Learn Beyond The Answer: Training Language Models with Reflection for Mathematical Reasoning

Zhihan Zhang[✉1†], **Zhenwen Liang**[1†], **Wenhao Yu**[2], **Dian Yu**[2], **Mengzhao Jia**[1†], **Dong Yu**[2], **Meng Jiang**[1]
[1]University of Notre Dame [2]Tencent AI Lab, Seattle
zzhang23@nd.edu

Abstract

Supervised fine-tuning enhances the problem-solving abilities of language models across various mathematical reasoning tasks. To maximize such benefits, existing research focuses on *broadening* the training set with various data augmentation techniques, which is effective for standard single-round question-answering settings. Our work introduces a novel technique aimed at cultivating a *deeper* understanding of the training problems at hand, enhancing performance not only in standard settings but also in more complex scenarios that require reflective thinking. Specifically, we propose **reflective augmentation**, a method that embeds problem reflection into each training instance. It trains the model to consider alternative perspectives and engage with abstractions and analogies, thereby fostering a thorough comprehension through reflective reasoning. Extensive experiments validate the achievement of our aim, underscoring the unique advantages of our method and its complementary nature relative to existing augmentation techniques.[1]

1 Introduction

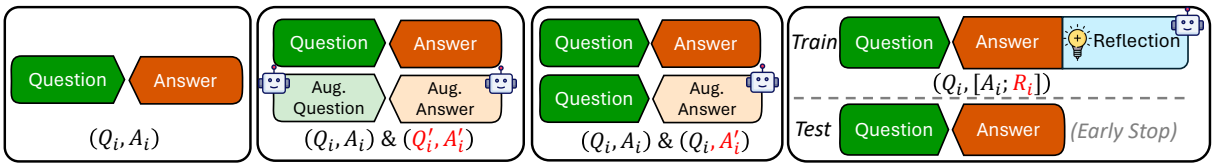
The ability to engage in step-by-step reasoning is pivotal for language models (LMs) to solve mathematical problems (Wei et al., 2022; Kojima et al., 2022). Supervised fine-tuning, particularly on data with detailed reasoning paths, effectively advances the problem-solving performance of LMs (Fu et al., 2023; Yue et al., 2023). To enlarge such benefits, most previous efforts focus on creating additional instances to augment model training (Luo et al., 2023a; Yu et al., 2024; Mitra et al., 2024; Li et al., 2024a). While these data expansion approaches allow LMs to handle a *broader* range of math problems by increasing the diversity of training data, stacking more training instances does not necessarily lead to a *deeper* understanding of each problem. Moreover, the scope of resulting models is confined to single-round question-answering (QA) settings that primarily require basic forward reasoning skills. Consequently, these methods provide limited benefits for more complex reflective reasoning scenarios that involve reviewing past steps for further reasoning, such as addressing follow-up questions, correcting errors, or leveraging external feedback (Liang et al., 2024; Wang et al., 2024a).

Similarly, the strategy in human learning is not always to practice an increasing number of problems (Rohrer and Taylor, 2006). Instead of merely memorizing superficial solutions to more problems, it can be more advantageous to gain a deep understanding of the existing problems (Semerci, 2005). *Reflection*, therefore, becomes an essential accompaniment to practice. Stacey et al. (1982) define reflection as "to review thoughtfully, consider alternatives and follow extensions", which encourages learners to contemplate their previous actions to engage in deeper reasoning, thereby fostering reflective thinking capabilities (Kagan et al., 1964; Anderson and Fincham, 2014).

Inspired by such human cognition, we propose a novel training strategy for LMs that integrates reflection into each math problem. Unlike traditional data expansion methods which operate on the instance dimension by adding more training examples (see Figures 1b & 1c), our approach targets a complementary direction, *i.e.*, the sequence dimension of the training data. We introduce *reflective augmentation* (**RefAug**), which appends a reflective section to the original answer of each training instance, advancing model learning beyond mere answer generation (see Figure 1d). Such a design not only strengthens the model's understanding of the associated knowledge and methodologies in training problems, but also maintains the inference efficiency as the model ceases generation before

[†] This work was done when Zhihan, Zhenwen, and Mengzhao were interns at Tencent AI Lab, Seattle.

[1]Code and data are available at https://github.com/ytyz1307zzh/RefAug.



(a) Standard data (b) Question augmentation (c) Answer augmentation (d) Reflective augmentation

Figure 1: Question augmentation creates new questions based on existing ones. Answer augmentation re-samples answers for each problem to increase diversity. Both methods expand the size of the training set. Reflective augmentation appends the original answer with a **reflective section**, which is complementary to traditional approaches. Corresponding training sequences are shown in an (input, output) format, where augmented parts are in red.

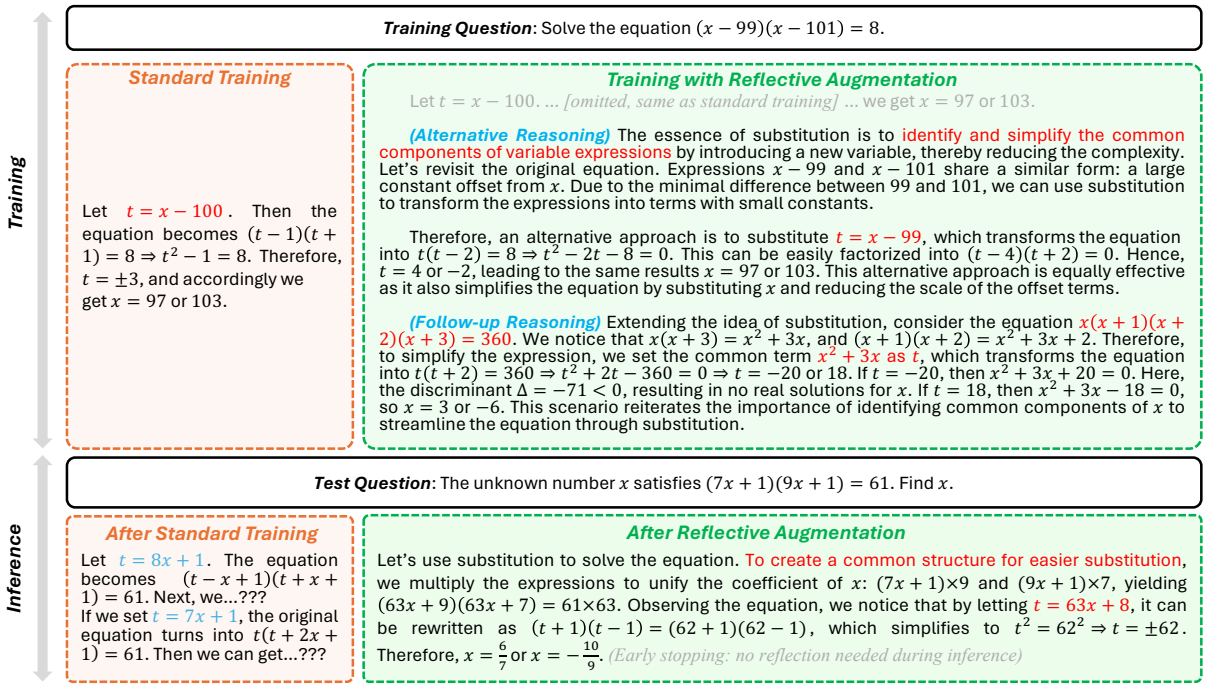


Figure 2: The model that learned the standard solution does not fully understand when and how to apply substitution when facing a different scenario. In contrast, the model trained with reflection on the substitution technique gains a deeper understanding of its principles, patterns, and its flexible application in new contexts.

decoding the reflective section during inference. Following the definition by Stacey et al. (1982), these reflective sections include two components: *alternative* and *follow-up reasoning*. For example, Figure 2 shows a scenario where the model struggles to apply the substitution technique in a different context if only rigidly transferring the pattern from the standard solution. In contrast, training the model to reflect on an equivalent substitution expression followed by devising a more challenging equation facilitates a deeper understanding of the principles and variations of the technique, thereby enabling flexible adaptation in new contexts.

Extensive experimentation on diverse math reasoning tasks reveals multiple benefits of RefAug: (1) It boosts the problem-solving performance of LMs in the standard single-round QA settings, yielding a +7.2 accuracy gain over direct fine-tuning. (2) It remarkably enhances the LMs' performance in multiple reflective math reasoning scenarios, where traditional data expansion methods fall short. (3) Its benefits are complementary to those of existing data expansion techniques, allowing for seamless integration that leads to even greater performance improvements.

2 Related Work

2.1 Data Augmentation for Math Reasoning

Due to the scarcity (Li et al., 2024a) and quality issues (Fan et al., 2024) of human-annotated data, data augmentation is a prevalent strategy in math reasoning tasks. Most research focused on creating additional training instances, typically using advanced LMs to minimize human effort. This include *question augmentation* which generates new questions from existing ones (Yu et al., 2024; Tang

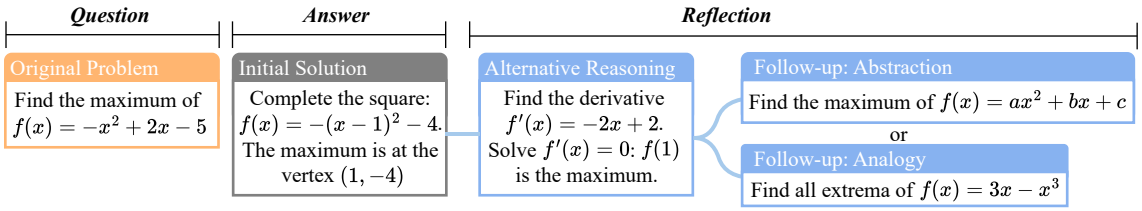


Figure 3: Relationship between the original instance and the reflective section. Either abstraction or analogy is annotated for each instance. Core ideas are shown but textual explanations (like those in Figure 2) are omitted.

et al., 2024; Li et al., 2024a; Liu et al., 2024; Huang et al., 2024b), and *answer augmentation* which resamples the answer for each question (Yuan et al., 2023; Li et al., 2023; Yu et al., 2024). Others also explored *answer refinement*, aiming to insert additional reasoning details (Anonymous, 2024) or to restructure answers for clearer reasoning paths (Fan et al., 2024). Not only is reflective augmentation complementary to existing approaches, but it also exhibits unique advantages in reflective reasoning scenarios, as we will show in §4.

Another branch of research augmented *code snippets* within problem solutions, which transforms text reasoning into code generation (Wang et al., 2023a; Gou et al., 2024; Lu et al., 2024). This method is effective for math problems but is typically considered a separate track since it uses external tools (*i.e.*, the code interpreter). Beyond supervised fine-tuning, some works augmented data for further preference optimization (Pang et al., 2024; Yuan et al., 2024), whereas we leave exploring reflective data in preference tuning for future work.

2.2 Reflection in LMs

Previous applications of reflection in LMs primarily focused on enabling LMs to rectify their own responses during inference (*i.e.*, self-reflect). Some works equipped the LM with external feedback, such as code execution or expert critiques (Shinn et al., 2023; Chen et al., 2024). Others prompted LMs to use only internal knowledge to correct answers (Madaan et al., 2023; Li et al., 2024b), though the effectiveness of this approach is under debate (Huang et al., 2024a). Some specific tasks (*e.g.*, math word problems) permit reverse verification, where the generated answer is used to rederive the question to confirm its correctness (Weng et al., 2023; Wu et al., 2024). These works demonstrate that reflection is a common aspect of language processing. However, RefAug explores augmenting reflective data for better training instead of answer refinement during inference. Unifying these approaches is a promising future study.

3 Approach

RefAug extends each training sequence with a reflective section that encourages the LM to reflect on its initial reasoning process to engage in further math reasoning. Figure 1 contrasts RefAug with traditional augmentation methods, and its detailed implementation is elaborated below.

Reflection Types Following the definition by Stacey et al. (1982) to "review thoughtfully, consider alternatives and follow extensions", we consider two types of reflection in composing the reflective section: *alternative reasoning* and *follow-up reasoning*.

Alternative reasoning involves thinking about the problem from different perspectives (Kagan et al., 1964; Wetzstein and Hacker, 2004). Therefore, besides the initial solution, we annotate an alternative approach that also effectively solves the problem. This helps the model master related methodologies and develop critical thinking skills.

Follow-up reasoning associates the initial solution to a broader class of problems (Silver, 1994; Lim et al., 2020). To fit various contexts, we consider two options: *abstraction* and *analogy*. Abstraction refers to creating a generalized form of the original problem, thereby encouraging the model to reduce dependency on specific numerical values. Analogy challenges the model in applying methodologies of solving the original problem to a more complex situation. Learning to design follow-up scenarios enables the model to understand the associated math concepts and principles better and apply them flexibly in new contexts. The relationship between the initial instance and components of the reflective section is illustrated in Figure 3.

Data Annotation Following a common approach (Li et al., 2023; Yu et al., 2024; Li et al., 2024a), we employ an expert LM, GPT-4-*turbo*, to annotate the reflective sections for high-quality rea-

soning paths and minimal human effort[2]. This entails reviewing the original problem and solution to generate a section consisting of the aforementioned two types of reflective reasoning. We prompt the expert model to choose between abstraction and analogy in follow-up reasoning based on the problem context. Figure 2 shows an annotated example with alternative reasoning and follow-up analogy, and the full annotation prompt is in Appendix E.

Training & Inference During training, given a math question as input, we include the reflective section in the training output immediately following the initial answer, starting with a `Reflection:` prefix. Thus, the training objective is to learn $\mathcal{P}([a; r]|q)$, where $[;]$ denotes sequence concatenation. Loss is calculated on tokens from both the initial answer and the reflective section. The format of the whole training sequence is detailed in Appendix D.

During inference, the generation early stops upon delivering the answer to the input question and ignores the reflective section, as shown in Figures 1-2. This is achieved by using `Reflection:` as a termination string during model generation.

4 Experiments

We test RefAug in a variety of mathematical tasks that cover both standard single-round QA and reflective reasoning scenarios. We mainly evaluate two aspects: **the influence of RefAug on LMs' math reasoning abilities** and **its interaction with existing augmentation techniques**. Besides, we extend our approach to code generation tasks and perform comprehensive analyses.

4.1 Standard Math Reasoning

4.1.1 Settings

Standard math reasoning tasks follow a single-round QA format. Following a popular approach, we use the training sets of GSM8k (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b). We additionally include out-of-distribution test sets from MAWPS (Koncel-Kedziorski et al., 2016), Mathematics (Davies et al., 2021), SVAMP (Patel et al., 2021), plus the math subsets of MMLU (Hendrycks et al., 2021a) and SAT (Zhong et al., 2023). We mainly experiment with two LMs known for superior reasoning performance: Mistral-7B (Jiang et al., 2023a) and Gemma-7B (Mesnard et al., 2024), and have also tested LLaMA-3-8B (Meta, 2024) in Appendix A.2. Models are trained for 3 epochs with batch size 128. The learning rate peaks at 1e-5 with a 3% warmup period followed by linear decay. Greedy decoding is applied during inference. Additional details of datasets and training settings are in Appendix B.1.

4.1.2 Existing Training Methods

- **Standard Fine-tuning** (Figure 1a): Utilizes original problem solutions from GSM8k and MATH, each containing a chain-of-thought reasoning process before reaching the final prediction.
- **Question Augmentation** (**Q-Aug**, Figure 1b): Involves training on both original and GPT-augmented questions. We adopt the augmentation prompt from Li et al. (2024a), detailed in Appendix C. We also explore **Q-Aug + RefAug** by applying RefAug to all questions after Q-Aug, and **Q-Aug×2** by adding a second augmentation round to further expand the dataset.
- **Answer Augmentation** (**A-Aug**, Figure 1c): Resamples the solution for each problem using GPT-4-*turbo*, following the approach of Yu et al. (2024). We also explore its combination with Q-Aug (**A-Aug + Q-Aug**), RefAug (**A-Aug + RefAug**), and another round of A-Aug (**A-Aug×2**).
- **MetaMath Augmentation:** MetaMath (Yu et al., 2024) creates a training set of 400K instances using various augmentation techniques. Due to budget constraints, we examine the following subsets: (1) A uniformly sampled 40K subset (**MetaMath$_{40k}$**), which we augment with RefAug to compare against an 80K sample (**MetaMath$_{80k}$**); (2) The entire 400K dataset, of which 40K instances are augmented with RefAug (**MetaMath$_{400k}$+RefAug$_{40k}$**), to compete with the public MetaMath checkpoint; (3) A one-epoch continual training (**CT**) from the public checkpoint on the same dataset as (2).

The augmentation prompt for Q-Aug and A-Aug, along with the sampling strategy on MetaMath can be found in Appendix C.

4.1.3 Results

Table 1 lists the QA accuracy of fine-tuned LMs. We summarize several findings on RefAug:

Enhancement in Single-Round Math Reasoning: RefAug boosts model performance across both in-distribution and out-of-distribution tasks, outscoring the direct fine-tuning approach by +7.2 across two base LMs. As the reflective section is

[2]We also tried LLaMA-3-70B for data annotation in Appendix A.1 but its performance lags behind GPT-4-*turbo*.

Model	Training Data	*In-Distribution* GSM	MATH	*Out-Of-Distribution* Mathematics	MAWPS	SVAMP	MMLU-Math	SAT-Math	Avg.
				Standard Training Data					
Mistral	Standard	56.25	13.96	14.80	73.07	53.50	37.68	31.82	40.15
	Standard + **RefAug**	**60.05**	**17.36**	**19.40**	**80.25**	**59.30**	**43.63**	**48.64**	**46.95**
Gemma	Standard	60.05	17.06	19.80	76.81	57.10	39.32	42.73	44.70
	Standard + **RefAug**	**64.59**	**23.04**	**26.70**	**85.64**	**64.70**	**46.61**	**55.00**	**52.33**
				Question Augmentation Data					
Mistral	Q-Aug	56.03	18.06	18.00	79.99	59.10	38.19	36.16	43.65
	Q-Aug×2	59.14	21.26	**20.90**	80.84	**61.50**	40.86	46.82	47.33
	Q-Aug + **RefAug**	**63.00**	**21.66**	20.50	**81.78**	60.20	**42.20**	**50.91**	**48.61**
Gemma	Q-Aug	61.11	21.98	23.90	81.78	59.70	40.45	48.18	48.16
	Q-Aug×2	63.68	24.42	23.50	82.12	59.50	42.71	48.18	49.16
	Q-Aug + **RefAug**	**68.61**	**26.38**	**28.70**	**85.39**	**66.00**	**48.05**	**51.82**	**53.56**
				Answer Augmentation Data					
Mistral	A-Aug	66.19	23.08	23.90	81.10	62.20	37.78	40.91	47.88
	A-Aug×2	67.93	27.12	28.30	83.26	66.50	42.61	45.91	51.66
	A-Aug + Q-Aug	69.67	24.32	26.90	81.82	61.20	38.50	46.82	49.90
	A-Aug + **RefAug**	**72.93**	**29.40**	**31.20**	**84.41**	**71.50**	**47.74**	**60.45**	**56.80**
Gemma	A-Aug	68.31	28.78	33.10	83.05	65.10	46.51	61.36	55.17
	A-Aug×2	70.66	31.14	33.30	85.22	**69.70**	47.13	54.55	55.96
	A-Aug + **RefAug**	**74.15**	**33.60**	**38.20**	**85.68**	69.10	**52.26**	**64.09**	**59.58**
				MetaMath Augmentation Data					
Mistral	MetaMath_{40k}	68.46	20.96	20.30	85.09	66.50	38.09	42.73	48.88
	MetaMath_{80k}	69.29	23.54	23.20	86.75	68.60	41.17	43.64	50.88
	MetaMath_{40k} + **RefAug**$_{40k}$	**73.84**	**26.60**	**27.00**	**87.68**	**75.30**	**44.15**	**53.18**	**55.39**
	MetaMath_{400k}*	77.48	28.42	33.00	90.10	**79.10**	48.77	55.00	58.84
	MetaMath_{400k} + **RefAug**$_{40k}$	**78.70**	**32.50**	**34.50**	**91.59**	77.90	**49.69**	**59.09**	**60.57**
	MetaMath_{400k} (CT)	78.39	28.72	32.70	90.87	78.90	49.08	55.91	59.22
	MetaMath_{400k} + **RefAug**$_{40k}$ (CT)	**78.92**	**30.12**	**36.20**	**91.46**	**79.90**	**49.69**	**57.27**	**60.51**

Table 1: Accuracy on single-round math reasoning tasks. * The public checkpoint released by Yu et al. (2024).

not utilized during inference, this advancement underscores RefAug's role in enhancing model learning, which strengthens math problem-solving capabilities without providing additional context.

Complementary Benefits with Existing Methods: While data expansion methods (Q-Aug, A-Aug, and MetaMath) have improved model performance, combining RefAug with them leads to further substantial gains, improving overall accuracy by +6.1 on average. This demonstrates that RefAug still holds value on high-quality data[3] and is complementary to data expansion strategies. Furthermore, such synergistic benefits outpace the diminishing returns seen with repeated dataset expansions: these three methods bring +6.8 improvement initially but only +2.3 in the second round. This disparity indicates that expanding data does not always yield proportionate gains, whereas the balance of practicing new problems and reflecting on existing ones maximizes the learning effect.

Effectiveness on Large Datasets: Even when only 10% of the full-sized MetaMath dataset includes the reflective section, the resulting model surpasses the public MetaMath checkpoint by ~2 points. This confirms RefAug's efficacy on larger scales of data. Additionally, the MetaMath model barely benefits from continual training on its original QA data, suggesting a good memorization of these math problems. Nevertheless, RefAug still manages to elevate its performance, indicating that the model has not fully internalized the dataset's knowledge and RefAug effectively deepens the model's understanding of these problems.

4.2 Reflective Math Reasoning

4.2.1 Tasks

Many realistic math applications require models to reflect on previous predictions and perform further reasoning. We employ three tasks of this kind: the follow-up QA (**FQA**) and error correction (**EC**) tasks of MathChat (Liang et al., 2024), and the math subset of **MINT** (Wang et al., 2024a). FQA involves solving two subsequent questions linked to each initial query, forming a three-round interaction. EC deliberately writes an erroneous solution to test the model's error identification and correc-

[3]In Appendix A.4, we show that GPT-written solutions are of higher quality than those original ones in GSM and MATH.

Training Data	MathChat-FQA			MathChat-EC	MINT-Math					
	1st	*2nd*	*3rd*		$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	Δ
Standard	56.25	25.72	15.25	50.68	20.88	24.91	27.47	28.57	28.94	8.06
Standard + **RefAug**	**60.05**	**35.36**	**27.54**	**72.99**	**22.34**	**33.70**	**37.00**	**38.10**	**39.56**	**17.22**
Q-Aug	56.03	30.65	21.02	65.48	21.98	27.47	30.04	31.87	32.60	10.62
Q-Aug×2	59.14	32.70	22.99	63.51	**27.11**	32.60	35.16	36.26	37.73	10.62
Q-Aug + **RefAug**	**63.00**	**42.19**	**34.37**	**76.48**	26.74	**37.36**	**41.03**	**42.86**	**43.22**	**16.48**
A-Aug	66.19	34.29	23.60	72.08	23.08	30.77	33.70	35.16	35.53	12.45
A-Aug×2	67.93	36.57	28.00	71.93	25.64	31.87	33.33	34.80	34.80	9.16
A-Aug + Q-Aug	69.67	37.86	27.31	69.58	23.44	31.87	35.16	37.36	38.10	14.66
A-Aug + **RefAug**	**72.93**	**44.92**	**36.19**	**80.20**	**28.94**	**42.12**	**46.15**	**47.28**	**47.99**	**19.05**
MetaMath	68.46	37.48	24.89	61.15	22.34	27.84	31.50	32.23	33.70	11.36
MetaMath×2	69.29	38.92	26.10	60.09	21.61	25.64	26.74	27.47	27.84	6.23
MetaMath + **RefAug**	**73.84**	**43.93**	**34.98**	**79.51**	**27.47**	**36.63**	**39.93**	**40.66**	**41.03**	**13.56**

Table 2: Accuracy on reflective math reasoning tasks. Each question in MathChat-FQA has two subsequent questions (*2nd* and *3rd* turns), and the accuracy of each turn is calculated separately. MINT evaluates whether the model solves the math problem within k interaction turns with the feedback from GPT-4, and we use the difference (Δ) between $k=5$ and $k=1$ to indicate the model's ability in leveraging external feedback.

Model	Data	FQA		EC	Avg.
		2nd	*3rd*		
MAmmoTH	184K	32.16	19.31	54.15	35.21
MetaMath	395K	43.98	32.16	56.30	44.15
WizardMath	112K*	44.81	<u>36.86</u>	68.22	49.96
InternLM2-Math	~2M	40.20	28.64	72.70	47.18
DeepSeek-Math	776K	**48.19**	35.70	74.34	52.74
Mistral+A-Aug+**RefAug**	30K	44.92	36.19	<u>80.20</u>	<u>53.77</u>
Gemma+A-Aug+**RefAug**	30K	<u>47.80</u>	**38.54**	**81.11**	**55.82**

Table 3: MathChat results compared with other open-source 7B math models. Baseline scores are from Liang et al. (2024). The best scores are **bolded** and the second bests are <u>underlined</u>. *Including both supervised fine-tuning and reinforcement learning data.

tion abilities. MINT evaluates the model's ability to leverage external language feedback to improve its reasoning process through up to k turns of interaction. More task details are in Appendix B.2.

4.2.2 Results

Results on reflective math reasoning tasks are displayed in Tables 2-3 for Mistral and Table 11 for Gemma. We summarize the key findings below.

Challenges for Data Expansion Methods: Despite improving single-round QA performance, methods like Q-Aug, A-Aug, and MetaMath fall short in enhancing LMs' reflective reasoning abilities. For instance, these methods hurt Mistral's error correction performance. Moreover, a second round of augmentation yields minimal or negative gains across key metrics on reflective reasoning: +2.5 in FQA-*3rd*, -1.1 in EC, -0.5 in MINT$_{k=5}$, and -4.2 in MINT$_{\Delta}$. This indicates that initial augmentation benefits are mainly due to the improved answer quality from GPT annotation[3] rather than an actual increase in reflective reasoning skills, which echos the findings of Liang et al. (2024) that conventional training approaches overly focus on the single-round QA setting and neglect many other important mathematical scenarios.

Superiority of RefAug in Enhancing Reflective Reasoning: RefAug significantly enhances the model's reflective reasoning performance, with gains of +12.3 in FQA-*3rd*, +22.3 in EC, +10.6 in MINT$_{k=5}$, and +9.2 in MINT$_{\Delta}$, far exceeding the corresponding improvements of +7.9, +15.5, +5.0, and +3.4 brought by three data expansion methods on average. An effective solution, however, is to combine RefAug with these methods, which yields substantial improvements over them, *e.g.*, +12 on FQA-*3rd* and +10.1 on MINT$_{k=5}$. These results highlight RefAug's exceptional capability to improve LMs' reflective math reasoning, which complements the disregard of existing augmentation methods on this dimension.

Comparison with Existing Open-Source Models: Our RefAug-enhanced models excel in the reflective reasoning scenarios of MathChat with just 30K training instances, surpassing many open-source models trained on larger math datasets or with reinforcement learning. This further supports RefAug's effectiveness in cultivating LMs' reflective reasoning skills in solving math problems.

Based on findings from §4.1 and §4.2, we conclude the benefits of RefAug on math reasoning as: ***Not only does it enhance LMs' basic problem-solving skills but also advances their reflective reasoning abilities, making it a valuable complement to existing augmentation techniques.***

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU-Math	SAT-Math	Avg.
Standard	56.25	13.96	14.80	73.07	53.50	37.68	31.82	40.15
+ Alternative Reasoning	59.51	16.42	17.90	79.57	58.30	39.63	44.09	45.06
+ Follow-up Reasoning	56.25	16.82	18.80	77.10	58.50	38.09	44.05	44.23
+ **RefAug**	**60.05**	**17.36**	**19.40**	**80.25**	**59.30**	**43.63**	**48.64**	**46.95**

Table 4: Accuracy on standard math reasoning tasks when varying the components of the reflective section.

Model	HE	HE+	MBPP	MBPP+	Avg.
CodeLlama-std	53.7	50.6	62.9	51.6	54.7
CodeLlama-**RefAug**	**57.9**	**53.0**	**65.4**	**52.4**	**57.2**
Mistral-std	38.4	35.4	53.1	40.1	41.7
Mistral-**RefAug**	**50.0**	**45.1**	**56.4**	**46.4**	**49.5**
StarCoder2-std	54.3	49.4	62.7	51.4	54.4
StarCoder2-**RefAug**	**56.7**	**50.6**	**66.7**	**51.6**	**56.4**
DeepSeekCoder-std	**67.1**	59.8	75.4	60.4	65.7
DeepSeekCoder-**RefAug**	**67.1**	**62.2**	**76.7**	**63.2**	**67.3**

Table 5: Pass@1 on code generation, scored by EvalPlus. -std denotes training with the standard QA setting.

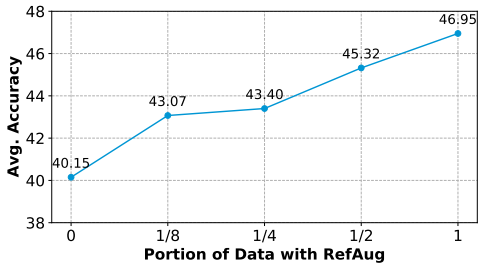


Figure 4: Average accuracy on 7 standard math reasoning tasks when different proportions of data are augmented with reflective sections (remaining data are in the standard QA form).

4.3 Code Generation

Besides math reasoning, we extend the application of RefAug to code generation. In this task, a query instructs the model to craft a code snippet that fulfills a specific functionality, which also requires a step-by-step logical flow. We use HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) as the evaluation benchmarks, along with their plus versions provided by EvalPlus (Liu et al., 2023). Training is conducted using the Python subset of Magicoder-OSS-Instruct (Wei et al., 2023), which includes 38K QA instances. Considering the abstractive nature of code, we annotate problem analogies as the follow-up section of RefAug.

The outcomes are summarized in Table 5, covering four different base LMs: CodeLLaMA (Rozière et al., 2023), Mistral, StarCoder2 (Lozhkov et al., 2024), and DeepSeekCoder (Guo et al., 2024). The results demonstrate that RefAug consistently elevates the LMs' proficiency in following instructions to generate accurate, reasonable code, as evidenced by an average improvement of +3.5 in Pass@1 across the evaluated benchmarks. These results indicate that RefAug is able to enhance LMs' capabilities in solving code problems, which reaffirms from another scenario that reflection is an essential ability for LMs to possess.

4.4 Analysis

In this section, we dive deeper into additional aspects of RefAug. Results are tested on Mistral.

4.4.1 Ablation Study

To further assess the efficacy of the reflective section, we conduct an ablation study on its two components: alternative and follow-up reasoning. According to Table 4, incorporating any single reflective component to the original data significantly enhances model performance by an average of +4.5 points. This suggests that the original solutions lack sufficient information for the model to fully grasp the math reasoning skills, which is consistent with the findings of Anonymous (2024). Combining both reflective components further enhances the model's comprehension of associated concepts and methodologies, improving the performance by +2.3 points over using any single one.

4.4.2 The Amount of RefAug Data

We explore the impact of varying the quantity of reflection-augmented instances in the whole training set. As depicted by Figure 4, the model's overall performance continually improves as more instances are augmented with reflective sections. When the model is trained through reflecting on all instances, the model maximizes its grasp of the training data and reaches the best performance, underscoring the scalability of RefAug's benefits.

4.4.3 RefAug *vs.* Chain-of-Thought

For a deeper understanding of the reflective section, we experiment with positioning it before the original solution, *i.e.*, modeling $\mathcal{P}([r; a]|q)$. This arrangement can be regarded as augmenting the chain-of-thought (CoT, Wei et al., 2022) for solv-

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU	SAT	Avg.	FQA-*2nd*	FQA-*3rd*	EC
A-Aug	66.19	23.08	23.90	81.10	62.20	37.78	40.91	47.88	34.29	23.60	72.08
+RefAug-*front*	72.78	27.34	28.30	**84.62**	70.30	47.23	56.82	55.34	30.96	20.64	68.29
+RefAug	**72.93**	**29.40**	**31.20**	84.41	**71.50**	**47.74**	**60.45**	**56.80**	**44.92**	**36.19**	**80.20**

Table 6: Comparison between RefAug and prepending the reflective section to the answer (RefAug-*front*).

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU-Math	SAT-Math	Avg.
Standard	56.25	13.96	14.80	73.07	53.50	37.68	31.82	40.15
+ RefAug #1	60.05	17.36	19.40	80.25	59.30	43.63	48.64	46.95
+ RefAug #2	62.70	17.26	19.20	82.16	60.40	42.51	44.55	46.97
+ RefAug #3	60.80	16.86	18.60	80.29	59.70	42.92	45.45	46.37
+ RefAug (Avg.)	$61.18_{\pm 1.1}$	$17.16_{\pm 0.2}$	$19.07_{\pm 0.3}$	$80.90_{\pm 0.9}$	$59.80_{\pm 0.4}$	$43.02_{\pm 0.5}$	$46.21_{\pm 1.7}$	$46.76_{\pm 0.3}$

Table 7: We sample the reflective sections three times using the same annotation prompt in Figure 8, and train a separate Mistral model using each batch of the augmented data (labeled as #1~#3). The last row lists the average scores of three runs as well as their standard deviation.

Training	Reasoning	Calculation	Total
Standard	424	287	577
RefAug	374(-50)	264(-23)	527

Table 8: Error analysis on GSM8k test set. The reduction of errors is denoted in gray parentheses.

ing the original problem. According to Table 6, since the reflective section contains relevant reasoning steps to the original problem, integrating it into CoT yields similar improvements as RefAug on single-round QA. However, such setup hurts performance in reflective math reasoning, which supports the original design of RefAug in developing reflective reasoning skills and reaffirms that **reflective reasoning demands distinct capabilities from standard forward reasoning**. Besides, augmenting CoT increases the token count required for predicting the final answer, thereby reducing inference efficiency (see Appendix A.6 for details).

4.4.4 Error Analysis

We analyze how the model's math capabilities has been enhanced through the lens of an error analysis. Following Li et al. (2024a), we classify errors in GSM8k into *calculation* errors and *reasoning* errors. Calculation errors include incorrect identification of arithmetic relationships or wrong numerical computations. Reasoning errors include mistakes pertaining to the reasoning logic, *e.g.*, incoherent reasoning steps, misunderstandings of the problem, etc. Using the gold reasoning paths from GSM8k test data as a benchmark, we employ GPT-4 to determine whether solutions contain calculation errors, reasoning errors, or both. As shown in Table 8, **the improvement mostly comes from the reduction of reasoning errors**. This supports the hypothesis that training with reflection enhances the model's problem-solving accuracy by deepening its grasp of underlying math reasoning skills.

4.4.5 Stability of RefAug Data Annotation

To verify the stability of the improvements and to avoid bias from cherry-picking augmented data, we sampled reflective sections three times using GPT-4-*turbo* with the same prompt in Figure 8. Each batch of augmented data is used to train a separate model. As shown in Table 7, the performance gains are consistent across all augmentation samples, with a minimal standard deviation of 0.3 in overall accuracy. These results confirm that reflective practices aid in model learning and that the **observed improvements are not due to the variability of data sampling**.

In addition to the above perspectives, further analyses on the risk of data contamination and efficiency statistics in training and inference are presented in Appendix A.5 and A.6, respectively.

5 Conclusion

This paper proposed reflective augmentation (RefAug) for math reasoning, a method that incorporates reflection into training problems and is complementary to existing data augmentation approaches. We proved the efficacy of RefAug in not only enhancing LMs' basic problem-solving skills on single-round math problems but also in cultivating their capabilities to solve more complex reflective reasoning tasks. We further verified the effectiveness of RefAug in code generation tasks and its scalability, along with ablation studies and analyses of the methodological choices, such as the impact of data sequencing and the stability of the annotation process.

Limitations

Some previous data augmentation studies in math reasoning created millions of data instances with OpenAI's GPT models (Li et al., 2024a; Tang et al., 2024; Huang et al., 2024b). While testing our method at a similar scale would be valuable, budget constraints limit our ability to do so. For instance, our augmentation data for MetaMath is capped at 40K instances. In Appendix A.1, we note that LLaMA-3-70B shows some promising performance in annotating RefAug data for math reasoning tasks, though its capabilities have not fully matched those of GPT-4 yet. We anticipate that the development of stronger open-source models will reduce researchers' dependence on paid services of proprietary models.

References

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. 2023. Learning from mistakes makes LLM better reasoner. *Arxiv preprint*, 2310.20689.

John R Anderson and Jon M Fincham. 2014. Extending problem-solving procedures through reflection. *Cognitive psychology*.

Anonymous. 2024. Enrichmath: Enriching idea and solution elicit mathematical reasoning in large language models. *OpenReview.net*.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *Arxiv preprint*, 2108.07732.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. Llemma: An open language model for mathematics. *Arxiv preprint*, 2310.10631.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, and et al. 2021. Evaluating large language models trained on code. *Arxiv preprint*, 2107.03374.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. In *EMNLP 2023*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2024. Teaching large language models to self-debug. In *ICLR 2024*.

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

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

Alex Davies, Petar Velickovic, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomasev, Richard Tanburn, Peter W. Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. 2021. Advancing mathematics by guiding human intuition with AI. *Nature*.

Run-Ze Fan, Xuefeng Li, Haoyang Zou, Junlong Li, Shwai He, Ethan Chern, Jiewen Hu, and Pengfei Liu. 2024. Reformatted alignment. *Arxiv preprint*, 2402.12219.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *ICML 2023*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2024. Tora: A tool-integrated reasoning agent for mathematical problem solving. In *ICLR 2024*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *Arxiv preprint*, 2401.14196.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *ICLR 2021*.

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

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024a. Large language models cannot self-correct reasoning yet. In *ICLR 2024*.

Yiming Huang, Xiao Liu, Yeyun Gong, Zhibin Gou, Yelong Shen, Nan Duan, and Weizhu Chen. 2024b. Key-point-driven data synthesis with its enhancement on mathematical reasoning. *Arxiv preprint*, 2403.02333.

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. 2023a. Mistral 7b. *Arxiv preprint*, 2310.06825.

Weisen Jiang, Han Shi, Longhui Yu, Zhengying Liu, Yu Zhang, Zhenguo Li, and James T. Kwok. 2023b. Forward-backward reasoning in large language models for verification. *Arxiv preprint*, 2308.07758.

Jerome Kagan, Bernice L Rosman, Deborah Day, Joseph Albert, and William Phillips. 1964. Information processing in the child: Significance of analytic and reflective attitudes. *Psychological Monographs: General and Applied*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS 2022*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *NAACL-HLT 2016*.

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. 2024a. Common 7b language models already possess strong math capabilities. *Arxiv preprint*, 2403.04706.

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

Yanhong Li, Chenghao Yang, and Allyson Ettinger. 2024b. When hindsight is not 20/20: Testing limits on reflective thinking in large language models. *Arxiv preprint*.

Zhenwen Liang, Dian Yu, Wenhao Yu, Wenlin Yao, Zhihan Zhang, Xiangliang Zhang, and Dong Yu. 2024. Mathchat: Benchmarking mathematical reasoning and instruction following in multi-turn interactions. *Arxiv preprint*, 2405.19444.

Woong Lim, Ji-Eun Lee, Kersti Tyson, Hee-Jeong Kim, and Jihye Kim. 2020. An integral part of facilitating mathematical discussions: Follow-up questioning. *International Journal of Science and Mathematics Education*.

Haoxiong Liu, Yifan Zhang, Yifan Luo, and Andrew Chi-Chih Yao. 2024. Augmenting math word problems via iterative question composing. *Arxiv preprint*, 2401.09003.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. In *NeurIPS 2023*.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu, Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, and et al. 2024. Starcoder 2 and the stack v2: The next generation. *Arxiv preprint*, 2402.19173.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. 2024. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. *Arxiv preprint*, 2402.16352.

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

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023b. Wizardcoder: Empowering code large language models with evol-instruct. *Arxiv preprint*, 2306.08568.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *NeurIPS 2023*.

Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, and et al. 2024. Gemma: Open models based on gemini research and technology. *Arxiv preprint*, 2403.08295.

Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. *Blog*.

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

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *Arxiv preprint*, 2404.19733.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *NAACL-HLT 2021*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: memory optimizations toward training trillion parameter models. In *SC 2020*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *KDD 2020*.

Doug Rohrer and Kelli Taylor. 2006. The effects of overlearning and distributed practise on the retention of mathematics knowledge. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *Arxiv preprint*, 2308.12950.

Nuriye Semerci. 2005. The effects of problem-based learning on the academic achievement of students in development and learning. *International Journal of Educational Reform*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *NeurIPS 2023*.

Edward A Silver. 1994. On mathematical problem posing. *For the learning of mathematics*.

Kaye Stacey, L Burton, and J Mason. 1982. *Thinking mathematically*. Addison-Wesley.

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

Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2023a. Mathcoder: Seamless code integration in llms for enhanced mathematical reasoning. *Arxiv preprint*, 2310.03731.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. 2024a. MINT: evaluating llms in multi-turn interaction with tools and language feedback. In *ICLR 2024*.

Yejie Wang, Keqing He, Guanting Dong, Pei Wang, Weihao Zeng, Muxi Diao, Yutao Mou, Mengdi Zhang, Jingang Wang, Xunliang Cai, and Weiran Xu. 2024b. Dolphcoder: Echo-locating code large language models with diverse and multi-objective instruction tuning. *Arxiv preprint*, 2402.09136.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023b. How far can camels go? exploring the state of instruction tuning on open resources. In *NeurIPS 2023*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS 2022*.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2023. Magicoder: Source code is all you need. *Arxiv preprint*, 2312.02120.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of EMNLP 2023*.

Annekatrin Wetzstein and Winfried Hacker. 2004. Reflective verbalization improves solutions—the effects of question-based reflection in design problem solving. *Applied Cognitive Psychology*.

Zhenyu Wu, Qingkai Zeng, Zhihan Zhang, Zhaoxuan Tan, Chao Shen, and Meng Jiang. 2024. Large language models can self-correct with minimal effort. *Arxiv preprint*, 2405.14092.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR 2024*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, Zhenghao Liu, Bowen Zhou, Hao Peng, Zhiyuan Liu, and Maosong Sun. 2024. Advancing LLM reasoning generalists with preference trees. *Arxiv preprint*, 2404.02078.

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*, 2308.01825.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *Arxiv preprint*, 2309.05653.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models. *Arxiv preprint*, 2304.06364.

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU	SAT	Avg.	FQA-*2nd*	FQA-*3rd*	EC
Standard	56.25	13.96	14.80	73.07	53.50	37.68	31.82	40.15	25.72	15.25	50.68
+ RefAug (GPT)	60.05	**17.36**	**19.40**	80.25	59.30	**43.63**	**48.64**	**46.95**	**35.36**	**27.54**	**72.99**
+ RefAug (LLaMA)	**62.02**	17.00	17.80	**80.29**	**61.60**	39.43	44.55	46.10	32.63	23.90	50.00

Table 9: Training Mistral-7B with data where reflection sections are annotated by GPT-4-*turbo* or LLaMA-3-70B-Instruct. Data annotated by LLaMA-3 yields similar improvements in standard math reasoning tasks, but fails to match GPT-annotated data in enhancing Mistral's reflective reasoning capabilities.

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU-Math	SAT-Math	Avg.
Standard	64.59	19.86	20.20	81.35	66.00	45.59	47.73	49.33
+ **RefAug**	**67.10**	**22.08**	**25.60**	**83.64**	**69.40**	**48.97**	**55.00**	**53.11**
GPT-Written Solutions	71.72	28.04	32.90	85.26	73.20	47.84	55.00	56.28
+ **RefAug**	**75.74**	**31.64**	**32.00**	**87.38**	**75.80**	**51.75**	**69.09**	**60.49**

Table 10: Results on LLaMA-3-8B. We test integrating RefAug with (1) the original training data, and (2) the data where answers are re-written by GPT-4-*turbo* (see Appendix A.4 for GPT answer re-writing).

A Additional Experiments

In this section, we present more experimental results in addition to those in §4.

A.1 Data Annotation with Open-Source Models

The RefAug data used in the main experiments are annotated by GPT-4-*turbo*. However, such annotation requires paid API calls to OpenAI's service, which comes with a restrictive license. To this end, we explore whether state-of-the-art open-source models can also serve as data annotators. We employ the recently released LLaMA-3-70B-Instruct model (Meta, 2024) for data annotation using the same prompt shown in Figure 8, and train a Mistral-7B model based on this data. According to results in Table 9, RefAug data annotated by LLaMA-3-70B-Instruct yields a similar improvement in Mistral's performance on standard math reasoning tasks. However, the reflective reasoning capability of the resulting model falls short of its counterpart trained with GPT-annotated data. This suggests that **developing models with advanced reflective math reasoning skills demands higher quality data than what is typically required for standard forward reasoning in single-round QA**.

A.2 Results on LLaMA-3

In addition to training Mistral-7B and Gemma-7B with RefAug, we also test LLaMA-3-8B (Meta, 2024) on the RefAug data. According to the results in Table 10, **RefAug enhances the math reasoning capabilities of LLaMA-3 as well**, no matter if integrating with the original solutions or with solutions re-written by GPT-4-*turbo*. This again shows the generalizability of the RefAug method, which leads to consistent improvements across various base models.

Training Data	MathChat-FQA *1st*	MathChat-FQA *2nd*	MathChat-FQA *3rd*	MathChat-EC
Standard	60.05	30.05	20.56	61.99
Standard + **RefAug**	**64.59**	**40.44**	**33.16**	**77.47**
Q-Aug	61.11	34.67	26.25	67.68
Q-Aug×2	63.68	34.45	26.40	70.41
Q-Aug + **RefAug**	**68.61**	**42.64**	**34.22**	**79.97**
A-Aug	68.31	41.05	29.59	73.98
A-Aug×2	70.66	42.79	32.25	77.39
A-Aug + **RefAug**	**74.15**	**47.80**	**38.54**	**81.11**

Table 11: Results of Gemma on reflective math reasoning tasks. The general trend is similar to that of Mistral (Table 2).

A.3 Gemma on Reflective Math Reasoning

Besides evaluating Mistral-based models on reflective reasoning tasks (shown in Table 2, we report scores on our Gemma-based models as well. As shown in Table 11, **the performance trends for Gemma models align with those observed on Mistral models**. RefAug demonstrates a clear advantage over traditional augmentation methods in enhancing reflective math reasoning capabilities of LMs. For instance, RefAug outscores both Q-Aug and A-Aug in the third round of follow-up QA and in the accuracy of error correction. Furthermore, as shown in Table 3, a combination of A-Aug and RefAug data results in the best-performing model on the reflective reasoning scenarios of MathChat, outperforming many open-source models that are

Data	GSM	MATH	Mathematics	MAWPS	SVAMP	MMLU-Math	SAT-Math	Avg.
Original Solutions	56.25	13.96	14.80	73.07	53.50	37.68	31.82	40.15
GPT-4-*turbo* Solutions	65.73	23.10	23.90	81.14	68.80	40.25	41.36	49.18
+ **RefAug**	**71.80**	**26.12**	**29.50**	**82.84**	**70.80**	**44.76**	**57.73**	**54.79**

Table 12: Comparison between using synthetic solutions written by GPT-4-*turbo* and using the originally annotated ones in GSM8k and MATH training sets, as well as applying RefAug on the synthetic solutions. Solutions written by GPT-4-*turbo* are of much higher quality than the original ones.

Dataset	Source	Target	Overlap
GSM8k	Train Question	Test Question	1
	Train Answer	Test Answer	0
	RefAug	Test Answer	0
MATH	Train Question	Test Question	228
	Train Answer	Test Answer	167
	RefAug	Test Answer	5*

Table 13: The contamination check on GSM8k and MATH: the number of instances from the test set (target) sharing n-gram overlaps with the training data (source). We use $n = 20$ for questions and $n = 30$ for answers. * The 5 test instances that overlap with the augmented reflective sections were already contaminated by the original MATH training set.

Training	Data	Time
Standard	15K	60 min
Q-Aug / A-Aug	30K	123 min
RefAug	15K	90 min

Table 14: The impact of various augmentation methods on dataset size and training time. These stats are tested on 8×A100 GPUs.

Training	Train Tokens	Test Tokens
Standard	171.4	185.5
GPT Solutions	358.3	423.5
RefAug-*front*	910.1	980.5
RefAug	892.3	219.1

Table 15: The resulting sequence lengths of each augmentation method during training and testing.

trained on substantially larger math datasets.

A.4 Quality of GPT-Written Answers

In Table 1, we find that answer augmentation significantly enhances performance. It improves the overall accuracy by +9.1 over the use of original training data, when averaged across Mistral and Gemma models. This surpasses the improvement of +7.2 on average seen with RefAug over the original data. A deeper analysis reveals that **the reasoning paths generated by GPT-4-*turbo* are of significantly higher quality than those originally provided in the GSM8k and MATH datasets**. As demonstrated in Table 12, merely replacing the original solutions with those generated by GPT-4-*turbo* increased the accuracy from 40.15 to 49.18 on Mistral. However, RefAug does not receive such benefits as it does not alter the original reasoning paths during augmentation. Given the complementary nature of these two augmentation methods, their combination further improves the model accuracy to 54.79. This echoes the synergistic performance advantage achieved by A-Aug+RefAug over both A-Aug and A-Aug×2 in Table 1.

A.5 Risk of Data Contamination

To prevent the augmented data from contaminating the test sets, we check the n-gram overlap between the augmented reflective sections and the gold solutions within the test sets of GSM8k and MATH. Following a common approach (Huang et al., 2024b; Liu et al., 2024), we utilize the test script provided by Azerbayev et al. (2023) and conduct a 20-gram check for questions and a 30-gram check for solutions. According to the results in Table 13, RefAug does not contaminate any test instances in GSM8k. In the MATH dataset, there is a pre-existing contamination issue: 228 questions and 167 solutions in the test set are already contaminated by the original training set. On the other hand, our RefAug data overlaps with only 5 instances in the test set, and these 5 instances were already contaminated by the training set. In other words, RefAug does not introduce new contamination to both test sets. In summary, **there is minimal contamination risk associated with RefAug in our experiments**.

A.6 Training and Inference Efficiency

For a deeper understanding of RefAug, we analyze its impact on the efficiency of model training and inference. To begin with, according to Table 14, while RefAug does introduce additional time overhead during model training, this increase is less significant than that caused by Q-Aug or A-Aug

Dataset	Train	Test
GSM8k (Cobbe et al., 2021)	7473	1319
MATH (Hendrycks et al., 2021b)	7500	5000
Mathematics (Davies et al., 2021)	-	1000
MAWPS (Koncel-Kedziorski et al., 2016)	-	2354
SVAMP (Patel et al., 2021)	-	1000
MMLU-Math (Hendrycks et al., 2021a)	-	974
SAT-Math (Zhong et al., 2023)	-	220
MathChat-FQA (Liang et al., 2024)	-	1319
MathChat-EC (Liang et al., 2024)	-	1319
MINT-Math (Wang et al., 2024a)	-	273
Magicoder (Wei et al., 2023)	38284	-
HumanEval (Chen et al., 2021)	-	164
MBPP (Austin et al., 2021)	-	399

Table 16: Statistics of all datasets used in our training and evaluation.

which doubles the optimization steps due to dataset expansion. Additionally, although RefAug results in longer sequence lengths in training instances, it does not impair inference efficiency, as shown by the average number of tokens generated in Table 15. This is due to the early stopping feature that eliminates the need to generate reflective sections during inference. Overall, **the efficiency impact brought by RefAug is minimal**.

B Detailed Task Settings

In this section, we detail the datasets, training hyper-parameters, and evaluation settings of each task used in our experiments. We list the size of all datasets in Table 16.

B.1 Standard Math Reasoning

Datasets In standard math reasoning, we follow a common approach (Wang et al., 2023a; Yu et al., 2024; Li et al., 2024a) to adopt the training data from GSM8k (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b) as they are paired with human-labeled reasoning paths. For evaluation, we employ a comprehensive suite of benchmarks that span a wide range of mathematical topics. Specifically, GSM8k, SVAMP (Patel et al., 2021), and MAWPS (Koncel-Kedziorski et al., 2016) focus mainly on arithmetic math word problems, while datasets such as MATH, Mathematics (Davies et al., 2021), MMLU (Hendrycks et al., 2021a), and SAT (Zhong et al., 2023) encompass a broader scope including algebra, geometry, number theory, probability, and formal logic. By difficulty levels, they cover elementary (MAWPS, SVAMP), middle school (GSM8K, SAT), and more advanced levels (Mathematics, MATH, MMLU), providing an exhaustive assessment of the mathematical capabilities of language models.

Training Settings During model training, we first tune the hyper-parameters using the original data under the standard fine-tuning recipe. then, these settings remain fixed across all models to avoid extensive hyper-parameter tuning for each variant. This approach is common in studies comparing models fine-tuned on varied datasets (Yuan et al., 2023; Li et al., 2023; An et al., 2023). Specifically, we train models for 3 epochs with a batch size of 128. The learning rate starts at 1e-5, including a warmup for the initial 3% of steps, and then linearly decreases to 20% of its initial value by the end of training. Training sequences are truncated to 4096 tokens. To speed up training, our model utilize `bfloat16` precision and are supported by FlashAttention-2 (Dao, 2023), DeepSpeed (Rasley et al., 2020), and ZeRO-3 optimization (Rajbhandari et al., 2020). For training on the full set of MetaMath, we follow the original authors' recommendation[4] to lower the learning rate to 2e-6, and for continued training on the public MetaMath checkpoint, we use a reduced learning rate of 1e-6 to be more consistent with its initial fine-tuning.

Evaluation To facilitate answer extraction during evaluation, we append `The answer is XXX.` to the reasoning path of each training instance so that the final predicted answer is explicitly stated. We adopt the evaluation script from Yue et al. (2023) that first extracts the predicted answer and then checks for an exact match with the ground-truth. Exceptions are MMLU and SAT which use multiple-choice formats instead of numerical answers. Since our training data does not contain multiple-choice questions, the model may predict the content of an option rather than its letter identifier. Thus, on these datasets, we leverage GPT-3.5-*turbo* to match the predicted content to the appropriate option before computing accuracy.

B.2 Reflective Math Reasoning

Reflective math reasoning encompasses scenarios where models must consider previously provided answers to engage in further reasoning. However, benchmarks that adequately capture this dynamic are scarce in the existing literature. Utilizing the

[4] https://huggingface.co/meta-math/MetaMath-Mistral-7B

currently available resources, we evaluate our models on three tasks: follow-up QA, error correction, and feedback utilization.

The **follow-up QA (FQA)** task is assessed using the MathChat dataset (Liang et al., 2024). Each test instance consists of three turns of questions. The first turn uses the original GSM8k test set, and subsequent turns contain follow-up questions based on earlier turns. These follow-ups often require a deeper understanding of the problem, such as performing subsequent calculations based on previous answers or introducing new constraints to the original question. The solutions generated by the model for each turn are incorporated into the input for the next turn, creating a multi-turn interaction. The accuracy of each turn is evaluated separately.

The **error correction (EC)** task, also sourced from the MathChat dataset and derived from the GSM8k test set, pairs each question with an intentionally incorrect answer. The model is then tasked with identifying and correcting errors in the reasoning process. Accuracy is determined by comparing the model's corrected answer to the ground truth.

For both tasks from MathChat, we follow the approach of Liang et al. (2024) to concatenate all previous turns into the instruction part of the input sequence. For example, in the third round of FQA, the model decodes $\mathcal{P}(a_3|[q_1; a_1; q_2; a_2; q_3])$; In EC, it decodes $\mathcal{P}(a|[q; a_{wrong}; f])$, where f is binary feedback indicating that a_{wrong} is incorrect.

The **MINT** (Wang et al., 2024a) benchmark evaluates the ability of LMs to leverage natural language feedback to improve their predictions. We utilize the math subset from the original benchmark, which includes 273 carefully selected instances from four datasets: 48 from GSM8k, 100 from MATH, 76 from MMLU, and 49 from TheoremQA (Chen et al., 2023). We adhere to the same evaluation protocols as the original paper except that we omit the code execution step as our math models are based on text reasoning. At each interaction turn, the model proposes a solution, and we collect binary feedback on answer correctness along with natural language feedback from an expert (*i.e.*, GPT-4). This feedback is then provided to the model in the subsequent turn of prediction. The model have at most $k = 5$ chances to propose solutions, and the accuracy of each turn is calculated independently. We also measure the improvement in accuracy (Δ) from the first to the fifth turn to assess the model's efficacy in leveraging feedback.

B.3 Code Generation

HumanEval (Chen et al., 2021) and **MBPP** (Austin et al., 2021) are the most popular benchmarks for evaluating code generation capabilities of LMs (Luo et al., 2023b; Wang et al., 2024b). Each test instance within these benchmarks includes a natural language prompt, based on which LMs generate a corresponding code snippet. The correctness of the code is verified using test cases. Additionally, EvalPlus (Liu et al., 2023) has developed enhanced versions of these benchmarks (**HumanEval+ / MBPP+**) that include more comprehensive test cases for a more rigorous evaluation. Therefore, we utilize the evaluation suite provided by EvalPlus on these benchmarks, where MBPP is reduced to 399 instances for quality control.

For the training dataset, we use the **OSS-Instruct** dataset collected by Magicoder (Wei et al., 2023), which consists of synthetic instruction-code pairs generated from random code snippets sourced from GitHub. Since HumanEval and MBPP focus on Python code, we extracted the Python subset from OSS-Instruct to reduce annotation costs, resulting in a total of 38K training instances. Given the abstractive nature of code generation, we opt for analogy annotations in the follow-up reasoning part of RefAug.

We adhere to the training settings outlined in the Magicoder paper for our experiments. Models are trained over two epochs with a batch size of 512. The learning rate is initiated at 5e-5, with 15 warm-up steps followed by a linear decay. Greedy decoding is employed during inference.

C Baseline Implementation

In this section, we detail our implementation of the major baseline methods that we compare with in the main paper, including question augmentation (Q-Aug), answer augmentation (A-Aug), and MetaMath augmentation.

C.1 Question Augmentation

A single round of Q-Aug enerates a new question from each existing question in the training set, effectively doubling the dataset (illustrated in Figure 1b). Both the augmented question and its solution are annotated by GPT-4-*turbo*. During the annotation, we employ a temperature of 0.7 and a top_p of 1.0 to ensure the diversity of math reasoning paths for both Q-Aug and A-Aug. we largely follow the question generation prompt from Li et al.

Training Prompt

<|system|>
Below is an instruction that describes a task. Follow the instruction to complete the request.
<|user|>
{Question}
<|assistant|>
{Answer}
Reflection:
{Reflection}

Figure 5: Prompt used for training the model. Text in gray are placeholders and will be replaced by the corresponding sections in the training instance.

(2024a) with minor adjustments. The detailed annotation prompt is provided in Figure 6.

C.2 Answer Augmentation

A single round of A-Aug involves re-sampling a solution for each math problem in the training set. The new solution, paired with the original question, forms a new training instance (illustrated in Figure 1c). Consistent with other methods, the augmented solution is generated by GPT-4-*turbo*. If the sampled solution diverges from the gold answer, it is discarded and re-sampled; And if a correct answer is not produced after five attempts, we retain the last sampled solution. Following the methodology described by Yu et al. (2024), the prompt for A-Aug simply instructs the model to solve an arbitrary math problem, which is detailed in Figure 7.

C.3 MetaMath

MetaMath (Yu et al., 2024) introduces a comprehensive suite of augmentation methods tailored for math reasoning tasks, which has received much attention. This suite includes answer augmentation, question rephrasing, and two backward reasoning augmentation techniques: self-verification (Weng et al., 2023) and FOBAR (Jiang et al., 2023b). Each method is sampled for multiple rounds to generate a large set of 400K training data. Please refer to Yu et al. (2024) for more details on these methods.

When creating the **MetaMath$_{40k}$** subset for our experiments in §4.1, we randomly select one instance from each of the four augmentation techniques for every seed math question, which we believe is the most uniform sampling strategy. For the **MetaMath$_{80k}$** subset, we add one more instance from each technique for every seed question. The initially sampled 40K instances are further equipped with RefAug to be included in the full-dataset training (**MetaMath$_{400k}$+RefAug$_{40k}$**).

D Training Prompt

The prompt we use to build training sequences is shown in Figure 5. The format mainly follows Wang et al. (2023b), and the reflection section is appended to the original answer as the output. Loss is only calculated to tokens after `<|assistant|>`.

E RefAug Annotation Prompt

The prompt we use for annotating reflective sections are detailed in Figure 8, which includes a description of the general principles of reflective reasoning and two in-context examples. We use temperature=0.7 and top_p=1.0 when sampling with GPT-4-*turbo*.

F License of Artifacts

All training and evaluation datasets used in our experiments are publicly accessible and have been used in many prior researches. We note that the collection of RefAug data, if annotated by an external model, should comply with its terms of use. For example, using GPT-generated data is subject to the terms of use of OpenAI services[5], and using LLaMA-generated data is subject to Meta's LLaMA license agreement[6].

[5] https://openai.com/policies/terms-of-use/
[6] https://llama.meta.com/llama3/license/

Question Augmentation Prompt

Please act as a professional math teacher. Your goal is to create high quality math problems to help students learn math. You will be given a math question. Please generate a similar but new question according to the Given Question.

You have four principles to do this.
Ensure the new question only asks for one thing, be reasonable, be based on the Given Question, and have a definite answer. For example, DO NOT ask, "what is the amount of A, B and C?".
Ensure the new question is in line with common sense of life. For example, the amount someone has or pays must be a positive number, and the number of people must be an integer.
Ensure your student can answer the new question without the given question. If you want to use some numbers, conditions or background in the given question, please restate them to ensure no information is omitted in your new question.
Ensure your created question is solvable. Write the solution to it after the question.

Given Question: $$QUESTION$$

Now write a new question and its solution. The question must begin with "New Question:" and the solution must begin with "Solution to the New Question:". The solution must end with "The answer is XXX" where XXX should be the final answer to the question.

Figure 6: Prompt for question augmentation, adopted from Li et al. (2024a). The only difference is that we combine question generation and solution annotation into a single prompt to save costs.

Answer Augmentation Prompt

Your task is to solve a math word problem. You should solve the problem step by step. At the end of your solution, write the final answer in the form of "The answer is X". Here are two examples:

Example 1

Question:
Let $F_1 = (0,1)$ and $F_2 = (4,1)$. Then the set of points P such that $PF_1 + PF_2 = 6$ form an ellipse. The equation of this ellipse can be written as $\frac{(x-h)^2}{a^2} + \frac{(y-k)^2}{b^2} = 1$. Find $h + k + a + b$.

Solution:
We have that $2a = 6$, so $a = 3$. The distance between the foci is $2c = 4$, so $c = 2$. Hence, $b = \sqrt{a^2 - c^2} = \sqrt{5}$. The center of the ellipse is the midpoint of $\overline{F_1F_2}$, which is $(2,1)$. Thus, the equation of the ellipse is $\frac{(x-2)^2}{3^2} + \frac{(y-1)^2}{(\sqrt{5})^2} = 1$. Hence, $h + k + a + b = 2 + 1 + 3 + \sqrt{5} = 6 + \sqrt{5}$. The answer is $6 + \sqrt{5}$.

Example 2

Question:
Each bird eats 12 beetles per day, each snake eats 3 birds per day, and each jaguar eats 5 snakes per day. If there are 6 jaguars in a forest, how many beetles are eaten each day?

Solution:
First find the total number of snakes eaten: 5 snakes/jaguar × 6 jaguars = 30 snakes. Then find the total number of birds eaten per day: 30 snakes × 3 birds/snake = 90 snakes. Then multiply the number of snakes by the number of beetles per snake to find the total number of beetles eaten per day: 90 snakes × 12 beetles/snake = 1080 beetles. The answer is 1080.

Now solve the following problem. The solution must end with "The answer is XXX" where XXX should be the final answer to the question.

Question:
$$QUESTION$$

Solution:

Figure 7: Prompt for answer augmentation, which is basically an in-context learning prompt for solving a given math problem. Two in-context examples come from MATH and GSM8k training sets, respectively.

Data Annotation Prompt

You are a professional math teacher, and your goal is to teach your student to learn a given math problem. Now that your student has successfully solved the original problem, in order to make the student thoroughly understand the involved knowledge and problem-solving methodology, your task is to write a reflection section that go through the problem-solving process and provide additional insights. The reflection section should include the following components:

1. Alternative Reasoning: Present an alternative approach to solve the original problem. This alternative approach should be distinct from the original solution and still lead to the correct answer. While writing the alternative reasoning approach, consider explaining the principle of the methodology used in the original solution, how the alternative approach differs from the original method, and why it leads to the same correct answer.

2. Follow-up Reasoning: Associate the solution to a broader class of problems. You can either create a general form of the original problem to encourage the student to reduce reliance on specific values (e.g., use letters or variables to replace specific numbers in the original problem), or apply the concepts and methodologies from the original problem to a more challenging situation. Please do not just replace the original numbers in the question with new numbers, because that is essentially the same problem. The follow-up problem must also be solvable, and you need to provide the solution for it. Besides, please explain briefly how the new scenario associates with the original problem.

Example 1:

Original Problem:
Youngsville had a population of 684 people. The town had a growth spurt and the population increased by 25% then they witnessed that 40% of the population moved away. What is the current population?

Solution to the Original Problem:
The town had 684 people, and then had a 25% growth spurt, so the population increased by $684\times0.25 = 171$ people. This increase brought the population to $684 + 171 = 855$ people. 40% of the population moved away, so $855\times0.40 = 342$ people moved away. The new population is $855 - 342 = 513$ people. The answer is 513.

Alternative Reasoning:
The key to solve the problem is to understand the concept of relative increase and decrease percentages. Increasing by a% means the population grows to $(100 + a)$% of the original, while decreasing by b% means the population reduces to $(100 - b)$% based on the increased population. Therefore, this is essentially a problem of consecutive multiplication: multiply the initial total population by the percentage of change twice.
Therefore, an alternative calculation involves deriving a single effective percentage change of the whole process. A 25% increase is equivalent to multiplying by 1.25, and a 40% decrease is equivalent to multiplying by 0.60. Combining these two changes, the effective percentage change is $1.25\times0.60 = 0.75$, which corresponds to a 25% decrease from the original population. Therefore, the current population is $684\times0.75 = 513$. The alternative approach leads to the same result because the associative property of multiplication: $(684\times1.25)\times0.60 = 684\times(1.25\times0.60) = 684\times0.75 = 513$.

Follow-up reasoning:
Let‘s think of a more general scenario. Suppose a town has a population of P people. The population increases by a percent, then b percent of the population moves away, and we would like to know the final population. In this context, the first increase corresponds to multiplying by $(1 + a/100)$, and the subsequent decrease corresponds to multiplying by $(1 - b/100)$. So the total population change is $(1 + a/100)(1 - b/100)$. Therefore, the final population is $P(1 + a/100)(1 - b/100)$. This abstract problem allows us to apply the same principles of relative percentage changes to calculate the final population based on the initial population and the two percentage changes. This generalization helps to understand the problem conceptually and apply it to various scenarios.

Example 2:

Original Problem:
Solve the equation $(x - 99)(x - 101) = 8$.

Solution to the Original Problem:
Let t=x-100. Then the equation becomes $(t - 1)(t + 1) = 8$, which transforms into $t^2 - 1 = 8$. Therefore, $t = 3$ or $t = -3$, and accordingly we get $x = 97$ or $x = 103$. The answer is 97 or 103.

Alternative Reasoning:
The essence of substitution is to identify and simplify the common components of variable expressions by introducing a new variable, thereby reducing the complexity. Let's revisit the original equation. Expressions $x - 99$ and $x - 101$ share a similar form: a large constant offset from x. Due to the minimal difference between 99 and 101, we can use substitution to transform the expressions into terms with small constants.
Therefore, an alternative approach is to substitute $t = x - 99$, which transforms the equation into $t(t - 2) = 8 \Rightarrow t^2 - 2t - 8 = 0$. This can be easily factorized into $(t - 4)(t + 2) = 0$. Hence, $t = 4$ or $t = -2$, leading to the same results $x = 97$ or $x = 103$. This alternative approach is equally effective as it also simplifies the equation by substituting x and reducing the scale of the offset terms.

Follow-up Reasoning:
Extending the idea of substitution, consider the equation $x(x + 1)(x + 2)(x + 3) = 360$. We notice that $x(x + 3) = x\text{^}2 + 3x$, and $(x + 1)(x + 2) = x^2 + 3x + 2$. Therefore, to simplify the expression, we set the common term $x^2 + 3x$ as t, which transforms the equation into $t(t + 2) = 360 \Rightarrow t^2 + 2t - 360 = 0 \Rightarrow t = -20$ or $t = 18$. If $t = -20$, then $x^2 + 3x + 20 = 0$. Here, the discriminant $\Delta = -71 < 0$, resulting in no real solutions for x. If $t = 18$, then $x^2 + 3x - 18 = 0$, so $x = 3$ or $x = -6$. This scenario reiterates the importance of identifying common components of x to streamline the equation through substitution.

Now write a reflection section for the following case based on the examples above. Make sure to use "Alternative Reasoning:" and "Follow-up Reasoning:" to separate the two components.

Original Problem:
$$QUESTION$$

Solution to the Original Problem:
$$RESPONSE$$

Figure 8: Prompt for annotating the reflective section. The prompt first explains the contents to annotate within the reflective section, and then presents two in-context examples for demonstration. GPT-4-*turbo* is employed for annotation.