasets generated using our approach, reducing the need to supplement with external exemplars.

4. Observations from the Question Generation Process

The question generation pipeline described in Section 2 was developed through an iterative process of refining prompts and design choices, and evaluating their impact on the quality of the final questions and solutions. Notably, the inclusion of the *attempted solution* and *question validation* steps significantly enhanced the pipeline's effectiveness. Despite the sophistication of the pipeline and prompts, we still observe instances where models fail to follow the given instructions. This section highlights prominent failure modes at various stages of the pipeline, which human raters need to be aware of. Additionally, we explore some intriguing behaviors of the models where they successfully create interesting and creative questions. Section A.2 details the role

[1]Points to `gpt-4-turbo-2024-04-09`

[2]Points to `claude-3-opus-20240229`

Table 2. Performance of models on MATH under two different prompting strategies. **Vanilla 4-shot CoT** prompting involves 4-shot prompting with exemplars taken from the MAmmoTH (Yue et al., 2023) evaluation suite. **Proposed 4-shot CoT** prompting consists of 4-shot prompting with exemplars taken from the generated set of questions. These exemplars are retrieved such that one of the two skills in each exemplar is the same as the skill required by the question at hand, as labelled by GPT-4. We show that few-shot prompting with exemplars retrieved from the generated set of questions consistently outperforms vanilla few-shot prompting with gains of upto 3.75% (for Mixtral-8×7B-Instruct (Jiang et al., 2024).

Method	GPT-4 Omni	GPT-4 Turbo	Llama-3-70B-Instruct	MetaMath-70B	MAmmoTH-70B	Mixtral-8×7B-Instruct
Vanilla 4-shot CoT	76.67%	71.89%	49.11%	25.77%	18.45%	30.77%
Proposed 4-shot CoT	**78.39%**	**74.19%**	**51.13%**	**26.60%**	**20.06%**	**34.52%**

of human raters in improving these questions.

4.1. Creative questions: Examples of Synergy from Human-AI interaction

The models frequently produced interesting and creative questions, although they often generated incorrect solutions. In these cases, the incorrect solutions usually contained enough correct ideas for a human to quickly complete them.

Human annotators were tasked with verifying the validity of the questions and the correctness of the solutions. They were instructed to look out for any failure modes discussed in Section 4.2. Their responsibilities included ensuring that the created questions actually employed the intended math skills, and improving the questions in terms of readability, quality, and difficulty when possible. They were encouraged to suggest changes that would make the problems harder for automated tools to solve while allowing easier or more elegant solutions for humans. The following examples illustrate this process:

`Original Question: Find the smallest positive integer` k `such that` $k^3 - k$ `is divisible by both 9 and 10, and the sum of digits of` k `in its decimal representation is a prime number.`

Our human team had not encountered such questions before. It requires recognizing that $k^3 - k = k(k-1)(k+1)$ is always divisible by 2 and 3. Thus, k must be such that $k(k-1)(k+1)/6$ is divisible by 15 (both 3 and 5). Additionally, the sum of the digits of k must be a prime number, a task challenging even for powerful LLMs.

`Original Question: Consider a collection of red, blue, and green beads arranged in an infinite series. The beads alternate in color, starting with red, then blue, then green, and this pattern repeats indefinitely. The number of beads in each colored section follows the pattern of powers of 2: the first red section has 2 beads, the first blue section has 4 beads, the first green section has 8 beads, the second red section has 16 beads, and so on. If a bracelet is made using a continuous, unbroken sequence of exactly 20 beads from this series, and each bead has a length of 0.5 units, how many different bracelets can be made such that the perimeter of the bracelet is an integer value?`

The original question combined elements in a novel way. The human rater modified the question to change the sequence size from 20 to 6 beads, maintaining the essential difficulty while making it more elegant for humans. All tested models failed on the modified question.

`Original Question: A container initially contains 500 mL of water. A scientist adds water to the container` $\frac{1}{4}$ `of the current amount every minute. After how many minutes will the container first contain more than 1 L but less than 2 L of water?`

`Modified Question: A container starts with 500 mL of water. Each minute, the scientist adds water equal to` $1/2$ `of the current amount. What is the smallest positive integer` n `such that the number of liters of water in the container is never in the interval` $[n, n+1]$`?`

This was one of many questions the models created about exponential growth and geometric series, possibly similar to standard math test questions. The human slightly altered it to simplify calculations by hand and substituted a different condition that the models found challenging, while humans could easily estimate an approximate answer and then verify.

`Original Question: Consider the sequence defined recursively by` $a_1 = 1$ `and` $a_{n+1} = 2a_n + n$ `for all` $n \geq 1$`. What is the product of the first five terms of this sequence?`

`Modified Question: A sequence` a_n `is defined as follows:` $a_1 = 2$ `and` $a_n = 2^{n-1} + a_{n-1} + n$`. What is the` $\lfloor \log_2 a_{500} \rfloor$`?`

An LLM can solve the original question through simple computation. The modified question, however, requires understanding an underlying pattern.

`Original Question: Find the sum of the smallest prime divisor and the largest prime divisor of the number` $N = 15^4 + 16^4$.

`Modified Question: Find the sum of the two smallest prime divisors of` $23^{17} + 17^{17}$.

Models tend to adopt a brute-force approach to the original question by calculating $15^4 + 16^4$. After rephrasing, the number $23^{17} + 17^{17}$ is too large for direct computation, requiring understanding of arithmetic modulo a prime.

These examples highlight the essential role of human oversight in refining and improving the questions generated by LLMs, ensuring they are challenging, creative, and suitable for advanced mathematical problem-solving.

4.2. Failure Modes

Despite the sophistication of our pipeline, models frequently exhibit several failure modes: (a) *Insufficient Involvement of Skills*: Models often generate questions that either miss one of the skills completely or require a very shallow application of one or both skills. For example, a geometry question may fail to involve ratio and proportion adequately, (b) *Insufficient Information*: Questions may lack essential details needed for solving, making them incomplete or ambiguous. For instance, a trigonometry question might omit necessary angles or distances, (c) *Unsolvable or Computationally Intractable Questions*: Some questions generated are either unsolvable or require excessive brute-force calculations, which are impractical for evaluating reasoning abilities, (d) *Nonsensical Questions*: Models sometimes produce questions that are logically inconsistent, confusing, or ambiguous, such as a probability problem with unclear parameters or an impossible geometry scenario, (e) *Deceitful Solutions*: Occasionally, models fabricate solutions to nonsensical or unsolvable questions, presenting incorrect logic as plausible reasoning and (f) *Finding a Needle in the Haystack*: Long and complex validation prompts sometimes cause models to confuse or overlook the specified skills, leading to incorrect evaluations. For a more detailed discussion and examples of questions in the various categories listed above, refer to Appendix A.1.

5. Conclusions: Limitations and Future Work

Summary. In this work, we introduced a framework that leverages the complementary strengths of humans and AI to generate new, challenging mathematics questions. Building on recent insights into LLM metaknowledge, we identified and named key skills necessary for solving math problems. Using these insights, we developed a pipeline that employs named skills from the well-known MATH dataset, facilitating multi-turn interactions with advanced models to generate questions that combine pairs of skills. These questions were subsequently reviewed and refined by human raters. The proposed pipeline produced questions with greater novelty and difficulty compared to those in the original MATH.

This framework also resulted in a new math evaluation that assesses the same skills as the MATH dataset but is significantly more challenging for leading models. Notably, GPT-4-T and GPT-4-O exhibited the smallest performance drops, which aligns with the subjective evaluations of our human raters. Additionally, we demonstrated that providing the newly generated questions as in-context examples improved GPT-4-O's performance on the MATH dataset more effectively than examples sourced directly from the MATH dataset. This finding further validates the quality of the questions produced by the proposed pipeline.

Our evaluation revealed that open models performed disappointingly on these new questions, highlighting the need for improved training methods. However, their weak performance also indicates that significant advancements can be made with novel questions of only moderate difficulty—questions that the proposed pipeline can generate at scale. We plan to release detailed information about the proposed pipeline to encourage further research and development in the field of open-source math models.

Limitations and Future Work. The proposed pipeline enhances the efficiency of generating challenging math evaluation benchmarks by leveraging advanced proprietary models, but it incurs high costs due to extensive use of these models and significant human verification. To improve efficiency, future work should focus on optimizing prompting strategies to produce higher-quality questions initially, thereby reducing the need for extensive filtering. Additionally, reducing human verification through the development of automated validation tools is crucial. Integrating a feedback loop, where the model learns from the questions that pass human verification, can further streamline the process by progressively improving question quality. Exploring cost-effective model alternatives and enhancing pipeline automation are also essential steps. These measures will reduce dependency on expensive proprietary models, lower overall operational costs, and maintain or even enhance the quality of the generated math evaluation benchmarks.

Looking ahead, an even more exciting prospect is the potential application of the proposed framework to efficiently produce high-quality data in domains beyond mathematics. We believe that this approach can substantially contribute to the advancement of AI capabilities across various fields and enhancing model performance through robust and challenging evaluations.

6. Acknowledgements

This research was enabled in part by compute resources provided by Mila (mila.quebec) and by resources provided by the Princeton Language and Intelligence (PLI). VS would like to thank Aniket Didolkar for helpful discussions throughout the project and for proof reading the paper. AG would like to thank Melvin Johnson, James McClelland and Yoram Bachrach for helpful discussions and useful feedback. AG would also like to thank Daan Wierstra, Melvin Johnson, Siamak Shakeri, Murray Shanahan, John Quan, Theophane Weber, Olivier Tieleman, David Silver, Charles Blundell, Behnam Neyshabur, Ethan Dyer and Nicolas Heess for support and guidance.

References

Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Arora, S. and Goyal, A. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*, 2023.

Bowman, S. and etal. Measuring progress on scalable oversight for large language models, 2022.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023. URL http://jmlr.org/papers/v24/22-1144.html.

Didolkar, A., Goyal, A., Ke, N. R., Guo, S., Valko, M., Lillicrap, T., Rezende, D., Bengio, Y., Mozer, M. C., and Arora, S. Metacognitive Capabilities of LLMs: An Exploration in Mathematical Problem solving. 2024. URL https://api.semanticscholar.org/CorpusID:269921384.

Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C. C. T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

Gupta, H., Scaria, K., Anantheswaran, U., Verma, S., Parmar, M., Sawant, S. A., Mishra, S., and Baral, C. Targen: Targeted data generation with large language models. *arXiv preprint arXiv:2310.17876*, 2023.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset, 2021.

Honovich, O., Scialom, T., Levy, O., and Schick, T. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*, 2022.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of experts, 2024.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Li, C., Wang, W., Hu, J., Wei, Y., Zheng, N., Hu, H., Zhang, Z., and Peng, H. Common 7b language models already possess strong math capabilities. *arXiv preprint arXiv:2403.04706*, 2024.

Lu, Z., Zhou, A., Ren, H., Wang, K., Shi, W., Pan, J., Zhan, M., and Li, H. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. *arXiv preprint arXiv:2402.16352*, 2024.

OpenAI. Gpt-4 technical report, 2023.

Patel, A., Raffel, C., and Callison-Burch, C. Datadreamer: A tool for synthetic data generation and reproducible llm workflows. *arXiv preprint arXiv:2402.10379*, 2024.

Reid, M., Savinov, N., Teplyashin, D., Lepikhin, D., Lillicrap, T., Alayrac, J.-b., Soricut, R., Lazaridou, A., Firat, O., Schrittwieser, J., et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Team, G. Gemini: A family of highly capable multimodal models, 2023.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Team, G., Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M. S., Love, J., et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Toshniwal, S., Moshkov, I., Narenthiran, S., Gitman, D., Jia, F., and Gitman, I. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *arXiv preprint arXiv:2402.10176*, 2024.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Wei, J., Wang, X., Liu, Q., Yang, B., Dong, X., Huang, H., and Wang, W. Chain-of-thought prompting elicits reasoning in large language models. *arXiv*, abs/2201.11903, 2022. URL https://doi.org/10.48550/arXiv.2201.11903.

Yu, D., Kaur, S., Gupta, A., Brown-Cohen, J., Goyal, A., and Arora, S. Skill-mix: a flexible and expandable family of evaluations for ai models, 2023a.

Yu, L., Jiang, W., Shi, H., Yu, J., Liu, Z., Zhang, Y., Kwok, J. T., Li, Z., Weller, A., and Liu, W. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023b.

Yu, Y., Zhuang, Y., Zhang, J., Meng, Y., Ratner, A. J., Krishna, R., Shen, J., and Zhang, C. Large language model as attributed training data generator: A tale of diversity and bias. *Advances in Neural Information Processing Systems*, 36, 2024.

Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

Zhang, H., Da, J., Lee, D., Robinson, V., Wu, C., Song, W., Zhao, T., Raja, P., Slack, D., Lyu, Q., et al. A careful examination of large language model performance on grade school arithmetic. *arXiv preprint arXiv:2405.00332*, 2024.

A. Appendix

Here we further analyse the quirks of the question generation pipeline and provide additional experimental details and results. In Appendix A.1 and Appendix A.2 we discuss several failure modes of the models which we notice during the question generation process as well as interesting behaviors exhibited by the models throughout the pipeline and interesting creative questions which the models came up with. Appendix A.3 discusses the different considerations which human annotators were intructed to take into account while annotating and verifying the questions generated by the proposed pipeline. Appendix A.4 provides details about the compute used for running our experiments as well as some further analysis of the model evaluations. Appendix A.5 gives a detailed description of the prompts used for each step in the question generation pipeline as well as evaluation of the models. It also provides an anonymized link to the skill exemplare repository and in-context exemplars used in the question generation process.

A.1. Failure Modes and Interesting Behaviors

Insufficient involvement of skills. Despite clearly specifying that solving the question should necessarily require a rigorous application of both skills, the models often generate questions that either miss one of the skills completely or require a very shallow application of one (while the other one is sufficiently involved) or both skills. This is the most prominent failure mode of the models in the context of question generation. This leads to potentially easy questions, defeating the purpose of skill composition. Consider the question given below which was generated by Claude Opus when asked to combine the skills `ratio_and_proportion` and `geometry`.

```
Question:  A square garden is to be divided into 4 smaller square plots by two
paths that are 1 meter wide and cross each other at right angles.  The paths run
North-South and East-West, splitting the garden symmetrically.  If the total area
occupied by the paths is 36 square meters, find the side length of the original
square garden.
```

Upon careful examination of the question, we note that although the question tests `geometry`, the involvement of `ratio_and_proportions` is practically non-existent. Further, the question validation step in some cases also fails to identify these flaws. Supplying multi-turn human-AI interactions where the user prompts a chatbot to generate a question combining two skills, in-context during the generation step helps the models to avoid such questions to a certain extent. Further, to make the question validation step more robust to such questions, we prompt the model to ensure that the complexity of each skill application in the question being validated in similar to or more than the complexity of these skills in the reference examples present in the skill descriptions. The combination of these two techniques helps us nearly eliminate questions where the absent one of the skills is absent completely and reduce questions involving shallow application of skills to a significant extent.

Insufficient information in the questions. Another common failure mode of the pipeline is the generated questions missing information or details essential for solving the question. For eg. in the question given below which is supposed to combine the skills `understanding_and_applying_floor_and_ceiling_functions` and `basic_trigonometry`, lacks sufficient detail about the inclinations and elevations of the paths relative to the streetlight's position which is necessary to answer the question.

```
Question:  Consider a scenario where you need to install a new streetlight
at a point such that it illuminates two paths meeting at a point, each path
making an angle of 45° with the horizontal.  The light from the streetlight
reaches a maximum distance of 10 meters on flat ground.  You are to install the
streetlight at the height of h meters (where h is the ceiling of the maximum
distance the light reaches horizontally) such that the edge of the light's reach
just touches the ground at the end of each path.  Determine the height h at which
the streetlight should be installed.
```

To screen such questions, we include and explicit clause in the question validation prompt as described in Section 2. Moreover, we also notice that the inclusion of the *solution attempt* step improves the chances of detecting such errors since the missing information may not always be apparent from just the question itself. In such cases, attempting a solution (with a defeatist approach) can help detect such flaws.

Unsolvable or Computationally Intractable Questions. There are instances when the model generates questions which are either unsolvable. For eg., the question given below has no solution which satisfies all three constraints (i.e., the area of the rectangle being 360 and the sides belonging to the two arithmetic progressions defined in the question.)

`Question 1: There's a rectangle with an area of 360 square units. The length of the rectangle is part of an arithmetic sequence starting at 5 and with a common difference of 7. If the other side of the rectangle is also part of an arithmetic sequence with the first term 10 and common difference 3, find the length of the shortest side of the rectangle.`

In other instances, the model generates questions that are computationally intractable or require manually and tediously iterating through a long sequence of values. For eg., solving the question given below requires manually calculating the first 100 terms of the sequence to find the sum

Question 2: Consider an infinite series of numbers arranged in sections, where the nth section contains the first $\binom{n+1}{2}$ positive integers that are divisible by n but not by any smaller positive integer (except 1). For example, the 1st section contains 1, the 2nd section starts with 2, 4, 6, 10, 12, and 16 the 3rd section starts with 3, 9, 15, 21, 33, ... and so on. Let S be the sum of the first 100 terms of this series. Find the sum of the digits of S.

While technically not wrong, such questions are not ideal for evaluating the *reasoning* abilities of the models since they mostly involve brute force calculations. Further, in cases where the sequence of calculations is very long, the LLM's performance may be bottlenecked by other limitations such as the context length of the model.

Thus, we strive to filter such questions out. We add an explicit condition to check for computational tractability and solvability of the generated questions in the verification prompt. This check is assisted by the *solution attempt* produced by the model which will potentially point out any such problems.

Nonsensical Questions. In several cases, the model comes up with questions which are nonsensical - confusing, incomprehensible, logically inconsistent or ambiguous. Consider the question given below.

Question 1: In a classroom of 30 students, each student is assigned a different integer from 1 to 30 as their badge number. If a student is selected at random, what is the probability that the student's badge number is such that the equation $3x - 4y =$ badge number has at least one solution in integers x and y?

It is not clear what the question is asking for. Given below is another example of a question which is logically inconsistent. More concretely, a square plot of land whose side length is equal to the radius cannot fit inside the quarter-circle.

`Question 2: A garden is designed in the shape of a quarter-circle with a radius of 8 meters. A square plot of land with a side length equal to the radius of the quarter-circle is placed inside this garden such that two of its sides are along the straight edges of the quarter-circle boundary. If the square plot of land is to be tiled entirely with square tiles each of area 64 square centimeters, what is the total number of tiles required?`

We add checks for such cases in the question validation prompt. Further, at the end of the final solution step (maj @ 4), we further check for cases where the final answer produced in all the 4 self-consistency trials are unique. If all answers are unique, we discard the question. The rationale behind this being that it is highly likely that the model produces a different answer every time due to some inherent ambiguity in the question which was not detected in the *solution attempt* and the *question validation* checks.

Deceitful Solutions. Although rare, we encounter cases where a the model makes up solutions even though the question is nonsensical or cannot be solved with the amount of information provided in the question. This happens very commonly in the solutions which are generated in the *question generation* prompt. Thus, we do not use these solutions and include the final solution step where the model is asked to solve the question again. Although, most of such solutions and thus questions are screened out in the *question validation* step and consistency check at the end of the *final solution* check, in rare cases we see this behavior in the solution produced after the *final solution* step as well. Given below is one such example.

Question: Consider the trigonometric identity $\sin^2(x) + \cos^2(x) = 1$ and the polynomial $P(x) = x^4 - x^2 - 12$. Using $x = \sin(\theta)$, solve $P(x) = 0$ for θ in the interval $[0, 2\pi)$.

While solving this question, the model arrives at the conditions $sin(\theta) = 2$ or $sin(\theta) = -2$. Clearly, these conditions have no solutions since $-1 \leq sin(\theta) \leq 1, \forall \theta \in [0, 2\pi]$. However, the model goes on to argue that the the closest values to 2 and -2 in the range of $\sin(\theta)$ are 1 and -1, and thus, goes on to solve the question for $sin(\theta) = 1$ or $sin(\theta) = -1$. We refer the reader to Appendix **??** for the full solution generated by the model.

Finding a needle in the haystack. In an attempt to make the *question validation* step robust to as many failure modes as possible, we come up with a long and sophisticated prompt (see Appendix A.5.4 for an example of this prompt). To elaborate, the validation prompt comprises of 1.) skill descriptions consisting of 3 exemplars for each of the 2 skills, 2.) 6 very long validation exemplars 3.) rest of the prompt consisting of the 7 conditions described in Section 2. In prior iterations of the validation prompt, the names of the two skills which are supposed to be combined in the question, were mentioned only once throughout the prompt. In such a case, we observed that more often than not, model confused the pair of skills at hand, with skills mentioned in the validation exemplars. We also observed cases where the model was unable to locate the original skill names, came up with its own skill names depending on the skill exemplars provided in the skill descriptions. More specifically, while checking the question for **Dual Skill Requirement** in Section 2, the models would check the question for the presence of skills distinct from the pair of skills at hand. We attribute this phenomenon to the the *lost in the middle* or *finding a needle in the haystack* challenges faced by LLMs. We provide an example of the phenomenon below. Notice how in Point 3 of the reasoning trace the model checks the question for the presence of two completely unrelated skills. We were able to nearly eliminate this problem by replacing referential terms (such as *above given skills*, *these skills*, etc.) to the two skills by the skill names.

```
Skill Pair to be checked for: [understanding_and_manipulation_of_rational_functions,
division_and_remainders]

Question: A fair 6-sided die is rolled 4 times. What is the probability that
the product of the numbers rolled is divisible by 36? Express your answer as a
simplified fraction.

Reasoning trace of the model:

# REASONING

Let's evaluate the given question and solution against the specified criteria:

1. **Single Answer Requirement**: The question asks for a single value - the
probability of a specific event occurring. This satisfies the single answer
requirement.

2. **Exact Answer Requirement**: The solution provides an exact answer in the
form of a simplified fraction (61/2386). No approximations or assumptions are
made, fulfilling the exact answer requirement.

3. **Dual Skill Requirement**: The question requires knowledge and application
of two distinct mathematical skills:      - Probability theory: Calculating the
probability of specific outcomes in a series of die rolls.      - Number theory:
Understanding divisibility rules for 4 and 9, and how they apply to the product
of numbers.

The solution demonstrates a rigorous application of both skills, using
probability calculations (such as the binomial probability for rolling a 4 at
least once) and number theoretic concepts (divisibility rules for 4 and 9) to
arrive at the final answer. The complexity of skill application is on par with
the provided examples.

4. **Clarity and Completeness**: The question is clearly stated and provides
all necessary information (fair 6-sided die, rolled 4 times, product divisible by
```

```
36) to solve the problem. No ambiguity or missing information is present.

5. **Relevancy of Information**: All given information is relevant and used in
the solution. There are no extra details that do not contribute to solving the
problem.

6. **Realism and Logic**: The scenario of rolling a die multiple times and
calculating probabilities is a realistic and commonly encountered problem in
probability theory. The data (6-sided die, 4 rolls) is logically consistent.

7. **Syntax and Grammar**: The question is grammatically correct and clearly
written, with no room for misinterpretation.

The solution is well-structured and easy to follow, breaking down the problem
into clear steps and explaining the reasoning behind each step. The mathematical
reasoning is sound, correctly applying probability and number theory concepts to
arrive at the final answer.

# FINAL ANSWER
Yes
```

Note that none of the above failure modes are completely eliminated in the pipeline described in Section 2. Thus, human verification is required.

Despite struggling with the failure modes described above, there also exist cases where the models exhibit positively surprising and creative behaviors. We talk about some of them below.

Thinking out of the box. Although rare, we observe instances where the models get creative while validating the question. Consider the question below

```
Question: A class of students is learning about combinatorics and geometry.
They are given a problem involving colored beads: Red, Blue, and Green. If
they need to form a necklace with 8 beads such that no two adjacent beads have
the same color and the necklace begins and ends with a bead of a different color,
how many different necklaces can they create? Each necklace is counted up to
rotation and reflection (considering the necklace can be flipped over).
```

When validating this question using prior iterations of the *question validation* prompt, which did not consist of the computational tractability check, the model output while validating the question consists of the following excerpt:

...This might introduce a significant challenge not solely due to the methodology's complexity but also due to the potential computational requirement, which may not be feasible in a standard test environment without tools. Furthermore, while the connection to practical geometry (reflective and rotational symmetry) and combinatorics (color patterning and adjacency constraints) is strong, the depth of understanding required to manually adjust for these symmetry considerations in a test question might be too intense or require more guided learning than a single evaluation question could provide....

i.e, the model takes into consideration the fact that the question involves a lot of brute force computation, despite there being no explicit check for computation complexity in the prompt, and classifies the question as in valid. We attribute such out-of-the-box thinking behavior to the role-playing nature of our prompts. Our prompts consist of a math teacher evaluating the the fitness of the given question for being used for testing students' reasoning and analytical skills in a math exam. This leaves room open for the model to detect potential problems not explicitly accounted for in the prompts which might make the question unfit for being used for evaluation.

A.2. Creative questions: Examples of Synergy from Human-AI interaction

The models had a nonnegligible rate of producing very interesting and creative questions. In such cases it often failed to produce the correct solution, but the failed solution had enough correct ideas to let the human quickly finish it.

Human annotators were tasked with double checking the validity of the question and the correctness of the solution. They were asked to look out for any of the failure modes discussed in Section 4.2. They had to check that the created question actually used the math skills it was supposed to exhibit and to improve the question with respect to readability, quality and difficulty when possible. They were encouraged to suggest changes that make the problem harder to solve using automated tools while allowing easier or more elegant solutions for the humans. We illustrate with examples.

`Original Question: Find the smallest positive integer` k `such that` $k^3 - k$ `is divisible by both` 9 `and` 10`, and the sum of digits of` k `in its decimal representation is a prime number.`

Our human team had not seen such questions before. It requires recognizing that $k^3 - k = k(k-1)(k+1)$ is always divisible by $2, 3$. Thus k must be such that we $k(k-1)(k+1)/6$ is divisible by 15, in other words by both 3 and 5. But, after that one has to worry about the about the sum of the digits of k being a prime number, and even powerful LLMs struggle to reason about a number's digits.

`Original Question: Consider a collection of red, blue, and green beads arranged in an infinite series. The beads alternate in color, starting with red, then blue, then green, and this pattern repeats indefinitely. The number of beads in each colored section follows the pattern of powers of 2: the first red section has 2 beads, the first blue section has 4 beads, the first green section has 8 beads, the second red section has 16 beads, and so on. If a bracelet is made using a continuous, unbroken sequence of exactly 20 beads from this series, and each bead has a length of 0.5 units, how many different bracelets can be made such that the perimeter of the bracelet is an integer value?`

Again, the question combines various elements that we had not seen combined before. The human rater modified the question to change the sequence size from 20 to 6, which retains the essential difficulty while making it elegant for humans. All tested models failed on the modified question.

`Original Question: A container initially contains 500 mL of water. A scientist adds water to the container` $\frac{1}{4}$`th of the current amount every minute. After how many minutes will the container first contain more than 1 L but less than 2 L of water?`

`Modified Question: A container starts with 500mL of water. At every minute the scientist adds water to the container equal to` $1/2$ `of the current amount. What is the smallest positive integer` n `such that the number of liters of water in the container is never in the interval` $[n, n+1]$`?`

This was one of many questions the models created about exponential growth and geometric series. We guess it is possibly similar to questions in standard math tests. The human slightly changed it to simplify calculation by hand, and to substituted a different condition obviously holds and the precise n is easier for humans to guesstimate. This question was hard for most models.

`Original Question: Consider the sequence defined recursively by` $a_1 = 1$ `and` $a_{n+1} = 2a_n + n$ `for all` $n \geq 1$`. What is the product of the first five terms of this sequence?`

`Modified Question: A sequence` $\{a_n\}$ `is defined as follows.` $a_1 = 2$ `and` $a_n = 2^{n-1} + a_{n-1} + n$`. What is the` $\lfloor \log_2 a_{500} \rfloor$`?`

An LLM can solve the original question above by simple computation. The modified the question requires understanding an underlying pattern.

`Original Question: Find the sum of the smallest prime divisor and the largest prime divisor of the number` $N = 15^4 + 16^4$`.`

`Modified Question: Find the sum of the two smallest prime divisors of` $23^{17} + 17^{17}$`.`

Models tend to adopt brute force approach on the original question, calculating $15^4 + 16^4$. After the rephrasing, the number $23^{17} + 17^{17}$ is too large for direct computation, and they need to understand arithmetic modulo a prime.

A.3. Considerations for human-annotaters

Human annotators were tasked with double checking the validity of the question and the correctness of the solution. They were asked to look out for any of the failure modes discussed in Section 4. They were asked to check that the created question actually used the math skills it was supposed to exhibit and to improve the question with respect to readability, quality and difficulty. They were encouraged to suggest changes that make the problem harder to solve using automated tools while retaining easiness for the humans. We illustrate with an examples.

GPT-4 created the following question given the skill-tags `recursive_functions_and_sequences` and `multiplication_and_division`:

Original Question: Consider the sequence defined recursively by $a_1 = 1$ and $a_{n+1} = 2a_n + n$ for all $n \geq 1$. What is the product of the first five terms of this sequence?

An LLM can solve this by simple computation. The human modified the question so that solving the problem requires understanding the underlying pattern.

Modified Question: A sequence is defined recursively as follows: the first term a_1 is 2, and for $n \geq 2$, $a_n = 2^{n-1} + n$. What is the logarithm (base 2) of the average of the first 50 terms of this sequence? Round down to the nearest integer.

For the modified question, one leading model mentioned calculation difficulties for the inability to give any answer, and another resorted to an incorrect numerical approximation that led to an incorrect answer.

Human annotators were also asked to go through the solutions carefully and correct or improve the solution for good questions if necessary. They were also asked to look out for questions that contain lot of enumeration, i.e. questions which are tedious and require significant amount of brute force computation. For such questions, the annotators were encouraged to reword them such that enumeration is not a feasible strategy below. For eg. given below is an example of an enumerative question which was modified to avoid enumeration.

Original Question: Find the sum of the smallest prime divisor and the largest prime divisor of the number $N = 15^4 + 16^4$.

Modified Question: Find the sum of the two smallest prime divisors of $23^{17} + 17^{17}$.

Models tend to adopt brute force approach on the original question calculating $15^4 + 16^4$. After rephrasing the models cannot use brute force on $23^{17} + 17^{17}$, instead being forced to check the divisors more analytically, in particular understanding of arithmetic modulo a prime.

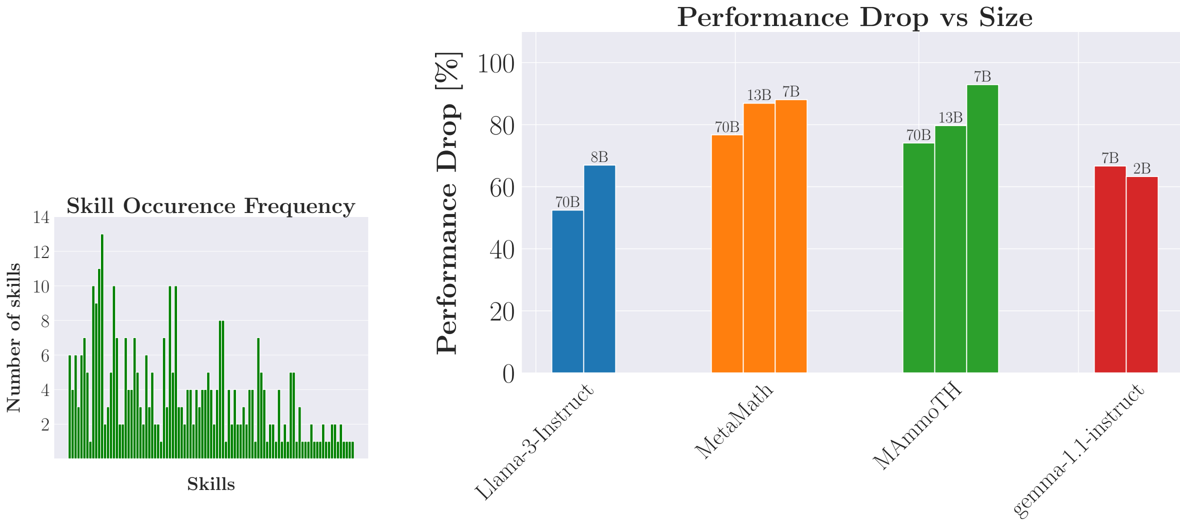
A.4. Further Experimental Details and Results

For open source LLMs, 2 80GB A100 GPUs and 72GB of RAM to run inference facilitated by vLLM (Kwon et al., 2023). For evaluating GPT-4 Omni and GPT-4 Turbo, we use 25 parallel workers to make the API calls, and for Claude-3 Opus we use 2 workers.

Figure 3(a) shows the distribution of different skills in the generated set of questions. Figure 3(b) shows the comparison in relative performance drops between models of different sizes belonging to the same family.

A.5. Prompt Examples

We detail the prompts used in the various stages of question generation and evaluation in the following sections. We present the skeleton of the prompts with placeholders for skill names, in-context exemplars, etc. in favor of keep the prompt brief. The in-context exemplars and the skill exemplar repository are made available here.



(a) (b)

Figure 3. (a) Shows the distribution of different skills extracted during the skill extraction process in the generated set of questions. The generated and human verified set of 180 questions consists of 97 skills out of the 114 skills extracted via the skill extraction process as described in (Didolkar et al., 2024), Each question in the generated set represents two skills. Note that the distribution of skills is not uniform with there being multiple skills which are represented by one one question. (b) Comparative bar plot showing smaller models generally show larger degradation in performance (relative to MATH) as compared to their larger counter-parts, with the exception of the Gemma family of models, where the 7B parameter model shows a larger deterioration of 66.669% as compare to 63.03% in the 2B parameter model

A.5.1. Skill Pair Validation

Prompt for validating a pair of skills

I have two basic mathematical skills described below, each followed by some sample question - solution pairs such that solving the questions involves the use of the particular skill.

SKILL 1: <skill_1>

Question 1: <skill_1_question_1>
Answer 1: <skill_1_solution_1>

Question 2: <skill_1_question_2>
Answer 2: <skill_1_solution_2>

Question 3: <skill_1_question_3>
Answer 3: <skill_1_answer_3>

SKILL 2: <skill_2>

Question 1: <skill_2_question_1>
Answer 1: <skill_2_solution_1>

Question 2: <skill_2_question_2>
Answer 2: <skill_2_solution_2>

Question 3: <skill_2_question_3>
Answer 3: <skill_2_solution_3>

I am going to use these two skills for framing a new question such that the question requires an expertise in both the skills in order to be solved, i.e. the question will compose these two skills. However, I do not want the two skills to be very similar, i.e., they should not mean the same thing. Go through the descriptions of the skills carefully. Based on your understanding of the skills, can you please tell me whether the two skills are essentially entirely the same or not? Think step by step and give a detailed explanation of your answer. The answer should begin with a prefix '# EXPLANATION '. Note that your understanding of the skills should not be restricted to the sample questions provided previously. They are just example questions. Use your own prior knowledge as well. End your response with a 'Yes' or 'No' answer to whether the skills are similar or not. This final answer should be on a new line and preceded by the prefix '# FINAL ANSWER '. Thank you very much!

A.5.2. Question Generation

Prompt for question generation

I am a math teacher trying to create challenging math questions for smart students. I was wondering if you could give me 1 (non multiple choice) question which tests both the following skills: (<skill_1>, <skill_2>) Please also provide a brief solution. Then please look over the question and the solution, and fix any issues so that my students do not get frustrated. This being a math exam, the answers should either be exact, or if not possible, then the question should clearly say the answer is only expected to be approximately correct. Further, for ease of evaluating the students' answers, the question should ask for a single final result.This process is difficult so I am attaching two sample conversations where (Agent) is an AI agent and (Query) is teacher feedback. The conversations revolve around framing such mathematical reasoning questions and using them for evaluating students. These should give you some idea of the expectations and the potential difficulties involved in this task. I am also giving three example question - answer pairs for both <skill_1> and <skill_2> skills, such that the example questions test the corresponding skill. Please ensure that the complexity / difficulty of application of <skill_1> and <skill_2> skills in the generated question is similar to the complexity / difficulty of the skills in the example questions. Please format your output as

'# QUESTION
<question>

SOLUTION
<solution>

DETAILS
<all other text>'

SKILL 1: <skill_1>

Question 1: <skill_1_question_1>
Answer 1: <skill_1_solution_1>

Question 2: <skill_1_question_2>
Answer 2: <skill_1_solution_2>

Question 3: <skill_1_question_3>
Answer 3: <skill_1_solution_1>

SKILL 2: <skill_2>

Question 1: <skill_2_question_1>
Answer 1: <skill_2_solution_1>

Question 2: <skill_2_question_2>
Answer 2: <skill_2_solution_1>

Question 3: <skill_2_question_3>
Answer 3: <skill_2_solution_3>

CONVERSATION 1
<agent_convo_1>

CONVERSATION 2
<agent_convo_2>

A.5.3. Attempted Solution

Prompt for solution attempt. Note that we instruct the model to take a defeatist approach towards solving the question

Prompt for solution attempt

You are a professional math teacher and you are given a question which is supposed to test the analytical and mathematical reasoning abilities of your students. You are supposed to provide a solution to the given question. However, the question may be flawed. For eg. it might have problems like question being unsolvable using the information provided, question being self-contradictory, the final answer being computationally intractable, the question being ambiguous and confusing, question having multiple possible interpretations, etc., which you may encounter while solving the problem. This question being used for evaluating students in math, the question should ideally have a single, exact answer, with no room for any deviations due to factors such as approximations, rounding errors, etc., unless explicitly specified in the question. Problems such as the ones described above, would prevent the students from solving the question properly, and thus, any question with either of these problems is unfit for testing the students. If you encounter any such problems, stop the solution right there and explain the problems. For eg. if you encounter the need to make any approximations or rounding which is not specified in the question, stop solving the question along with the reason. You do not need to solve the question further once you encounter any such problem. If you do not encounter any such problem, solve the question to achieve the single exact answer which the question asks for.

QUESTION
<question>

A.5.4. Question Validation

Note that how in the first paragraph, the names of the two skills are mentioned even time instead of using referential phrases. This is done to address the *lost in the middle* problem

Prompt for validating the questions

You are a professional math teacher. You want to evaluate the analytical and mathematical reasoning abilities of your students in a math exam. The students are supposed to sit in an examination hall and solve the questions within a given time limit, without access to any computational devices. The evaluation is designed to test the students' expertise in using two given mathematical skills simultaneously, namely <skill_1> and <skill_2>. This is achieved by asking them to solve a question that necessitates expertise in both <skill_1> and <skill_2> skills, to be solved completely. Since evaluating the students is a critical task allowing very little margin for any error in the process, it is very important to ensure that the questions used for evaluating are high quality and fit for being used to evaluate the students. You need to carefully review the question and a given attempt at solving it, and ensure that the question is of high quality and fit to assess students. In order to do this, you should check the quality of the question with respect to several criteria, such as:

- Single Answer Requirement: The question should ask for one and only one final result. It should not request multiple distinct answers or pieces of information.
- Exact Answer Requirement: It should be possible to achieve one,exact answer to the question, without the need of making any approximations or assumptions whatsoever, unless explicitly specified in the question. There should be no margin for the students to arrive at any other possible answer due to things like rounding errors, etc.
- Dual Skill Requirement: The question must require rigorous expertise in both a: a) '<skill_1>' and b) '<skill_2>', for resolution. Application of both <skill_1> and <skill_2> and their subskills should be, necessary and contribute directly to obtaining the final answer; <skill_1> and <skill_2> skill should be applicable separately and critically during the problem-solving process. You are also given three example question - answer pairs for both <skill_1> and <skill_2> skills in order to help you better understand the meaning of each skill. Please carefully review the question and its attempted solution, paying close attention to how well it aligns with the examples provided for each skill. Consider the depth and breadth of knowledge demonstrated in the examples. The complexity / difficulty of application of both <skill_1> and <skill_2> in the question should be similar or greater than the complexity / difficulty of <skill_1> and <skill_2> in the example question-answers given for that respective skill.
- Clarity and Completeness: The question should be unambiguous and contain all the information necessary to complete the solution. Any required assumptions not common knowledge should be explicitly stated. Check for any ambiguity that might confuse students. Carefully go through the solution to check if it makes any assumption or approximation in order to solve the question.
- Computational Tractability: Since the students are supposed to solve the questions within a given time limit and without access to any computational devices such calculators, computer, mobile phones, etc., you must ensure that the question is computationally tractable and all the computations involved can be done by hand in a limited amount of time.
- Relevancy of Information: The question should not have any extra details that do not contribute to the solving of the problem.
- Realism and Logic: The question should involve realistic scenarios or hypotheses with logically consistent data. The specified operations and the contextual setup should reflect plausible mathematical situations. (e.g., positive amounts for transactions, integers for counts).
- Syntax and Grammar: The question must be grammatically correct and clearly written to prevent misinterpretation.
- etc. (any other problems which you think make the question not fit for being used for evaluating the students)

Your task is to give a 'Yes' or 'No' assessment, indicating whether the question is high quality and suitable for evaluating the students on simultaneous application of the skills <skill_1> and <skill_2>. Provide thorough reasoning for your assessment based on the conditions mentioned above and any other relevant analytical points concerning mathematical reasoning and problem-solving. Your response should be structured as follows:

REASONING
<Your detailed analysis justifying your decision>

FINAL ANSWER
<'Yes' or 'No'. No other text should be present in this section>

Ensure to review the combination of skills intended for assessment, and check the logical flow and mathematical correctness from the question's setup to the solution's conclusion. Look out for any problems in the question which are pointed out in the attempted solution. Account for all the potential pitfalls such as logical inconsistencies, unnecessary complexity, or insufficient detail that may obstruct the clarity or solvability of the question. Given below are the two skills and some example question-answer pairs for the two skills. This process is difficult so I am attaching a few sample conversations where (agent) is an AI agent who is trying to verify the questions and (query) is teacher feedback. This should give you some idea of potential difficulties in this task. This is followe by the question which you need to check (preceded by ' QUESTION TO BE CHECKED') and its attempted solution (preceded by ' SOLUTION ATTEMPT').

SKILL 1: <skill_1>

Question 1: <skill_1_question_1>
Answer 1: <skill_1_solution_1>

Question 2: <skill_1_question_2>
Answer 2: <skill_1_solution_2>

Question 3: <skill_1_question_3>
Answer 3: <skill_1_solution_3>

SKILL 2: <skill_2>

Question 1: <skill_2_question_1>
Answer 1: <skill_2_solution_1>

Question 2: <skill_2_question_2>
Answer 2: <skill_2_solution_2>

Question 3: <skill_2_question_3>
Answer 3: <skill_2_solution_3>

CONVERSATION 1
<validation_exemplar_1>

CONVERSATION 2
<validation_exemplar_2>

......

CONVERSATION 6
<validation_exemplar_6>

QUESTION TO BE CHECKED
<question>

SOLUTION ATTEMPT
<solution>

Thank you very much!

A.5.5. Final Solution

For the final solution, we make use in-context exemplars from MATH (Hendrycks et al., 2021) as opposed to the attempted solution step.

Prompt for the final solution

I have two basic mathematical skills described below, each followed by some sample question - solution pairs such that solving the questions involves the use of the particular skill in order to be solved.

SKILL 1: <skill_1>

Question 1: <skill_1_question_1>
Answer 1: <skill_1_solution_1>

Question 2: <skill_1_question_2>
Answer 2: <skill_1_solution_2>

Question 3: <skill_1_question_3>
Answer 3: <skill_1_solution_3>

SKILL 2: <skill_2>

Question 1: <skill_2_question_1>
Answer 1: <skill_2_solution_1>

Question 2: <skill_2_question_2>
Answer 2: <skill_2_solution_2>

Question 3: <skill_2_question_3>
Answer 3: <skill_2_solution_3>

Go through the descriptions of the skills carefully. Now, here is a new question such that the question requires an expertise all both the skills in order to be solved. That is, the question composes these two skills

QUESTION: <question>

Based on your understanding of the skills, can you please solve the question accurately? Think step by step and explain the solution. Finally, end your response by stating the final numerical answer obtained using the solution. Note that your understanding of the skills should not be restricted to the sample questions provided in their description. They are just example questions. Use your own prior knowledge as well. The explanation of your solution and the final numerical answer should each be on a new line, and should be preceded by the prefixes '# SOLUTION ' and '# ANSWER ' respectively. Thus, your response should be in the format:

'# SOLUTION
<solution>

ANSWER
<final_answer; no other text should be present in this section>'.

Thank you very much!

A.5.6. Evaluation

Prompt given to the GPT-4 for evaluating the model's solution

You are a professional math teacher and are tasked with evaluating your students on a math exam. You are will be given a question, the correct solution to the question and the student's solution. You need to tell me whether the student solved the question correctly, thus matching the answer obtained by the correct solution. Think step-by-step and give a detailed explanation of your answer. At the end, give a 'Yes' or 'No' answer to whether the student's solution is correct. Your output should be in the following format:

STEP BY STEP EXPLANATION
<detailed explanation of your thought process>

CORRECTNESS
<'Yes' if the student's solution is correct. 'No' otherwise. This section should not contain any other text>

Here are the question, correct solution to the question and the student's solution:

QUESTION: <question>

CORRECT SOLUTION: <correct_solution>

STUDENT'S SOLUTION: <student's_solution>