Key-Point-Driven Data Synthesis with Its Enhancement on Mathematical Reasoning

Yiming Huang*†, Xiao Liu†, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, Weizhu Chen

Microsoft
yeeelow233@gmail.com, xiaoliu2@microsoft.com

Abstract

Large language models have shown great potential in complex reasoning tasks, yet their performance is often hampered by the scarcity of high-quality and reasoning-focused training datasets. Addressing this challenge, we propose Key-Point-Driven Data Synthesis (KPDDS), a novel data synthesis framework that synthesizes question-answer pairs by leveraging key points and exemplar practices from authentic data sources. KPDDS ensures the generation of novel questions with rigorous quality control and substantial scalability. As a result, we present KPMath, an extensive synthetic dataset tailored for mathematical reasoning, comprising over 800K question-answer pairs. Utilizing KPMath and augmenting it with additional reasoning-intensive corpora, we create the comprehensive KPMath-Plus dataset. Our experiments demonstrate that this dataset can enhance the mathematical reasoning performance of models across various architectures and sizes. The Qwen1.5-72B model, fine-tuned on KPMath-Plus, achieves 87.0% accuracy on GSM8K and 58.3% on MATH, surpassing competitors in the 7B to 72B range and best commercial models like GPT-4 across multiple math reasoning datasets.

Introduction

The recent advent of large language models (LLMs) such as GPT-4 (OpenAI 2023), Gemini (Team et al. 2023), and Mistral (Jiang et al. 2023) has sparked significant interest due to their advanced capabilities in diverse domains. Despite this, their reasoning process, particularly in challenging domains like advanced mathematics (Lewkowycz et al. 2022), competitive programming (Huang et al. 2023), and integrated vision-language planning (Cen et al. 2024), remains under scrutiny. In current mathematical reasoning corpora, such as OpenWebMath (Paster et al. 2023) and MathPile (Wang, Xia, and Liu 2023), the vast internet-sourced data often suffers from poor quality and relevance to the subject matter. Conversely, manually annotated high-quality datasets like the MATH dataset (Hendrycks et al. 2021) are scarce and sometimes lack detailed reasoning steps.

Prior efforts to boost the mathematical reasoning capabilities of LLMs using synthetic data have primarily adopted two strategies. The first strategy focuses on augmenting existing datasets. It involves question rephrasing (Yu et al. 2023) or generating similar questions (Luo et al. 2023a; Liu et al. 2024). The primary issue of this strategy is that the generated questions are not only textually or conceptually similar but also uncontrollable in their variations. The second strategy seeks to broaden the training dataset by generating new questions from knowledge concepts. Knowledge bases are either compiled from online educational resources, such as Khan Academy's math courses (Huang et al. 2024), or synthesized from scratch using models like GPT-4 (Li et al. 2024). These methods depend on constructed knowledge that might not align with the existing dataset's distributions and are difficult to comprehend without examples to illustrate the concepts.

Considering these disadvantages of the two strategies, we introduce a novel data synthesis paradigm termed **Key-Point-Driven Data Synthesis (KPDDS)**, which capitalizes on the strengths of both data synthesis strategies. As depicted in Figure 1, it delves into datasets for knowledge mining, using relevant key points and associated problems to inform the generation of new problems. (1) For knowledge construction, we begin by extracting topics and key points from seed problems using a labeling model, followed by a clustering algorithm to ensure deduplication and alignment. Therefore, we obtain the Math Practices with Key Points (MPKP) dataset and construct the Topic-level Co-occurrence Probability Matrix (TCPM) to understand the combination patterns of topics within the dataset. (2) For practice synthesis, we sample multiple topics and key points from MPKP using the TCPM as a guide. These key points, along with corresponding example practices, serve as input for the synthesizing model to generate new questions. A scoring model then assesses the quality of these questions, allowing only those with high scores to proceed. Then, a reasoning model generates a range of answer options, which are later consolidated into consensus solutions through a voting mechanism.

Utilizing the training sets of the MATH (Hendrycks et al. 2021) and GSM8K (Cobbe et al. 2021) datasets as foundational data, we developed a novel dataset named **KPMath**. Our training corpus was further enriched by integrating a series of mathematical reasoning datasets, leading to the creation of a comprehensive training dataset, **KPMath-Plus**. Our experiments demonstrate that this dataset can enhance the mathematical reasoning performance of models across

*This work was done during the internship of Yiming Huang at Microsoft.

†Corresponding author.

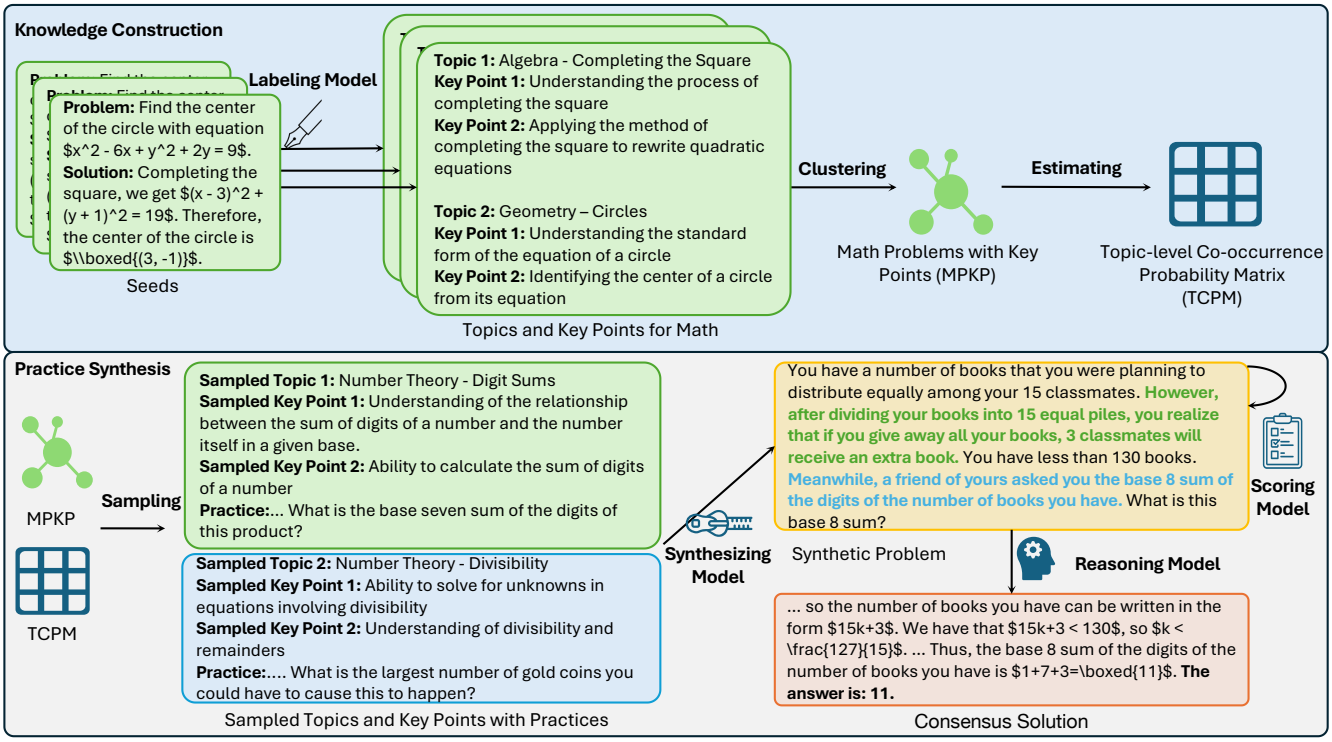

Figure 1: Overview of the Key-Point-Driven Data Synthesis (KPDDS) pipeline, from knowledge extraction to practice synthesis.

various architectures and sizes. By fine-tuning the Qwen1.5-72B model (Bai et al. 2023) on KPMath-Plus, we achieved zero-shot PASS@1 accuracies of 87.0% on the GSM8K test set and 58.3% on the MATH test set, culminating in a promising average of 81.5% across six math reasoning datasets. This performance exceeds that of all competitors within the 7B to 72B model size range and best commercial models like GPT-4. In the Hungarian Exam Score test, the KPMath-Plus-Mistral-7B model also outperforms the majority of models, indicating its competitive performance.

Related Work

Math Reasoning with LLMs

Recently, solving math problems is treated as an important aspect of evaluating LLM's reasoning ability. However, the LLMs trained for general purposes like GPT-4 (OpenAI 2023), Llama2 (Touvron et al. 2023), Mistral (Jiang et al. 2023), Qwen (Bai et al. 2023), Gemini (Team et al. 2023) and DeepSeek (Bi et al. 2024) have shown limited capabilities in math reasoning. To enhance the math reasoning ability of LLMs, researchers have turned their attention to research directions like prompting methods (Chia et al. 2023; Zheng et al. 2023; Chen et al. 2023), data construction for pretraining (Taylor et al. 2022; Lewkowycz et al. 2022; Paster et al. 2023; Azerbayev et al. 2023) and instruction tuning (Yue et al. 2024; Yu et al. 2023; Luo et al. 2023a; Gou et al. 2024b; An et al. 2023; Liu et al. 2024; Huang et al. 2024; Li et al. 2024, 2023), interacting with external tools (Mishra et al. 2022; Gao et al. 2022; Gou et al. 2024a,b; Yue et al. 2024; Zhou et al. 2023; Zhang et al. 2024), and reinforcement learning with rewards (Lightman et al. 2023; Wang et al. 2023; Luong et al. 2024) for either outcomes or steps. This work is in line with math reasoning data construction for instruction tuning.

Data Synthesis for Math Reasoning

In the realm of math reasoning, data synthesis is usually applied for instruction tuning, with each data sample encompassing a question text and its corresponding answer text. To advance this field, research efforts focus on three critical aspects: enhancing the quality of answers, generating novel questions, and implementing quality control measures.

For answer quality, some works focus on chain-of-thought (CoT) (Chia et al. 2023; Yu et al. 2023) style answers, while others investigate program-based answers. Yue et al. synthesize program-of-thought (PoT) (Chen et al. 2022) style answers using GPT-4. Gou et al. further explore interleaved answers with program-based tool use. In this work, we focus on the synthesis of CoT-style answers.

For question novelty, some works start from existing problems. Shao et al. (2023) explore answer-first data synthesis and Yu et al. (2023) utilize backward reasoning, while Luo et al. (2023a), An et al. (2023), and Liu et al. (2024) focus on evolution instruction and iterative composition using reasoning steps. Other works are grounded in knowledge. Huang et al. (2024) extracts concepts from Khan Academy and Li et al. (2024) uses GPT-4 to create a concepts taxonomy. The former approach is limited by poor scalability with existing data, and the latter often yields a synthetic data distribution that significantly deviates from real data. In our work, we

create questions by extracting key points from real data and then synthesizing new problems based on these key points with authentic and reliable exercises.

For synthetic data quality, Huang et al. (2024) prompt GPT-4 to convert CoT-style answers into verifiable Lean-3 code, while Trinh et al. (2024)'s AlphaGeometry ensures Euclidean geometry theorem accuracy using symbolic deduction. In contrast, We assess synthetic question and answer quality through GPT-4 scored evaluations and consensus scoring via repeated sampling.

Data Synthesis for Other Applications

The aim of synthetic data is to offer a convincing and fuller depiction of the actual data source, maintaining key statistical characteristics such as the distribution patterns of continuous variables, categorical ratios, and the latent relationships among different variables. Except for math reasoning, there are also works on data synthesis for other applications like code (Luo et al. 2023b; Wei et al. 2023), table reasoning (Lei et al. 2023), medical application (Zhang et al. 2023; Tang et al. 2023), visual reasoning (Du et al. 2023), and general purposes (Xu et al. 2023; Li et al. 2024).

Method

Overview

The KPDDS methodology is systematically delineated into two primary phases: Knowledge Construction and Practice Generation, each consisting of two components. We will introduce these four components separately: Knowledge Extraction, Topic-level Co-occurrence Probability Matrix (TCPM) Construction, Question and Solution Generation.

Knowledge Extraction

We employ GPT-4 to extract knowledge on each question and answer pair across two levels. The first level of knowledge is the topics, which correspond to the subject and its subcategories that are pertinent to the problem, such as "Geometry - Circles". The secondary level is key points (KPs), which comprise the theorems or methods essential for the resolution process, like "Determining the center of a circle from its equation". Figure 1 presents a complete example.

> As a mathematics education specialist, please analyze the topics and key points of the provided question and its answer. These analysis should serve as a guide for teachers to craft analogous problems and as focal learning objectives for students when approaching the problem. Be sure to avoid repetition of Key Points for clarity and conciseness. Specific requirements are as follows:
> 1. Identify and categorize the main mathematical topics involved in the problem. If knowledge from non-mathematical fields is used, it is classified into Others - xxx, such as Others - Problem Context.
> 2. For each topic, enumerate the essential Key Points relevant to the problem.
> ... [omitted two examples]

Figure 2: Prompt snippet for knowledge extraction.

To ensure that the extraction captures a wide range of pertinent information without constraint, we allow the model to freely identify and categorize relevant subjects instead of providing a predefined list of topics in the prompt. However, this approach can result in an extensive number of topics, many of which exhibit semantic overlap, such as "Arithmetic - Percentages" and "Arithmetic - Percentage." Therefore, we further process the extracted knowledge data. Specifically, we use OpenAI's text-embedding-ada-002 to embed all KPs, represent the topics by the average value of the embeddings of their included KPs. Then, we calculate the cosine similarity of the topic embeddings for deduplication and clustering, obtaining several representative topics. Finally, we construct the Math Practices with Key Points (MPKP) dataset.

Algorithm 1: TCPM Calculation

Require: MPKP dataset with N unique topics
Initialize $TCPM$ matrix of size $N \times N$ with zeros
for each document d in dataset **do**
  for each topic i in d **do**
    for each topic j in d **do**
      if $i \neq j$ **then**
        $TCPM[i][j] \mathrel{+}= 1$
      end if
    end for
    if Number of KPs in topic $i > 5$ **then**
      $TCPM[i][i] \mathrel{+}= 1$
    end if
  end for
end for
$TCPM \leftarrow \log_{10}(TCPM + 1)$
return $TCPM$

TCPM Construction

Mathematical problems typically involve multiple topics and KPs, and the combination of topics within these problems follows a discernible pattern. For example, semantically highly similar topics do not appear repeatedly in the same problem, whereas arbitrarily meshing unrelated topics tends to result in nonsensical questions. In light of this structured complexity, we compute the Topic-level Co-occurrence Probability Matrix (TCPM) from the topics present in mathematical questions within the MPKP dataset. Our methodology is systematically outlined in algorithm 1. This algorithm quantifies the co-occurrence and self-interaction of topics within a dataset by constructing a matrix that logs the frequency of topic pairs and the instances where the number of KPs for individual top-

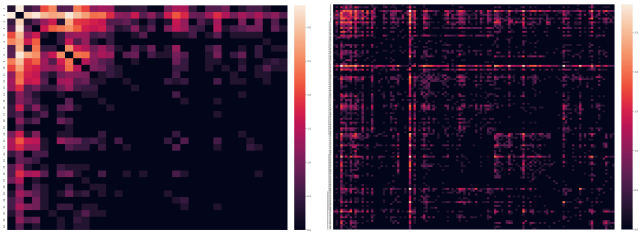

Figure 3: Visualized heat map of Topic-level Co-occurrence Probability Matrix. Left: GSM8K (34 topics), Right: Math (119 topics).

ics exceeds five, followed by a logarithmic normalization. An increased co-occurrence probability between topic clusters indicates a likelihood of their concurrent appearance in the examined problems. Figure 3 presents heat map visualization of the TCPM for GSM8K and MATH.

Question Generation

By extracting knowledge and constructing the TCPM from the seed problems, we pave the way for generating new problems that are similar yet varied in nature, building upon their foundational elements. Leveraging the TCPM, we perform probabilistic sampling of topics, with the probability calculation method as follows:

$$V_n = \begin{cases} \sum_j \text{TCPM}_{ij}, & \text{if } n = 1, \\ \text{TCPM}_{T_i,\cdot}, & \text{if } n = 2, \\ \text{TCPM}_{T_{n-1},\cdot} \circ \text{TCPM}_{T_{n-2},\cdot}, & \text{if } n > 2, \end{cases} \quad (1)$$

where V_n represents the vector used for probabilistic topic sampling, i and j are index variables, T_i denotes the i-th topic, and $\text{TCPM}_{T_n,\cdot}$ denotes the n-th row vector in TCPM. $\circ$ denotes the Hadamard product (element-wise multiplication).

We proceed to sample two to three topics, and for each topic, we randomly select a problem along with the associated KPs for that topic. This process yields a foundational KPs-Practices information set as the basis for our problem generation. Employing GPT-4, we use this set to generate new problems, with the prompt presented in Figure 4.

> You are a math teacher. Now, you need to help your students to learn the following math knowledge. There are some key points and example problems:
> {2 or 3 KPs-Practices information sets }
> Using these key points and example problems as a guideline, please construct a new, original math problem that requires an understanding and application of all the {len of selected kps} knowledge points.
> {selected kps}
> Write your new math problem, using <Q>and </Q>to indicate the question.

Figure 4: Prompt snippet for question generation.

Following the generation of problems, we conduct a quantitative evaluation to determine the quality of each problem by GPT-4. This assessment is based on two criteria: the presence of the provided KPs and the absence of logical or factual errors. Each problem is assigned a quality score on a continuous scale from 0 to 1. In assembling quality-assured questions, a threshold of 0.85 is instituted to screen the newly generated problems, saving about 51% high-quality question. Figure 5 displays an example of a high-quality and a poor-quality problem originating from identical initial inputs.

Solution Generation

Prior work in the domain often lacked comprehensive quality control measures, relying heavily on answers generated by models like GPT-4 without additional verification. Our

> **High-quality** ✓ You have a number of books that you were planning to distribute equally among your 15 classmates. However, after dividing your books into 15 equal piles, you realize that if you give away all your books, 3 classmates will receive an extra book. You have less than 130 books. Meanwhile, a friend of yours asked you the base 8 sum of the digits of the number of books you have. What is this base 8 sum?

> **Poor-quality** ✗ You discover less than 1000 gold coins and plan to share them with 13 hunters, giving 3 extra coins to some. An ancient riddle claims the sum of the coins' digits in base 13 equals the coins' remainder when divided by 10. What's the maximum number of coins possible?

Figure 5: Examples of high-quality and poor-quality synthetic questions generated from the same initial inputs.

methodology integrates three key strategies to improve answer correctness: few-shot learning to guide the initial response generation, computational verification to validate the answers mathematically, and a consensus voting mechanism to ensure the accuracy of the results. This multi-faceted approach is designed to minimize the impact of noisy data and enhance the reliability of the answer generation process.

For each synthesized question, we utilize the same KPs-Practices information set that were employed in its creation as demonstration inputs. This approach ensures that the examples, which inherently share key points with the question, significantly enhance the quality of the model's responses by aligning closely with the problem's structure and content. To generate a diverse array of CoT rationales, we employ nucleus sampling and configure GPT-4 with a temperature setting of 0.75 and a top-p value of 0.95. For each question, we derive 10 potential responses. Next, we extract mathematical expressions from each solution and use a Python program to verify their accuracy, excluding any solutions with computational errors. Additionally, we exclude data containing more than three sub-questions to maintain analytical clarity.

Finally, we use a voting mechanism to aggregate the solutions. The voting mechanism leverage packages such as `sympy` [1] to ensure that equivalent answers, albeit in different forms (e.g., fractions and decimals), are recognized as equal. Consider a problem p with n sub-questions that has m potential solutions. Let s_i denote one of these solutions, and each solution s_i addresses the n sub-questions within the problem. The Consensus Score Vector (CSV) for a solution s_i is defined as:

$$\text{CSV}(s_i) = [c_{i1}, c_{i2}, \ldots, c_{in}] \quad (2)$$

where each c_{ij} represents the consensus score for the j-th sub-question in solution s_i. The range of each c_{ij} is from $\frac{1}{m}$ to 1. A score of 1 indicates that all solutions agree with s_i for the sub-question. A score of $\frac{1}{m}$ indicates that s_i is the only solution with its particular answer, showing no agreement with other solutions. The Consensus Score (CS) for a solution s_i is represented by the maximum value in its CSV:

$$CS(s_i) = \max(\text{CSV}(s_i)) \quad (3)$$

[1] https://www.sympy.org

Define CS_{max} as the highest consensus score among all solutions for problem p:

$$CS_{\text{max}} = \max(\{CS(s_1), CS(s_2), \ldots, CS(s_m)\}) \quad (4)$$

A solution s_i is retained if it meets the following criteria:

$$\text{Retain}(s_i) = \begin{cases} \text{YES}, & \text{if } CS(s_i) = CS_{\text{max}} \text{ and } CS(s_i) > \frac{1}{m}, \\ \text{NO}, & \text{otherwise.} \end{cases} \quad (5)$$

KPMath-Plus Dataset

Dataset Construction

The KPMATH-Plus dataset is composed of three distinct components, collectively encompassing a total of 1,576K data points. We use min-hash techniques to minimize redundancy and exclude entries featuring excessively long numbers. Here are the unique attributes of each segment:

KPMATH-G (613K) This segment is based on the GSM8K (Cobbe et al. 2021) training set, which offers 7,473 samples of grade school math problems characterized by their 2 to 8 step solutions. KPMATH-G focuses on basic arithmetic operations within various contexts,

KPMATH-M (252K) This segment is based on the MATH (Hendrycks et al. 2021) dataset's training set, which consists of 7,500 samples from high school math competitions, encompassing seven subjects and five difficulty levels. KPMATH-M includes more challenging problems in areas such as algebra, calculus, and geometry, designed to improve advanced mathematical thinking and problem-solving skills.

MixMath (711K) To ensure diversity and quality, we curated a comprehensive collection from various high-quality open-source mathematical reasoning datasets. The collection encompasses the complete datasets of MetaMath (Yu et al. 2023), MMIQC (Liu et al. 2024), and Open-Platypus (Lee, Hunter, and Ruiz 2023), in addition to the training sets of GSM8K (Cobbe et al. 2021), MATH (Hendrycks et al. 2021), TAL-SCQ5K-EN[2], and MathInstruct (Yue et al. 2024).

Human Evaluation of Data Accuracy

To validate our methods, we conducted human evaluation on a random sample of 100 questions from our generated dataset. Specifically, the KPMATH-G subset achieved a correctness rate of 95%, with observed errors primarily in problem statements or logical reasoning rather than calculation errors. The KPMATH-M subset demonstrated an 81% correctness rate, with the criterion that a multi-question problem is considered correct if at least one response is accurate. This assessment yielded a high accuracy rate, confirming the reliability of our automated verification and filtering techniques.

Data Contamination Test

To mitigate data contamination risk in our benchmark, we used the method by Azerbayev et al. (2023) to scrutinize n-gram overlaps between our dataset and the Math and GSM8K test sets. We checked for 20-gram overlaps in questions and 30-gram in solutions. Our analysis revealed no overlaps in GSM8K and fewer overlaps in Math compared to its training set, with 102 hits in questions and 108 in solutions, against the training set's 181 and 144, respectively. A manual review confirmed these hits were due to recurring problem contexts or reasoning steps, not exact duplicates, indicating a low risk of contamination for KPMath.

[2]https://github.com/math-eval/TAL-SCQ5K

Experiment

Implementation Details

In our supervised fine-tuning (SFT) experiments, we employed chat message templates to transform question-answer pairs into the format: "User: {question}\nEnclose the final answer using \boxed{}.\n\nAssistant: {answer}". We utilized the LLaMa-Factory repository (Zheng et al. 2024) to fine-tune the models for 3 epochs across all experiments. We adopted a linear learning rate schedule with a 3% warm-up ratio. The maximum learning rate is 1e-5, except for DeepSeek-Math, which is 5e-5. We trained all models with BFloat16 numerical format, *DeepSpeed ZeRO Stage3* (Rajbhandari et al. 2021) and *Flash-Attention 2* (Dao 2023). For evaluation, we adopted the same template in SFT to prompt all questions. We employed greedy decoding with a maximum sequence length of 2,048 tokens.

Evaluation and Metrics

We evaluate our fine-tuned models on GSM8k (Cobbe et al. 2021) and MATH (Hendrycks et al. 2021), along with 4 out-of-distribution datasets, namely SVAMP (Patel, Bhattamishra, and Goyal 2021), ASDIV (Miao, Liang, and Su 2021), TabMWP (Lu et al. 2022), MAWPS (Koncel-Kedziorski et al. 2016). We utilize an enhanced version of the script from Gou et al. (2024b) to extract answers, parse expressions, and compare the equivalency of the answers. Additionally, we test the Hungarian Exam, adhering to the evaluation methodology proposed by Paster (2023), which segments the exam into 33 challenging problems suitable for model processing. These problems require manual answer verification. We report the zero-shot PASS@1 accuracy in all experiments.

Baselines

We present results from a range of state-of-the-art (SoTA) proprietary LLMs, including OpenAI's GPT-4 (OpenAI 2023), ChatGPT (gpt-3.5-turbo), Google's PaLM-2 (Anil et al. 2023), and Anthropic's Claude-2 (Anthropic 2023). Regarding open-source models, we consider base models such as Llama 2 (Touvron et al. 2023), Llama 3 (Dubey et al. 2024), DeepSeekMath (Shao et al. 2024), Mistral(Jiang et al. 2023), Llemma (Azerbayev et al. 2023), and Qwen1.5(Bai et al. 2023). Supervised Fine-Tuning (SFT) uses CoT rationales from the original GSM8k and MATH training sets (15K samples) for fine-tuning. We also showcase the performance of advanced models using SFT or RLHF on various mathematical reasoning datasets, including MAmmoTH (Yue et al. 2024), WizardMath (Luo et al. 2023a), Platypus-2 (Lee, Hunter, and Ruiz 2023), MetaMath (Yu et al. 2023) and MMIQC (Liu et al. 2024).

Model	Base	Size	ZS	GSM8k	MATH	SVAMP	TabMWP	ASDiv	MAWPS	AVG
Proprietary Models										
GPT-4 (0613)	-	-	✗	92.0	42.5	93.1	67.1	91.3	97.6	80.6
ChatGPT (gpt-3.5-turbo)	-	-	✗	80.8	35.5	83.0	69.1	87.3	94.6	75.1
Claude-2	-	-	✗	85.2	32.5	-	-	-	-	-
PaLM-2	-	540B	✗	80.7	34.3	-	-	-	-	-
Open-Source Models										
Llama-2	-	7B	✗	13.3	4.1	38.0	31.1	50.7	60.9	33.0
Llama-2 SFT	-	7B	✓	41.3	7.2	31.9	27.8	47.4	60.0	35.9
Platypus-2	Llama-2	7B	✗	14.4	5.4	36.7	26.5	47.9	58.4	31.6
MAmmoTH	Llama-2	7B	✓	45.9	7.3	48.7	28.9	62.3	74.8	44.7
WizardMath	Llama-2	7B	✓	54.9	10.7	57.3	38.1	59.1	73.7	49.0
MetaMath	Llama-2	7B	✓	66.6	20.7	68.8	43.8	72.5	86.9	59.9
KPMath-Plus	Llama-2	7B	✓	**75.6**	**34.0**	**73.1**	**51.7**	**82.1**	**92.9**	**68.2** (+35.2)
Llemma	-	7B	✗	40.4	18.9	56.5	49.0	68.7	83.3	45.8
KPMath-Plus	Llemma	7B	✓	**76.7**	**41.9**	**77.5**	**67.8**	**84.1**	**93.2**	**73.5** (+27.7)
Mistral	-	7B	✗	42.9	12.9	65.1	55.6	68.4	86.8	55.3
MAmmoTH	Mistral	7B	✓	52.7	14.5	54.1	49.1	64.9	77.5	52.1
MMIQC	Mistral	7B	✓	74.8	36.0	73.1	62.5	81.9	90.5	69.8
MetaMath	Mistral	7B	✓	77.8	29.0	78.6	64.7	81.1	93.4	70.8
KPMath-Plus	Mistral	7B	✓	**82.1**	**46.8**	**76.4**	**66.4**	**86.7**	**94.2**	**75.4** (+20.1)
DeepSeekMath	-	7B	✓	63.3	32.3	73.2	68.6	82.9	92.4	68.8
KPMath-Plus	DSMath	7B	✓	**83.9**	**48.8**	**81.5**	**78.7**	**88.9**	**94.8**	**79.4** (+10.6)
Llama-3	-	8B	✓	55.6	18.0	69.2	67.4	72.6	91.0	62.3
KPMath-Plus	Llama-3	8B	✓	**83.2**	**46.5**	**80.3**	**71.2**	**87.0**	**94.4**	**77.2** (+14.9)
Llama-2	-	13B	✗	24.3	6.3	43.1	39.5	56.3	70.4	36.2
Llama-2 SFT	-	13B	✓	51.1	9.2	46.3	35.8	58.6	75.0	42.6
Platypus-2	Llama-2	13B	✗	23.7	7.1	50.7	45.3	55.1	69.6	38.0
MAmmoTH	Llama-2	13B	✓	49.6	9.9	49.6	40.5	60.0	73.4	47.2
WizardMath	Llama-2	13B	✓	63.9	14.0	64.3	46.7	65.8	79.7	51.8
MetaMath	Llama-2	13B	✓	71.0	23.2	71.9	52.8	75.7	87.0	63.6
KPMath-Plus	Llama-2	13B	✓	**81.6**	**41.0**	**76.7**	**63.9**	**83.2**	**92.3**	**73.1** (+36.9)
Llemma	-	34B	✗	55.4	24.4	68.0	57.2	75.9	90.5	61.9
MMIQC	Llemma	34B	✓	79.2	38.7	80.4	70.1	85.0	94.0	74.6
KPMath-Plus	Llemma	34B	✓	**82.4**	**48.6**	**81.2**	**71.9**	**87.5**	**94.5**	**77.7** (+15.8)
Llama-2	-	70B	✗	57.8	14.4	73.6	57.5	76.0	92.4	58.2
Llama-2 SFT	-	70B	✓	69.3	14.9	64.0	53.0	71.3	84.8	56.6
Platypus-2	Llama-2	70B	✗	45.9	15.0	74.3	47.3	72.7	91.1	53.0
WizardMath	Llama-2	70B	✓	81.6	22.7	80.0	49.8	76.2	86.2	63.8
MetaMath	Llama-2	70B	✓	82.0	27.2	85.8	63.4	84.0	95.4	73.0
MAmmoTH	Llama-2	70B	✓	65.1	14.6	60.1	38.2	70.2	80.3	54.8
KPMath-Plus	Llama-2	70B	✓	**87.4**	**48.6**	**81.2**	**75.1**	**89.0**	**95.4**	**79.4** (+21.2)
Qwen1.5	-	72B	✗	77.6	38.2	82.5	52.0	85.1	95.9	71.9
KPMath-Plus	Qwen1.5	72B	✓	**87.0**	**58.3**	**82.1**	**76.7**	**89.2**	**95.5**	**81.5** (+9.6)

Table 1: Results on six mathematical reasoning tasks. The results of our model are bolded. ZS: Zero-shot inference without demonstrations. Vanilla models are tested with CoT.

Main Results

Table 1 presents the results on six widely-used mathematical benchmarks, highlighting several key observations:

Our models achieve state-of-the-art results in models with parameters ranging from 7B to 72B. Specifically, the KPMath-Plus-Qwen1.5-72B model achieves accuracies of 87.0% on GSM8K, and 58.3% on MATH, culminating in a promising average of 81.5% across six math reasoning datasets. This performance exceeds that of all competitors and even the best commercial models like GPT-4.

KPMath-Plus significantly boosts mathematical reasoning performance across a range of model architectures and sizes, including the latest SoTA models like Llama 3, DeepSeek-Math, and Qwen, with average accuracy gains ranging from 10.6% to 36.9%. Additionally, it delivers greater performance increases than other datasets with the same base models.

Although only GSM8K and MATH were used as seed data, KPMath-Plus also improves performance on OOD benchmarks. Additionally, Figure 6 displays the Hungarian Exam Score versus GSM8K Performance of various models. Notably, KPMath-Plus-Mistral-7B trails only GPT-4 (OpenAI 2023) and Grok-1 (xAI 2023). Compared to other fine-tuned

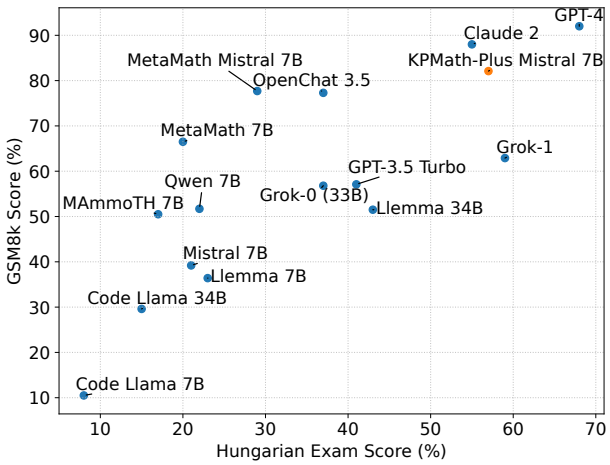

Figure 6: Hungarian Exam vs GSM8K Performance.

Key Points	Sampling Method	Retention Ratio
w/o	Random	27.4%
w/o	TCPM-based	30.1%
w/	Random	39.2%
w/	TCPM-based	**50.8**%

Table 2: Retention ratios of synthesized questions with different configurations.

models, our models demonstrate a well-balanced performance between the two test sets, suggesting they do not overfit the seed data.

Ablation Study

Knowledge Construction To ensure high data quality and conserve API resources, we conducted preliminary experiments with various methods before proceeding with large-scale data synthesis. We generated a set of 500 new questions across three scenarios, applying a stringent acceptance threshold of 0.85. The retention rates for questions synthesized using different configurations are summarized in Table 2. This study reveals that key points significantly enhance question quality and coverage of essential knowledge. Conversely, without these annotations, the synthesized questions notably lack in quality and relevant content. Furthermore, TCPM-based sampling improves the correlation among provided key points, resulting in more coherent and diverse questions.

Training Data Components We conducted an ablation study with the KPMath-Plus data components on the Mistral-7B model, training over 3 epochs. Results in Table 3 indicate that integrating KPMath-G, derived from the GSM8K dataset, enhances performance on GSM8K by 5% compared to training solely on MathMix. Improvements extend to SVAMP, ASDiv, and MAWPS, while a slight performance decline in MATH and TabMWP is observed, potentially due to their higher complexity. Moreover, combining KPMath-M, based on the MATH dataset, with MixMath consistently increases scores by over 1% across all datasets. Merging KPMath-G and KPMath-M significantly boosts overall performance,

Data	GSM8K	MATH	OOD
MixMath	75.7	43.3	**77.2**
MixMath + KPMath-G	80.7	43.0	**78.9**
MixMath + KPMath-M	77.0	45.9	**78.4**
KPMath-Plus	**82.1** (+6.4)	**46.8** (+3.5)	**80.9** (+3.7)

Table 3: Performance comparison of training with different data components. OOD: average accuracy of SVAMP, TabMWP, ASDiv, and MAWPS.

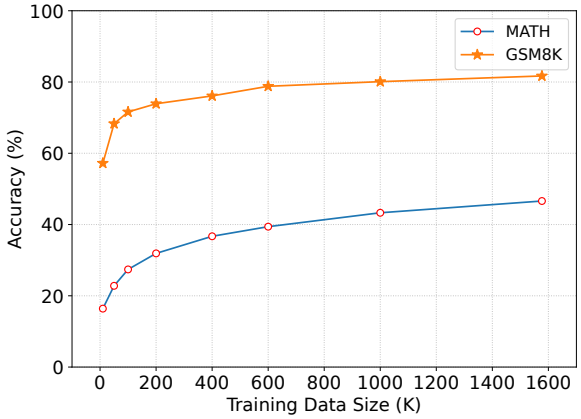

Figure 7: Performance of KPMath-Plus-Mistral-7B across various training data size.

with gains of 6.4% on GSM8K and 3.5% on MATH, averaging a 4.1% improvement, illustrating the comprehensive benefits of our synthesized data within KPMath-Plus for mathematical reasoning.

Training Data Size We also investigated the impact of training data size on the KPMath-Plus-Mistral-7B model's performance. As demonstrated in Figure 7, model performance exhibits a logarithmic increase with the expansion of training data. The model achieves impressive results with small data size and maintains a steady growth trend. This study underlines the exceptional quality of our data and establishes a clear linkage between training data size and model performance, particularly in tackling complex tasks. In our future work, we aim to further explore larger and higher-quality datasets to continue improving model performance.

Conclusion

In this paper, we propose a new data synthesis paradigm that is focused on the generation of large-scale, high-quality, symbolically-driven training datasets. Leveraging this paradigm, we have developed an extensive synthetic dataset tailored for mathematical reasoning. By utilizing this data set, our fine-tuned model achieved excellent performance in multiple data sets including MATH and GSM8K, and the performance exceeded all 7B to 72B competitors. Our research underscores the effectiveness of integrating key points in data synthesis and implementing automated verification methods for both questions and answers.

References

An, S.; Ma, Z.; Lin, Z.; Zheng, N.; Lou, J.-G.; and Chen, W. 2023. Learning From Mistakes Makes LLM Better Reasoner. *arXiv preprint arXiv:2310.20689*.

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

Anthropic. 2023. Model card and evaluations for claude models. Technical report, Anthropic.

Azerbayev, Z.; Schoelkopf, H.; Paster, K.; Santos, M. D.; McAleer, S.; Jiang, A. Q.; Deng, J.; Biderman, S.; and Welleck, S. 2023. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.10631*.

Bai, J.; Bai, S.; Chu, Y.; Cui, Z.; Dang, K.; Deng, X.; Fan, Y.; Ge, W.; Han, Y.; Huang, F.; et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Bi, X.; Chen, D.; Chen, G.; Chen, S.; Dai, D.; Deng, C.; Ding, H.; Dong, K.; Du, Q.; Fu, Z.; et al. 2024. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism. *arXiv preprint arXiv:2401.02954*.

Cen, J.; Wu, C.; Liu, X.; Yin, S.; Pei, Y.; Yang, J.; Chen, Q.; Duan, N.; and Zhang, J. 2024. Using Left and Right Brains Together: Towards Vision and Language Planning. *arXiv preprint arXiv:2402.10534*.

Chen, J.; Pan, X.; Yu, D.; Song, K.; Wang, X.; Yu, D.; and Chen, J. 2023. Skills-in-context prompting: Unlocking compositionality in large language models. *arXiv preprint arXiv:2308.00304*.

Chen, W.; Ma, X.; Wang, X.; and Cohen, W. W. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Chia, Y. K.; Chen, G.; Tuan, L. A.; Poria, S.; and Bing, L. 2023. Contrastive chain-of-thought prompting. *arXiv preprint arXiv:2311.09277*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

Dao, T. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Du, Y.; Guo, H.; Zhou, K.; Zhao, W. X.; Wang, J.; Wang, C.; Cai, M.; Song, R.; and Wen, J.-R. 2023. What makes for good visual instructions? synthesizing complex visual reasoning instructions for visual instruction tuning. *arXiv preprint arXiv:2311.01487*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gao, L.; Madaan, A.; Zhou, S.; Alon, U.; Liu, P.; Yang, Y.; Callan, J.; and Neubig, G. 2022. PAL: Program-aided Language Models. *arXiv preprint arXiv:2211.10435*.

Gou, Z.; Shao, Z.; Gong, Y.; yelong shen; Yang, Y.; Duan, N.; and Chen, W. 2024a. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. In *ICLR*.

Gou, Z.; Shao, Z.; Gong, Y.; yelong shen; Yang, Y.; Huang, M.; Duan, N.; and Chen, W. 2024b. ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving. In *ICLR*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. *NeurIPS*.

Huang, Y.; Lin, X.; Liu, Z.; Cao, Q.; Xin, H.; Wang, H.; Li, Z.; Song, L.; and Liang, X. 2024. MUSTARD: Mastering Uniform Synthesis of Theorem and Proof Data. *arXiv preprint arXiv:2402.08957*.

Huang, Y.; Lin, Z.; Liu, X.; Gong, Y.; Lu, S.; Lei, F.; Liang, Y.; Shen, Y.; Lin, C.; Duan, N.; et al. 2023. Competition-level problems are effective llm evaluators. *arXiv preprint arXiv:2312.02143*.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; and Hajishirzi, H. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, 1152–1157.

Lee, A. N.; Hunter, C. J.; and Ruiz, N. 2023. Platypus: Quick, Cheap, and Powerful Refinement of LLMs. *arXiv preprint arxiv:2308.07317*.

Lei, F.; Liu, Q.; Huang, Y.; He, S.; Zhao, J.; and Liu, K. 2023. S3Eval: A Synthetic, Scalable, Systematic Evaluation Suite for Large Language Models. *arXiv preprint arXiv:2310.15147*.

Lewkowycz, A.; Andreassen, A. J.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V. V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; Wu, Y.; Neyshabur, B.; Gur-Ari, G.; and Misra, V. 2022. Solving Quantitative Reasoning Problems with Language Models. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Li, H.; Dong, Q.; Tang, Z.; Wang, C.; Zhang, X.; Huang, H.; Huang, S.; Huang, X.; Huang, Z.; Zhang, D.; et al. 2024. Synthetic Data (Almost) from Scratch: Generalized Instruction Tuning for Language Models. *arXiv preprint arXiv:2402.13064*.

Li, Z.-Z.; Zhang, M.-L.; Yin, F.; and Liu, C.-L. 2023. LANS: A layout-aware neural solver for plane geometry problem. *arXiv preprint arXiv:2311.16476*.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*.

Liu, H.; Zhang, Y.; Luo, Y.; and Yao, A. C. 2024. Augmenting Math Word Problems via Iterative Question Composing. In

ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models.

Lu, P.; Qiu, L.; Chang, K.-W.; Wu, Y. N.; Zhu, S.-C.; Rajpurohit, T.; Clark, P.; and Kalyan, A. 2022. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023a. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Luo, Z.; Xu, C.; Zhao, P.; Sun, Q.; Geng, X.; Hu, W.; Tao, C.; Ma, J.; Lin, Q.; and Jiang, D. 2023b. WizardCoder: Empowering Code Large Language Models with Evol-Instruct. *CoRR*, abs/2306.08568.

Luong, T. Q.; Zhang, X.; Jie, Z.; Sun, P.; Jin, X.; and Li, H. 2024. Reft: Reasoning with reinforced fine-tuning. *arXiv preprint arXiv:2401.08967*.

Miao, S.-Y.; Liang, C.-C.; and Su, K.-Y. 2021. A diverse corpus for evaluating and developing English math word problem solvers. *arXiv preprint arXiv:2106.15772*.

Mishra, S.; Finlayson, M.; Lu, P.; Tang, L.; Welleck, S.; Baral, C.; Rajpurohit, T.; Tafjord, O.; Sabharwal, A.; Clark, P.; et al. 2022. LILA: A Unified Benchmark for Mathematical Reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 5807–5832.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Paster, K. 2023. Testing Language Models on a Held-Out High School National Finals Exam.

Paster, K.; Santos, M. D.; Azerbayev, Z.; and Ba, J. 2023. Openwebmath: An open dataset of high-quality mathematical web text. *arXiv preprint arXiv:2310.06786*.

Patel, A.; Bhattamishra, S.; and Goyal, N. 2021. Are NLP models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Rajbhandari, S.; Ruwase, O.; Rasley, J.; Smith, S.; and He, Y. 2021. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–14.

Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. *arXiv preprint arXiv:2302.00618*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Zhang, M.; Li, Y.; Wu, Y.; and Guo, D. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Tang, R.; Han, X.; Jiang, X.; and Hu, X. 2023. Does synthetic data generation of llms help clinical text mining? *arXiv preprint arXiv:2303.04360*.

Taylor, R.; Kardas, M.; Cucurull, G.; Scialom, T.; Hartshorn, A.; Saravia, E.; Poulton, A.; Kerkez, V.; and Stojnic, R. 2022. Galactica: A Large Language Model for Science. arXiv:2211.09085.

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

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv e-prints*, arXiv–2307.

Trinh, T. H.; Wu, Y.; Le, Q. V.; He, H.; and Luong, T. 2024. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995): 476–482.

Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2023. Math-Shepherd: A Label-Free Step-by-Step Verifier for LLMs in Mathematical Reasoning. *arXiv preprint arXiv:2312.08935*.

Wang, Z.; Xia, R.; and Liu, P. 2023. Generative AI for Math: Part I–MathPile: A Billion-Token-Scale Pretraining Corpus for Math. *arXiv preprint arXiv:2312.17120*.

Wei, Y.; Wang, Z.; Liu, J.; Ding, Y.; and Zhang, L. 2023. Magicoder: Source code is all you need. *arXiv preprint arXiv:2312.02120*.

xAI. 2023. Open Release of Grok-1. Technical report, xAI.

Xu, C.; Sun, Q.; Zheng, K.; Geng, X.; Zhao, P.; Feng, J.; Tao, C.; and Jiang, D. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Yu, L.; Jiang, W.; Shi, H.; Yu, J.; Liu, Z.; Zhang, Y.; Kwok, J. T.; Li, Z.; Weller, A.; and Liu, W. 2023. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. *arXiv preprint arXiv:2309.12284*.

Yue, X.; Qu, X.; Zhang, G.; Fu, Y.; Huang, W.; Sun, H.; Su, Y.; and Chen, W. 2024. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. In *The Twelfth International Conference on Learning Representations*.

Zhang, B.; Zhou, K.; Wei, X.; Zhao, X.; Sha, J.; Wang, S.; and Wen, J.-R. 2024. Evaluating and improving tool-augmented computation-intensive math reasoning. *Advances in Neural Information Processing Systems*, 36.

Zhang, X.; Tian, C.; Yang, X.; Chen, L.; Li, Z.; and Petzold, L. R. 2023. Alpacare: Instruction-tuned large language models for medical application. *arXiv preprint arXiv:2310.14558*.

Zheng, C.; Liu, Z.; Xie, E.; Li, Z.; and Li, Y. 2023. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*.

Zheng, Y.; Zhang, R.; Zhang, J.; Ye, Y.; Luo, Z.; and Ma, Y. 2024. LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models. *arXiv preprint arXiv:2403.13372*.

Zhou, A.; Wang, K.; Lu, Z.; Shi, W.; Luo, S.; Qin, Z.; Lu, S.; Jia, A.; Song, L.; Zhan, M.; et al. 2023. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. *arXiv preprint arXiv:2308.07921*.