ControlMath: Controllable Data Generation Promotes Math Generalist Models

Nuo Chen[♠] Ning Wu[♣] Jianhui Chang[♢] Linjun Shou[♣] Jia Li[♠*]

[♠]Hong Kong University of Science and Technology (Guangzhou)
Hong Kong University of Science and Technology
[♣]Microsoft
[♢]China Telecom Cloud Computing Research Institute
nchen022@connect.ust.hk, changjh1@chinatelecom.cn, jialee@ust.hk

Abstract

Utilizing large language models (LLMs) for data augmentation has yielded encouraging results in mathematical reasoning. However, these approaches face constraints in problem diversity, potentially restricting them to in-domain/distribution data generation. To this end, we propose **ControlMath**, an iterative method involving an equation-generator module and two LLM-based agents. The module creates diverse equations, which the Problem-Crafter agent then transforms into math word problems. The Reverse-Agent filters and selects high-quality data, adhering to the "less is more" principle, achieving better results with fewer data points. This approach enables the generation of diverse math problems, not limited to specific domains or distributions. As a result, we collect ControlMathQA, which involves 190k math word problems. Extensive results prove that combining our dataset with in-domain datasets like GSM8K can help improve the model's mathematical ability to generalize, leading to improved performances both within and beyond specific domains.

1 Introduction

Currently, mathematical reasoning (Cobbe et al., 2021; Chen et al., 2023d; Zhou et al., 2022; Weng et al., 2022) is regarded as the one of the most challenging areas for current Large Language Models (LLMs). Typically, prompting-based approaches (Wei et al., 2022b,a; Wang et al., 2022) are common ways to improve the mathematical abilities of closed-source LLMs. These methods design different prompts for these LLMs to solve multi-step and complicated math problems, setting a high benchmark and demonstrating the potential of LLMs in tackling sophisticated mathematical problems.

Recently, the focus has shifted towards improving the capabilities of smaller, open-source LLMs

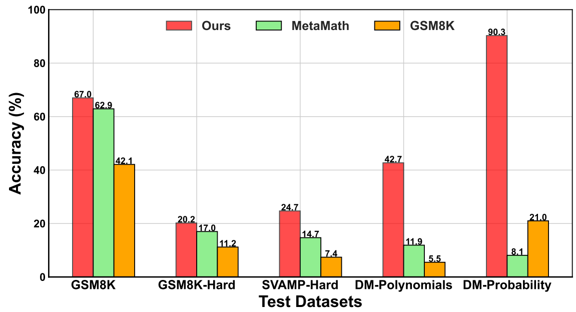

Figure 1: We train LLaMA 2-7B model (Touvron et al., 2023b) with different training corpus and present results in out-of-domain/distribution datasets: GSM8K-Hard (Gao et al., 2023), SVAMP-Hard (Chen et al., 2023b), DM-Polynomials and Probability subdatasets (Saxton et al., 2018).

through instruction-tuning (Yu et al., 2023; Chen et al., 2024a; Ouyang et al., 2022; Peng et al., 2023; Zhang et al., 2023; Longpre et al., 2023; Toshniwal et al., 2024). A significant advancement in this area is the use of data augmentation, using frontier LLMs like ChatGPT or GPT-4 to increase the size of available datasets to improve model performance (Yuan et al., 2023; Yu et al., 2023; Li et al., 2023a; Tang et al., 2024). For example, MetaMath (Yu et al., 2023) and MuggleMath (Li et al., 2023a) employ different augmented methods, such as question rephrasing, on the GSM8K (Cobbe et al., 2021) or MATH (Hendrycks et al., 2021) datasets to generate new problems. However, these methods depend heavily on seed questions from the training datasets, resulting in new samples that closely resemble the original ones. This lack of diversity can result in models overfitting to specific domains or data distributions, as the new problems maintain similar topics, reasoning steps, and numerical operations as the original ones. Thus, the models are effective within certain domains/distributions but less capable outside them (Li et al., 2023a).

Figure 1 showcases the effects of in-domain

[*] Corresponding author.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 12201–12217
November 12-16, 2024 ©2024 Association for Computational Linguistics

data augmentation using MetaMath on the GSM8K dataset. MetaMath achieves a significant performance boost on GSM8K, increasing accuracy by over 20%. However, when tested on out-of-domain datasets like SVAMP-Hard, DM-Polynomials, and DM-Probability, the model's performance drops sharply, sometimes even below the baseline trained only on GSM8K. Notably, on the GSM8K-Hard dataset, which only varies the distributions of the numbers, there is also a substantial decline in accuracy. This highlights that while MetaMath improves in-domain performance, it struggles to generalize to more diverse or challenging datasets.

Moreover, another critical oversight in previous approaches is the inclusion of low-quality or redundant samples in augmented datasets. Although methods like MetaMath have significantly expanded original datasets from 14k to 400k, concerns remain about the quality and utility of all generated samples. Some samples may be of low quality or redundant, failing to contribute meaningfully to the model's learning process and increasing useless training costs (Chen et al., 2023a). This highlights the need for an effective strategy to filter out ineffective training samples, a challenge still unaddressed in mathematical reasoning.

To address the above challenges, we propose a simple and iterative approach for controllable mathematical data generation, termed **ControlMath**. The core concept is *from equation to math word problem* and the essence of control lies in two primary perspectives: (1) Controllable equation generation: By specifying requirements such as reasoning steps, designed operators, and numerical ranges, we generate corresponding equations through an equation generation module. The frontier LLM-based Problem-Crafter Agent then creates mathematical word problems based on these equations, ensuring control over the diversity and distribution of the final set of problems. (2) Adaptively efficient data selection: This is achieved through a Problem Rewriter-Agent to ensure the effectiveness of the training samples. This agent rewrites the generated mathematical problems to create variations. Both the original and rewritten problems are fed to the LLM; if the LLM solves both correctly, it indicates that this generated problem does not contribute to further learning and can be discarded. This approach mimics human cognitive learning, where redundant information does not enhance learning efficiency. Filtering out redundant samples allows us to focus on the most beneficial data, enhancing overall training efficiency. The overview of our approach is presented in Figure 2.

By controlling the generation and selection of training samples, our method embodies the principle of "less is more", focusing on quality over quantity to enable stronger models with smaller datasets. To prove its effectiveness, we curate a dataset-ControlMathQA-comprising approximately 190k training samples. By seamlessly integrating this dataset with established GSM8K and MetaMath datasets, we embark on training LLMs. Our empirical results prove the resulting generalist LLMs clearly show significant performance improvements in both in-domain and out-of-domain datasets, as shown in Figure 1. In general, our contributions are summarized as follows:

- We propose a new data generation strategy, named ControlMath, for mathematical reasoning, which consists of controllable equation generation and an adaptively efficient data selection process.
- We collect ControlMathQA, a new training corpus that involves about 190k samples to help build math generalist models.
- We further investigate different variants of ControlMath to show its scaleability and generalization.

2 Related Works

Mathematical Reasoning with LLMs. In recent times, large language models (LLMs) (Brown et al., 2020; Hu et al., 2021; OpenAI, 2023; Touvron et al., 2023a; Huang et al., 2024; Chen et al., 2023c) have demonstrated remarkable abilities in handling complex mathematical reasoning tasks (MR) (Scao et al., 2022; Cobbe et al., 2021; Zhou et al., 2022; Weng et al., 2022; Chen et al., 2023e; Imani et al., 2023a). Two primary approaches are used: (1) Prompting-based methods (Imani et al., 2023b; Wang et al., 2022; Gao et al., 2023; Shi et al., 2022; Chen et al., 2023e), which employ diverse prompts to aid LLMs in solving mathematical problems. A notable example is Chain-of-Thought (CoT) (Wei et al., 2022b), which guides LLMs through step-by-step prompting. (2) Finetuning-based approaches (Li et al., 2023b; Yuan et al., 2023), which unlock the potential of open-source LLMs for mathematical reasoning through instruction-tuning, relying on effective downstream training data.

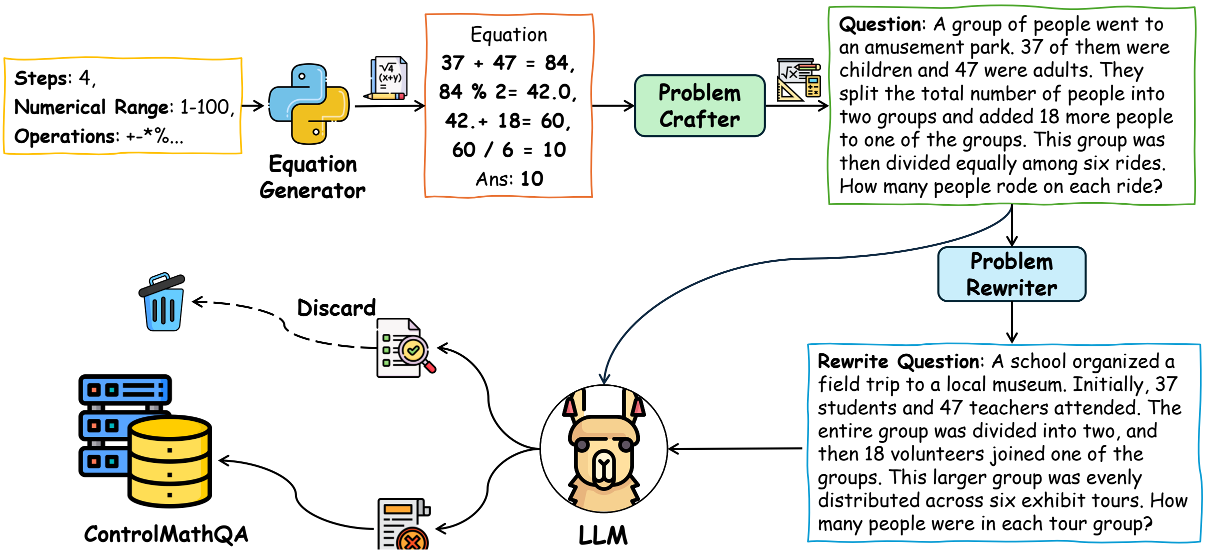

Figure 2: The overview of our ControlMath. Here, we present the example of generating multi-calculation math word problems.

Data Augmentation with LLMs. In the era of LLMs, data augmentation is regarded as a useful approach to improve smaller LLMs' performances in down-stream tasks. This approach often involves distilling outputs of stronger LLMs to generate the SFT datasets (Luo et al., 2023; Mitra et al., 2024; Yin et al., 2024; Yue et al., 2024; Meng et al., 2022). When applying data augmentation into math reasoning, there are two variants: query augmentation and response augmentation. Typical works towards query augmentation include MuggleMath (Li et al., 2023a), MetaMath (Yu et al., 2023) and MathScale (Tang et al., 2024), which use different strategies to obtain the new questions. Meanwhile, RFT (Yuan et al., 2023) and xRFT (Chen et al., 2023d) concentrate on response augmentation, which utilize rejection sampling to generate diverse reasoning paths for the same question. However, most of these approaches rely on seed questions in training sets, which pose challenges in generating diverse domain or distribution data.

3 ControlMath

The primary motivation behind ControlMath is to address the challenges of overfitting and inefficiency in training LLMs for mathematical reasoning tasks. To achieve this, ControlMath employs a three-step approach involving two specialized agents and one Python function module. In this section, we present the details of ControlMath.

3.1 Controllable Equation Generation

Our first step is the controllable generation of mathematical equations, driven by the need to ensure diversity in the following generated math word problems. Specifically, we employ an equation generation module to allow precise control over the characteristics of the equations, addressing the issue of homogeneous training samples that limit the model's ability to generalize.

In our implementation, the equation generation module is a Python function designed to produce equations based on specified parameters. We focus on generating three main types of equations: 1) **Multi-step Calculation**: These include basic arithmetic operations (addition, subtraction, multiplication, division), as well as more complex functions like trigonometric functions, roots, and exponents. For generating such equations, we specify the input parameters, including *steps*, *operators* and *numerical ranges*, to create diverse equations; 2) **Polynomials**: These equations involve terms with variables raised to various powers, often requiring multiple steps to simplify or solve. By generating polynomials of varying *degree* of the polynomial and the *coefficients*, we ensure that the model encounters a wide range of polynomial-related problems; 3) **Probability**: These equations involve calculating probabilities, which often require understanding combinations, permutations, and probability rules. We follow (Hendrycks et al., 2021) and set predefined topics: *a set of letters* or *sequence* as templates. These templates serve as the basis for generating diverse probability problems, ensuring the LLMs develop a strong foundation in statistical reasoning.

By customizing different inputs, we generate a wide variety of formulas that cover a broad spectrum of mathematical concepts and operations. This controlled generation ensures the foundation

You are a math expert. Given an equation list that contains multi-step mathematical formulas and its final answer. Your task is to create a math word problem and its reference solution based on the provided series of mathematical equations and the final answer.

`{few shot examples}`

Equation List: `{Equation}`
Final Answer: `{Answer}`

You should keep in mind that:
(1) Your produced math problem should be diverse and avoid using words like "mathematical", "calculation", etc.
(2) You should present equations in LaTeX format.

Table 1: Prompt Template for Problem-Crafter Agent.

for creating diverse mathematical word problems.

Problem-Crafter Agent. The second step involves translating the abstract equations into math word problems. A frointer LLM-based Problem-Crafter agent is used for this purpose, which generates *contextually rich problems* and their *reference solutions* from the generated equations. It ensures that problems vary in context and application while maintaining a strong connection to the underlying equations. We present the template for prompts in Table 1. To guide the agent in question formulation, we include several examples in the prompts, which are from manual interactions between Web GPT-4.

3.2 Efficient Data Selection: More data is not always helpful

Problem-Rewriter Agent. The third step focuses on selecting effective training data and eliminating redundant or ineffective samples. The motivation for this step is to enhance training efficiency by ensuring that only useful data is retained, thereby improving model performance without overfitting. The Problem-Rewriter agent is crucial for this step. After generating an initial problem by the Problem-Crafter, this agent rewrites the problems to create variations. Note that the rewriter agent is constrained to maintain the original numerical values of the problems, changing only the phrasing or topics, as shown in Appendix A.

Efficient Data Selection. Then, both the original and rewritten math word problems are presented to the smaller open-sourced LLM; if the LLM can solve both correctly, it means such problem does not contribute to further learning and can be discarded because model may have mastered this level of problems. However, if the model answers incorrectly to one of them, it indicates a genuine gap in understanding, making these problems valuable for further training. Such problem will be collected in our **ControlMathQA**. This process mimics human cognitive learning, where exposure to redundant information does not enhance understanding. By focusing on non-redundant, effective samples, we improve the overall quality and efficiency of the training data. To reduce variance, the temperature is set to 0 when sampling LLM's outputs.

Validation. In practice, we observe that sometimes the new problems generated by the Problem-Crafter Agent might not correctly match the corresponding formulas or the generated reference solutions might be inaccurate. To address these issues, we introduce a validation step using GPT-4 (`gpt-4-turbo`). For validation, we input the formula, the generated problem, and the corresponding solution into GPT-4. If GPT-4 identifies any mismatches—either the problem does not align with the formula, or the solution does not correctly solve the problem—we discard the problematic entry. This validation step ensures that only accurate and relevant problems are included in our dataset, further enhancing the quality of the training data. The prompts are presented in Appendix A.

Iterative Process. In theory, our approach can be iteratively applied indefinitely to continually generate needed corpus. However, due to resource constraints, we limit experimentation to **three iterations**. In the first iteration, we use Mistral-7B which has not undergone any fine-tuning. For the second and third iterations, we fine-tune the model using the data generated from the previous iterations. This iterative process allows us to progressively enhance the model's performance by continually refining the training data and focusing on the model's weaknesses identified in each round.

Why we need to rewrite the problem? The purpose of rewriting the problem is to ensure the robustness of the model's understanding. Several works (Chen et al., 2024b; Yu et al., 2023) have proved that simply rephrasing or changing the order of the question could confuse LLMs. Thus, simply testing the model on the original problem might not fully capture its comprehension, as the model could be overfitting to specific phrasings or patterns in the training data. By presenting both the original and the rewritten problem, we can more accurately assess whether the model truly understands the underlying concepts. We have also com-

Specifications **Generated Equation**	**Steps**: 4, **Operators**: [sqrt(), ×, %, +, -], **Numerical Range**: 1-1000 4 - 1 = 3, 3 * 16 = 48, sqrt(16) = 4, 30 % 6=5
Math Word Problem	Tom sets sail on his ship at 1 PM, traveling at a speed of 10 miles per hour. He reaches his destination at 4 PM. After spending some time there, he decides to return. However, due to different weather conditions, he sails back at a slightly slower speed that is the square root of the original speed. Using this information, can you determine how long it takes Tom to return to his starting point?
Specifications **Generated Equation**	**Degree**: 2, **Coefficients**: -100-100 -41*c - 16*c**2 + 18*c + 25*c
Math Word Problem	Express -41*c - 16*c**2 + 18*c + 25*c in the form q*c**2 + p*c + u and give p.
Specifications	**Templates**: A set of letters, **Unique characters**: 6, **Without Replacement**
Math Word Problem	What is probability of picking 1 k, 1 h, and 1 c when three letters picked without replacement from {c: 1, y: 1, e: 1, n: 1, k: 1, h: 2}?

Table 2: Several generated examples in our ControlMathQA, including input specifications, generated equation and math word problems. More cases could be seen in Appendix B.

plie a confusion matrix showing the proportions of cases where the model answers both questions correctly, both incorrectly, or one correctly and one incorrectly, detailed in the Appendix, Table 6.

Correlation with AdaBoost. Our approach is theoretically similar to AdaBoost (Freund et al., 1999), a machine learning algorithm that iteratively trains weak classifiers on the hardest-to-classify samples, reweighting data to focus on errors. Similarly, ControlMath focuses on generating and selecting the most challenging problems for the LLM, effectively "reweighting" the training data towards the areas where the model struggles the most. By collecting only the problems the model answers incorrectly, we ensure that subsequent training rounds address these weaknesses, analogous to how AdaBoost emphasizes misclassified samples to improve overall model performance. This similarity to AdaBoost gives ControlMath a theoretical base, showing that it has the potential to make LLMs more reliable in general mathematical reasoning.

3.3 Statistics

In each iteration, we generate 160k multi-step calculation formulas, 10k polynomials and 10k probability problems (See details in following Section 4.1). After data selection and GPT-4 validation process, we collect 110k math word problems in total. Inspired by (Yuan et al., 2023), response augmentation also contributes to the benefit of math reasoning LLMs, we also include this type augmentation in our ContronlMathQA, where we utilize `GPT-3.5-Turbo-0613` to generate answers for each question three times and remain the correct ones. As a result, we collect about **190k** question-answer pairs, where 155k for multi-calculation, 15k for polynomials and 20k for probability problems. Note, we decide to keep only original problems as we find including rewritten ones contributes limited improvements (Results in Appendix, Figure 7).

4 Experiments

In this section, we first present the details of our implementation, including data generation, training settings. Then we show the main results of different backbones with ControlMathQA.

4.1 Implementation

4.1.1 Data Generation

In equation generation (Section 3.1), we introduce several constraints to ensure the generated equations are both high in quality and diverse: (1) **Unique equations**: To secure robust samples, it is essential to keep each generated equation distinct; (2) **Diverse Distributions**: *a*) For *multi-step calculation formulas*, we set the range for steps from 2 to 9, covering most multi-step reasoning scenarios. The numerical range is divided into four tiers: 1-100, 100-k, 1k-10k, and 10k-1million. During each iteration, these ranges are randomly combined to produce 5,000 unique equation examples for each combing group, with operators including basic arithmetic (addition, subtraction, multiplication, division), square roots, and exponents. *b*) For *polynomials*, we set the degree range from 1 to 3 and coefficients from -100 to 100, producing polynomials of different complexities, ensuring the LLMs encounter a broad range of polynomial-related problems. Each iteration randomly samples 5,000 polynomial equations. *c*) Following Saxton et al. (2018), we consider two sampling settings: with or without replacement to generate *probability problems*. The sample space templates consist of a set of characters and a sequential sequence, where

In-Domain Training Datasets: GSM8K								
Models	In-Domain		Out-Of-Domain					**Avg.**
	GSM8K	GSM8K-Hard	SVAMP	SVAMP-Hard	MATH	Polynomials	Prob.	
LLaMA 2-7B	42.1	11.2	35.8	7.4	4.7	5.5	21.0	18.2
Ours	49.3	18.4	51.6	18.2	9.5	39.0	89.3	**39.3**
LLaMA 2-13B	49.7	13.5	50.9	14.7	6.0	5.5	22.0	23.2
Ours	57.1	16.6	63.1	20.4	11.2	41.2	90.7	**42.9**
Mistral-7B	51.5	16.1	37.8	11.3	7.3	4.5	44.9	24.8
Ours	59.1	19.7	55.7	19.1	12.4	42.1	94.9	**43.3**

In-Domain Training Datasets: MetaMath								
Models	In-Domain			Out-Of-Domain				**Avg.**
	GSM8K	GSM8K-Hard	MATH	SVAMP	SVAMP-Hard	Polynomials	Prob.	
LLaMA 2-7B	62.9	17.0	19.7	60.8	14.7	11.9	8.1	27.9
Ours	67.0	20.2	21.4	68.9	24.7	42.7	90.3	**47.9**
LLaMA 2-13B	66.9	18.7	22.4	65.7	17.9	14.0	9.3	30.7
Ours	70.0	22.5	24.1	72.4	26.5	43.4	90.1	**49.9**
Mistral-7B	67.6	21.8	22.7	66.5	19.8	20.8	9.1	32.6
Ours	71.4	24.5	26.1	71.2	28.7	50.9	94.7	**52.5**

Table 3: Model performances with different training datasets. Prob. refers to Probability. Ours denotes that we add our ControlMathQA with specific in-domain training datasets. More comprehensive evaluation of the whole Mathematics and MMLU Datasets can be seen in Appendix, Table 8.

the number of unique characters does not exceed 20. In each interaction, we randomly sample 5000 examples. Of note, this setting directly produces math word problems, eliminating the need for the problem-crafter agent, as an example in Table 2. We use *math*, *sympy*, *mathematics_dataset* libraries to generate equations with python.

In Section 3.2, both Problem-Crafter and Rewritten Agents are based on `gpt-4-turbo`. Key implementation details include setting the temperature to 0.9 to ensure diverse problem generation, max tokens to 2,000 to accommodate complex problem formulations, and top-p to 1.

4.1.2 Experimental Settings

Training Settings. The question-answer pairs in ControlMathQA are formatted in Alpaca-format (Taori et al., 2023). Experimentally, we select the LLaMA 2-7B,13B and Mistral-7B (Jiang et al., 2023) as backbone models. We use a batch size of 128, 512 max token length and train on the ControlMathQA dataset for 3 epochs using a learning rate of 2e-5 on NVIDIA A100 GPUs. To better illustrate the generalization of our ControlMathQA, we separately combine it with GSM8K and MetaMath in-domain training datasets to validate its effectiveness in in-domain and out-of-domain results.

Evaluation Settings. We evaluate different models in the following datasets: (1) **SVAMP** (Patel et al., 2021), an elementary-level math dataset with 1,000 test examples. (2) **SVAMP-Hard** (Chen et al., 2023b), which replaces numbers in SVAMP questions with values between 100k and 10M. (3) **GSM8K** (Cobbe et al., 2021), a dataset of 1,391 linguistically diverse grade school MWPs crafted by human writers. (4) **GSM8K-Hard** (Gao et al., 2023), an advanced version of GSM8K with larger numerical values to test LLM generalization. (5) The **Polynomials** and **Probability** subsets from Mathematics Datasets (Saxton et al., 2018), each containing 2,000 samples. (6) **MATH** (Hendrycks et al., 2021), which provides competition-level challenges across different mathematical domains.

4.2 Results

We evaluate ControlMathQA to answer the following questions: **Q1**: Can our dataset promote math generalist LLMs across various domains? **Q2**: Does our adaptively efficient data selection strategy help model achieve better results with less data?

RQ1: ControlMathQA promotes Math Generalist Models. Table 3 presents three backbones' performances with combining ControlMathQA and other in-domain training datasets, separately. Obviously, ours could help different LLMs exhibit exceptional performances across different domains. A notable example is that when training LLaMA 2-7B with ours and GSM8K, the resulting model attains impressive improvements of 15.8% and 68.3% on SVAMP and Probability datasets. Interestingly, though MetaMath could significantly boost the model performances in-domain bench-

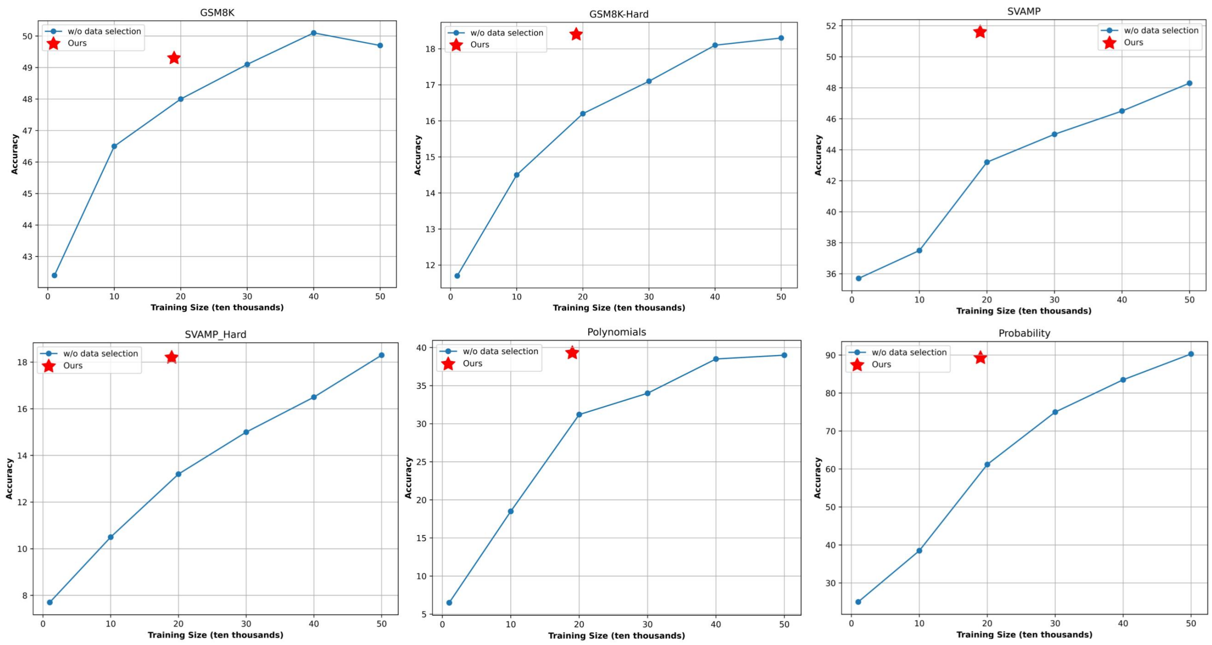

Figure 3: Here, we present the LLaMA 2-7B performances with different size corpus when we don't apply our efficient data selection strategy. Here, we train the model with ControlMathQA and GSM8K.

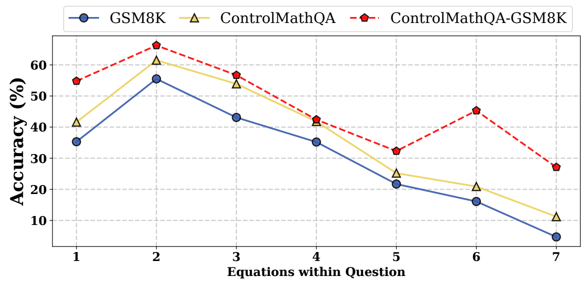

Figure 4: Here, we present the LLaMA 2-7B performances with different training corpus in GSM8K.

marks, it downgrades the baseline results in Probability problem solving. This highlights that overfitting in-domain helps little or even decreases out-of-domain mathematical capabilities. These results underscore the effectiveness of our approach in enhancing mathematical generalization.

RQ2: More data is not always useful. In this component, we explore whether our selection strategy can achieve higher training efficiency. Figure 3 shows extensive experiments using GSM8K as the in-domain training corpus without data selection. We can draw the several observations: 1) Scaling up the dataset size generally improves model performance, though the improvement diminishes with larger size. 2) With our data selection strategy, the model achieves much better performances vs. using the same data size without our data selection strategy. And mostly ours also achieve better performances when the model trained with 500k data size (except GSM8K). This highlights that more data is not always helpful and data quality is more important than data quantity.

Augmentation	SVAMP	SVAMP-H	MATH	Poly.	Prob.
-	35.8	7.4	4.7	5.5	21.0
w. Multi-step	47.5	15.9	6.4	9.3	21.2
w. Poly.	39.1	10.3	7.5	37.8	26.0
w. Prob.	37.5	9.7	7.3	10.1	88.1

Table 4: Ablations for different augmentation problems. Here, we train LLaMA 2-7B with our datasets and GSM8K. More results seen in Appendix, Table 7.

Ablations. Table 4 shows the ablations of different augmentation problem types in out-of-domain datasets. Obviously, each augmentation contributes to the improvements in different datasets. Multi-step calculation augmented problems lead to better results in SVAMP related datasets. Meanwhile, probability augmentation yields the highest improvements in probability-related tasks.

4.3 Discussion 1: Tailor ControlMath for specific datasets

Our method can focus on both generalization and adaptation to a targeted dataset. A trivial version of our method involves adaptively generating data based on poor performance on a targeted dataset. Here, we provide an example of how to augment the GSM8K dataset using our method:

First, we train a base SFT model (LLaMA 2-7B) using the GSM8K training set. Then, we test this

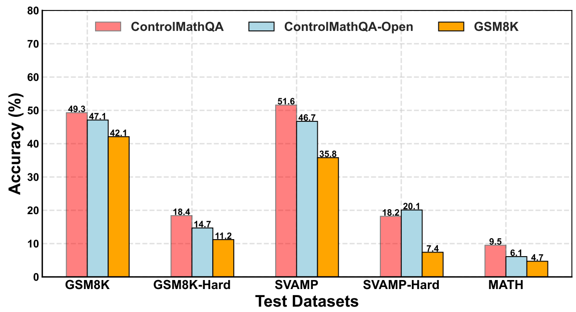

Figure 5: Here, we present the LLaMA 2-7B performances with training different corpus.

model on the test set, recording the error rates for problems of varying difficulty levels. The difficulty of a problem is primarily defined by the complexity of the equations involved. Using our ControlMath method, we generate new problems matching the difficulty distribution of the test set problems. We generate more data for problem types with higher error rates, proportional to their contribution to the overall error rate across different difficulty levels. In each iteration, we generate a total of 15,000 new problems and repeat this process three times to iteratively generate the final corpus, termed as **ControlMathQA-GSM8K.**

Figure 4 presents the accuracy when the model is trained with different corpus. We can find that after such tailored data augmentation, the resulting model significantly improves in all difficulty type questions. Moreover, the overall model performance increases from 42.1% to 55.3%, a better result compared with base ControlMathQA (49.3%). The results indicate that our ControlMath can be regarded as a plug-and-play module to adaptively generate training samples for targeted datasets.

4.4 Discussion 2: Cost-Effective ControlMath

Using GPT-4's API for data generation and validation is costly, resulting in more than 10k dollars. Intuitively, can we use smaller LLMs for data generation to reduce these costs while maintaining high quality? To explore this, we collect related 20k training samples from GPT-4 to train Mistral-7B as the two Problem-Crafter and Problem-Rewriter Agents. We also apply rejection sampling to conduct answer augmentation. With the trained Mistral-7B agents, we conduct data augmentation over nearly two months on 8xA100 GPUs, resulting in the collection of approximately 1 million training samples. We term the collected dataset as **ControlMathQA-Open**.

Interestingly, as shown in Figure 5, although ControlMathQA-Open consists of more than five times as much data as the base version, it performs worse across almost five datasets. This suggests that the quality of data generated by Mistral-7B is inferior to that of GPT-4. To investigate the reasons behind this, we calculate the perplexity (Marion et al., 2023; Wang et al., 2024) of both datasets using an unfine-tuned Llama 2-7B model. Figure 6 shows that ControlMathQA has a significantly lower perplexity compared to ControlMathQA-Open. This indicates that ControlMathQA is inherently easier to learn from, which might better enhance an LLM's problem-solving abilities (Yu et al., 2023). Moreover, we hypothesize that data augmentation by LLMs inherently resembles knowledge distillation, where a superior teacher model provides more effective problems and answers, thus better facilitating learning. We observe a similar phenomenon during answer augmentation: models trained on answer paths generated by GPT-4 or GPT-3.5 significantly outperform those trained on data from the LLaMA or Mistral series.

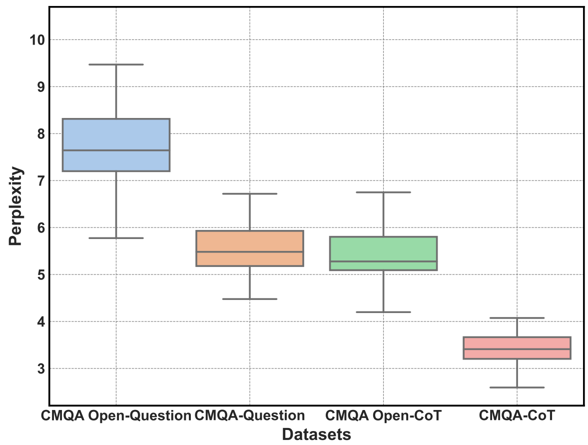

Figure 6: The perplexity of questions and CoT answers in different datasets. CMQA is short for ControlMathQA.

Dataset	gsm8k	MetaMath	Ours
Diversity Gain	-	0.09	0.41

Table 5: The diversity gain between MetaMath and ours

4.5 Discussion 3: Diversity Gains

we evaluate the diversity of our generated math word problems using the *diversity gain* (Bilmes, 2022) metric. This metric measures how much a new dataset contributes to the diversity of a base dataset.

Concretely, given a base dataset D_{base}: Comprises (N) samples, each represented as (q_i, r_i)

where q_i is the question, r_i is the CoT answer. And given a new dataset, D_{new}: Comprises (M) samples. Then: 1) For each sample (x_i) in D_{new}, find the sample (x_j) in D_{base} that is most similar based on feature extraction. 2) Calculate the squared Euclidean distance in the feature space between (x_i) and its nearest (x_j) using the formula: $(\min_{x_j \in D_{\text{base}}}(\|f(x_i) - f(x_j)\|^2))$. 3) Average these distances over all samples in D_{new}.

We use gsm8k as base dataset and OpenAI Embedding API text-embedding-ada-002 to extract sentence embeddings. The results are presented in Table 5. ControlMathQA dataset is much more diverse than gsm8k and MetaMath, covering more unseen scenarios.

5 Conclusion

In this paper, we propose a new data augmentation approach for mathematical reasoning, called ControlMath. The core of this approach lies into two lines: 1) It first utilizes an equation generation module to control the distributions of the generated equations. Then a Probelm-Crafter Agent generates the math word problems based on these equations. 2) It introduces a Problem-Rewriter Agent to help ensure the effectiveness of the generated samples, filtering out redundant ones. Our approach could generate diverse problems and focus on quality over quantity to enable stronger math generalist models with smaller data. The resulting dataset, ControlMathQA, could help LLMs obtain improvements in both in-domain and out-of-domain datasets. Future works include expanding the size of ControlMathQA and exploring other strategies to select more efficient data.

Limitation

In this paper, we focus on using large language models to generate training samples for mathematical reasoning, Some issues remain to be explored:

In practice, we observe that `gpt-4-turbo` may generate unrelated questions from the given equations, especially for complex ones involving 7 or 8 steps. We found that the error rate for such cases can be as high as 25-35%, leading to significant cost inefficiencies. A more efficient solution needs to be explored to address this issue.

Another limitation is that we have not yet conducted experiments on larger LLMs like LLaMA 2-30B or 70B models , primarily due to training resource constraints.

Additionally, due to the specificity of our method, we can generate large amounts of data tailored to specific datasets, potentially causing the model to overfit within that domain and thus achieve better performance. This could lead to unfair advantages in leaderboard rankings, though it ultimately depends on the practitioners' professional ethics.

Acknowledgement

This work was supported by National Key Research and Development Program of China Grant No. 2023YFF0725100 and Guangzhou-HKUST(GZ) Joint Funding Scheme 2023A03J0673

References

Jeff Bilmes. 2022. Submodularity in machine learning and artificial intelligence. *Preprint*, arXiv:2202.00132.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023a. Alpagasus: Training A better alpaca with fewer data. *CoRR*, abs/2307.08701.

Nuo Chen, Hongguang Li, Baoyuan Wang, and Jia Li. 2023b. From good to great: Improving math reasoning with tool-augmented interleaf prompting. *arXiv preprint arXiv:2401.05384*.

Nuo Chen, Yuhan Li, Jianheng Tang, and Jia Li. 2024a. Graphwiz: An instruction-following language model for graph computational problems. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 353–364, New York, NY, USA. Association for Computing Machinery.

Nuo Chen, Yan Wang, Haiyun Jiang, Deng Cai, Yuhan Li, Ziyang Chen, Longyue Wang, and Jia Li. 2023c. Large language models meet harry potter: A dataset for aligning dialogue agents with characters. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8506–8520, Singapore. Association for Computational Linguistics.

Nuo Chen, Zinan Zheng, Ning Wu, Linjun Shou, Ming Gong, Yangqiu Song, Dongmei Zhang, and Jia Li. 2023d. Breaking language barriers in multilingual mathematical reasoning: Insights and observations. *arXiv preprint arXiv:2310.20246*.

Xinyun Chen, Ryan A Chi, Xuezhi Wang, and Denny Zhou. 2024b. Premise order matters in reasoning with large language models. *arXiv preprint arXiv:2402.08939*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2023e. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Yoav Freund, Robert Schapire, and Naoki Abe. 1999. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023a. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023b. Mathprompter: Mathematical reasoning using large language models. In *ACL (industry)*, pages 37–42. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Chengpeng Li, Zheng Yuan, Guanting Dong, Keming Lu, Jiancan Wu, Chuanqi Tan, Xiang Wang, and Chang Zhou. 2023a. Query and response augmentation cannot help out-of-domain math reasoning generalization. *arXiv preprint arXiv:2310.05506*.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023b. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pretraining llms at scale. *Preprint*, arXiv:2309.04564.

Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating training data with language models: Towards zero-shot language understanding. *Advances in Neural Information Processing Systems*, 35:462–477.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

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

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2018. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. 2022. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*.

Zhengyang Tang, Xingxing Zhang, Benyou Wan, and Furu Wei. 2024. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *arXiv preprint arXiv:2402.10176*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2024. Making large language models better reasoners with alignment.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

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

Yixuan Weng, Minjun Zhu, Shizhu He, Kang Liu, and Jun Zhao. 2022. Large language models are reasoners with self-verification. *arXiv preprint arXiv:2212.09561*.

Shuo Yin, Weihao You, Zhilong Ji, Guoqiang Zhong, and Jinfeng Bai. 2024. Mumath-code: Combining tool-use large language models with multi-perspective data augmentation for mathematical reasoning. *Preprint*, arXiv:2405.07551.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *CoRR*, abs/2309.12284.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A Prompts

In this section, we show the prompts for Problem-Rewriter agents and training.

Prompts A.1: Prompts for Problem-Rewriter Agents

You are a language expert. Your task is to rewrite the given question to make it more diverse by rephrasing or changing the topic.
The following are several examples:

Question: Sam bought a dozen boxes, each with 30 highlighter pens inside, for $10 each box. He rearranged five of these boxes into packages of six highlighters each and sold them for $3 per package. He sold the rest of the highlighters separately at the rate of three pens for $2. How much profit did he make in total, in dollars?
New Question: Sam purchased 12 boxes, each containing 30 highlighter pens, at $10 per box. He repackaged five of these boxes into sets of six highlighters and sold them for $3 per set. He sold the remaining highlighters individually at a rate of three pens for $2. What is the total profit he made in dollars?

Question: A group of people went to an amusement park. 37 of them were children and 47 were adults. They split the total number of people into two groups and added 18 more people to one of the groups. This group was then divided equally among six rides. How many people rode on each ride?
New question: A school organized a field trip to a local museum. Initially, 37 students and 47 teachers attended. The entire group was divided into two, and then 18 volunteers joined one of the groups. This larger group was evenly distributed across six exhibit tours. How many people were in each tour group?

Question: Express -41*c - 16*c**2 + 18*c + 25*c in the form q*c**2 + p*c + u and give p
New question: Rewrite the polynomials -41*x - 16*x**2 + 18*x + 25*x in the form a*x**2 + b*x + c and determine the coefficient b.

Question: What is probability of picking 1 k, 1 h, and 1 c when three letters picked without replace- ment from {c: 1, y: 1, e: 1, n: 1, k: 1, h: 2}?
New question: What are the odds of drawing one k, one h, and one c from a collection that includes one each of c, y, e, n, k, and two h, if three letters are chosen sequentially without putting any back?

Question: `{Question}`

You should keep in mind that you can not change the numbers and operators in the question.

Prompts A.2: Prompts for GPT-4 Evaluation

You are a math expert. Given a question, a reference solution, and an equation, assess whether all components are logically consistent and correctly matched. Ensure that the question, solution, and equation are congruent. This means that: 1) The solution should correctly answer the question. 2) The equation should be applicable and necessary for deriving the solution. 3) All mathematical operations and logic used in the solution should be supported by the equation.

Question: `{Question}`
Reference Solution: `{Solution}`
Equation: `{Equation}`

If the question, reference solution and equation all align and math correctly, please output **True**. Otherwise, output **False**.

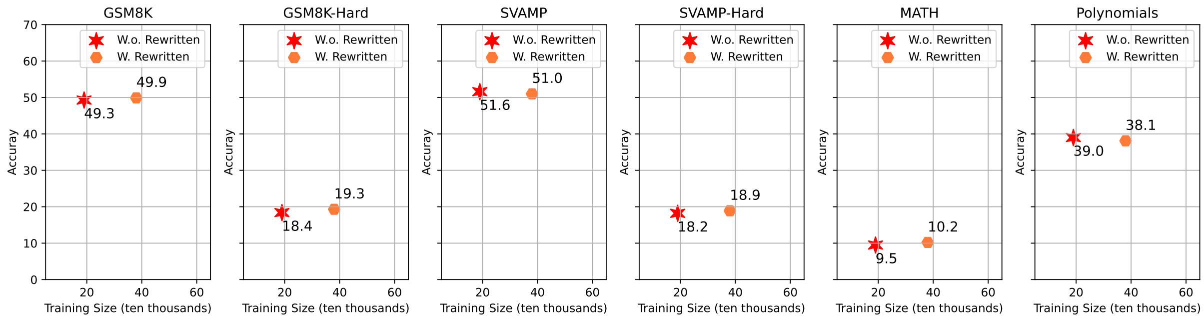

Figure 7: Here, we present the LLaMA 2-7B performances with including rewritten problems vs. without rewritten problems (Ours). Here, we train the model with ControlMathQA and GSM8K.

Prompts A.3: Prompts for Fine-tuning

Below is an instruction that describes a task. Write a response that appropriately completes the request.### Instruction: `{Query}` ### Response:

Problem Typs	Original Problem			
	Percentage (%)	Wrong	Correct	Total
Rewritten Problems	Wrong	33.1	13.4	46.5
	Correct	6.5	47	53.5
	Total	39.6	60.4	100

Table 6: Confusion matrix of the proportions between rewritten and original problems in total three iterative processes.

Augmentation	**GSM8K**	**GSM8K-Hard**	**SVAMP**	**SVAMP-H**	**MATH**	**Poly.**	**Prob.**
-	42.1	11.2	35.8	7.4	4.7	5.5	21.0
w. Multi-step	47.1	16.3	47.5	15.9	6.4	9.3	21.2
w. Poly.	45.1	14.3	39.1	10.3	7.5	37.8	26.0
w. Prob.	44.3	13.7	37.5	9.7	7.3	10.1	88.1

Table 7: Ablations for different augmentation problems. Here, we train LLaMA 2-7B with our datasets and GSM8K.

Methods	**Algebra**	**Arithmetic**	**Calculus**	**Numbers**	**Comparison**	**Polynomials**	**Probability**	**MMLU-Stem**	**Avg.**
GSM8K	9.5	46.5	3.3	35.4	70.1	5.5	21.0	29.7	27.6
MetaMath	13.2	61.2	8.8	60.1	77.5	11.9	8.1	42.6	35.5
Ours	**74.5**	**70.6**	**28.9**	**87.3**	**91.4**	**89.3**	**39.3**	**46.3**	**65.9**

Table 8: Evaluation results in Mathematics Dataset and MMLU-Stem. Here, the models are trained on LLaMA2-7B.

B Cases

Examples B.1: Several cases of our models solving Probability problems

Question: Three letters picked without replacement from y: 3, a: 1, p: 3, v: 2, c: 2, t: 2. Give prob of sequence cca.

Answer: To find the probability of picking the sequence "cca" without replacement from the set $\{y:3, a:1, p:3, v:2, c:2, t:2\}$, we calculate the probability step by step: The total number of letters is $3+1+3+2+2+2=13$. The probability of picking "c" first is $\frac{2}{13}$ since there are 2 "c"s out of 13 letters. After picking the first "c", there are now 12 letters left, and the probability of picking another "c" is $\frac{1}{12}$. After picking the second "c", there are now 11 letters left, and the probability of picking "a" is $\frac{1}{11}$. The probability of the sequence "cca" is the product of these probabilities:

$$\frac{2}{13} \times \frac{1}{12} \times \frac{1}{11} = \frac{1}{858}$$

Therefore, the probability of picking the sequence "cca" without replacement is $\frac{1}{858}$. Therefore, the answer is 1/858. ### 1/858.

Question: Two letters picked without replacement from wdwwwwwwrdwrrww. Give prob of sequence dr.

Answer: To find the probability of picking the sequence "dr" without replacement from the string "wdwwwwwwrdwrrww", we calculate as follows: The total number of letters is 14. There are 2 "d"s and 3 "r"s in the string. The probability of picking "d" first is $\frac{2}{14}$ since there are 2 "d"s out of 14 letters. After picking "d", there are now 13 letters left, and the probability of then picking "r" is $\frac{3}{13}$, as there are 3 "r"s remaining. The probability of the sequence "dr" is the product of these probabilities:

$$\frac{2}{14} \times \frac{3}{13} = \frac{3}{91}$$

Therefore, the probability of picking the sequence "dr" without replacement is $\frac{3}{91}$. Therefore, the answer is 3/91. ### 3/91.

Examples B.2: Several cases of our models solving Polynomial problems

Question: Express -37 - f**3 + 70 + 26 as b + w*f**2 + x*f + y*f**3 and give b.

Answer: Given equation: $-37 - f^3 + 70 + 26$ Simplify the constants: $-37 + 70 + 26 = 59$ Therefore, the equation can be rewritten as: $59 - f^3$ In the form $b + wf^2 + xf + yf^3$, we have:
$b = 59$
$w = 0$ (since there is no f^2 term)
$x = 0$ (since there is no f term)
$y = -1$ (coefficient of f^3)
Therefore, the answer is 59. ### 59.

Question: Frankie, a young scientist, is experimenting with temperature effects on a certain chemical compound. She measures the change in energy, denoted as E, when a sample is cooled. The energy change depends on the temperature in Fahrenheit, f, according to the equation:

$$E = -37 - f^3 + 70 + 26$$

Frankie records a temperature of -2 Fahrenheit during one of her experiments. Calculate the energy change for this temperature using the simplified equation.

Answer: Substituting $f = -2$ into the simplified equation:

$$E = -(-2)^3 + 59 = -(-8) + 59 = 8 + 59 = 67$$

The energy change at $f = -2$ Fahrenheit is 67 units. Therefore the answer is 67 ###67.

Examples B.3: Several cases of our models solving GSM8K/GSM8K-Hard problems

Question: Susan made 100 cookies for Christmas and was going to equally divide them between her 6 nephews. Before Susan could package them, her husband snuck 4 cookies for himself. How many cookies will each of Susan's nephews get?

Answer: Susan made 100 cookies in total.Her husband snuck 4 cookies for himself, so there are «100 - 4 = 96» cookies left.Susan wants to divide these cookies equally between her 6 nephews.To find out how many cookies each nephew will get, we divide the total number of cookies by the number of nephews: «96 / 6 = 16». Each of Susan's nephews will get 16 cookies. Therefore, the answer is 16. ### 16.

Question: Siobhan has 2 fewer jewels than Aaron. Aaron has 5 more jewels than half of Raymond's jewels. If Raymond has 973054 jewels, how many jewels does Siobhan have?

Answer: We know that Raymond has 973054 jewels.Half of Raymond's jewels is «973054/2 = 486527» jewels.Aaron has 5 more jewels than half of Raymond's jewels, so Aaron has «486527 + 5 = 486532» jewels.Siobhan has 2 fewer jewels than Aaron, so Siobhan has «486532 - 2 = 486530» jewels.Therefore, Siobhan has 486530 jewels. Therefore, the answer is 486530. ###486530.