

Can Large Models Teach Student Models to Solve Mathematical Problems Like Human Beings? A Reasoning Distillation Method via Multi-LoRA Interaction

Xinhe Li¹, Jiajun Liu¹ and Peng Wang^{1,2*}

¹School of Computer Science and Engineering, Southeast University

²Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education
lixinhe669@gmail.com, {jiajliu, pwang}@seu.edu.cn

Abstract

Recent studies have demonstrated that Large Language Models (LLMs) have strong mathematical reasoning abilities but rely on hundreds of billions of parameters. To tackle the challenge of poor reasoning in Small Language Models (SLMs), existing methods typically leverage LLMs to generate massive amounts of data for cramming training. In psychology, they are akin to System 1 thinking, which resolves reasoning problems rapidly based on experience and intuition. However, human learning also requires System 2 thinking, where knowledge is first acquired and then reinforced through practice. Inspired by such two distinct modes of thinking, we propose a novel method based on the multi-LoRA Interaction for mathematical reasoning Distillation (LoRID). First, we input the question and reasoning of each sample into an LLM to create knowledge-enhanced datasets. Subsequently, we train a LoRA block on the student model as an Intuitive Reasoner (IR), which directly generates Chain-of-Thoughts for problem-solving. Then, to imitate System 2 thinking, we train the Knowledge Generator (KG) and Deep Reasoner (DR), respectively. The former outputs only knowledge after receiving problems, while the latter uses that knowledge to perform reasoning. Finally, to address the randomness in the generation of IR and DR, we evaluate whether their outputs are consistent, and the inference process needs to be iterated if not. This step can enhance the mathematical reasoning ability of SLMs through mutual feedback. Experimental results show that LoRID achieves state-of-the-art performance, especially on the GSM8K dataset, where it outperforms the second-best method by 2.3%, 16.1%, 2.4%, 12.3%, and 1.8% accuracy across the five base models, respectively. Meanwhile, we select four strong baselines as System 1, and after integrating them with our method, the reasoning ability of student models is consistently and significantly improved. The datasets and codes are available at <https://github.com/Xinhe-Li/LoRID>.

*Corresponding author.

Question: Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?
Reasoning: Natalia sold $48 / 2 = 24$ clips in May. Natalia sold $48 + 24 = 72$ clips altogether in April and May.
Answer: 72

System 1 Thinking [LLMs Teaching SLMs]

Question: John read 36 non-fiction books in January and one-third as many fiction books in February. How many books did John read altogether in January and February?



Answer Wrong!



Lots of Training Data

System 2 Thinking [Human Beings Teaching Students]

Knowledge: Divide the given number by the specified fraction to determine the quantity for the second period. Add the original amount to the quantity calculated for the second period to determine the total quantity.



Answer Correct! (72)



Knowledge



Exercises

Figure 1: The LLMs teaching SLMs learning pattern vs. Human beings teaching students learning pattern.

1 Introduction

Large Language Models (LLMs) [Achiam *et al.*, 2023; Team *et al.*, 2023] have demonstrated superiority in mathematical reasoning with the help of Chain-of-Thought (CoT) [Wei *et al.*, 2022; Kojima *et al.*, 2022] prompts. However, although these closed-source models have strong capabilities in a variety of Natural Language Processing tasks, such as semantic understanding [Hu *et al.*, 2024; Tang *et al.*, 2023], instruction following [Longpre *et al.*, 2023; Wu *et al.*, 2024], and code generation [Chen *et al.*, 2021; Zhou *et al.*, 2024], they rely on hundreds of billions of parameters. This makes these models undeployable at scale due to overwhelming computational requirements and inference costs [Wei *et al.*, 2022]. Although Small Language Models (SLMs) have fewer parameters, they face the challenge of poor reasoning ability. For example, LLaMA-2-7B [Touvron *et al.*, 2023] and Mistral-7B [Jiang *et al.*, 2023] have only 14.6% and 15.5% accuracy on the GSM8K [Cobbe *et al.*, 2021] dataset after in-context learning [Brown *et al.*, 2020]. Therefore, how to effectively distill the mathematical reasoning ability of teacher models into SLMs is still a non-trivial problem.

To address this issue, existing works [Yu *et al.*, 2024b;

Li *et al.*, 2024a; Luo *et al.*, 2023] mainly use powerful LLMs to perform various data augmentations on CoTs (e.g., Monte Carlo Tree Search [Chaslot *et al.*, 2008; Zhang *et al.*, 2024a]) and distill reasoning capabilities into the student model through supervised fine-tuning. Meanwhile, some methods [Yin *et al.*, 2024; Yue *et al.*, 2024; Gou *et al.*, 2024] highlight the synergy between LLMs and external tools (e.g., code interpreter) to reduce computational errors. They train SLMs with extensive programming language data to develop code-generation capabilities.

However, as shown in Figure 1, the LLMs teaching SLMs learning pattern is fundamentally different from the human beings teaching students learning pattern. In psychology, there are two thinking modes: System 1 and System 2 [Kahneman, 2011]. The former typically generates quick but error-prone results, while the latter reasons through a slower and deeper thought process. Inspired by this, on one hand, the data augmentation process of most methods does not explicitly induce the knowledge and capabilities of teacher language models, which contrasts with the way humans transfer knowledge. Taking the math problem in Figure 1 as an example, they require LLMs to generate several similar questions based on the original question as a training set, instead of imitating teachers to explicitly tell the knowledge to students, which is crucial in the deep thinking of System 2. On the other hand, the model distillation process does not fully consider the interaction between System 1 and System 2, which is contrary to the way humans acquire knowledge. Intuition and deep thinking often play different roles in reasoning, and thus their complementarity aids in problem-solving. Meanwhile, although tool-based methods achieve good performance in tasks involving complex computations, they often promote excessive dependence on external tools [Li *et al.*, 2024b] and need to repeatedly send the code generated by student models to a compiler until it executes correctly.

To deal with the above issues, inspired by the human beings teaching and learning pattern, we propose a novel method based on the multi-LoRA [Hu *et al.*, 2022] Interaction for mathematical reasoning Distillation (LoRID). First, we construct the training sets by prompting a closed-source teacher model (e.g., GPT-4) with zero-shots [Wang *et al.*, 2019] to generate the knowledge required to solve math problems. Secondly, analogous to System 1, we train a LoRA block on the student model as the Intuitive Reasoner (IR), directly generating Chain-of-Thought, similar to most data augmentation-based methods. Thirdly, analogous to System 2, we train Knowledge Generator (KG) and Deep Reasoner (DR), respectively. These two modules are designed to imitate the processes of students learning knowledge and applying that in practice. Finally, inspired by the integration of System 1 and System 2 in human learning, if the outputs of IR and DR are inconsistent, the three LoRA blocks mentioned above will continue to iteratively infer until the termination conditions are met. Through this multi-LoRA interaction on the same student model, they continuously provide feedback to each other, thereby enhancing the overall problem-solving ability in a parameter-efficient manner.

We conduct experiments on the GSM8K [Cobbe *et al.*, 2021] and MATH [Hendrycks *et al.*, 2021] datasets

using LLaMA-2-7B [Touvron *et al.*, 2023], LLaMA-3-8B [Grattafiori *et al.*, 2024], Mistral-7B [Jiang *et al.*, 2023], Qwen2.5-Math-7B [Yang *et al.*, 2024], and DeepSeekMath-7B [Shao *et al.*, 2024] as our base models. Experimental results demonstrate that the interaction between System 1 and System 2 significantly enhances the mathematical reasoning abilities of student models. Especially on the GSM8K dataset, LoRID outperforms the second-best method by 2.3%, 16.1%, 2.4%, 12.3%, and 1.8% accuracy across the five base models. Furthermore, due to the plug-and-play flexibility of LoRA blocks, we select four strong baselines (MuggleMath [Li *et al.*, 2024a], MuMath [You *et al.*, 2024], MetaMath [Yu *et al.*, 2024b], and RFT [Yuan *et al.*, 2023]) as System 1, and after integrating our method, the accuracy of student models shows consistent and significant improvement.

The main contributions of this paper are three-fold:

- We focus on the mathematical reasoning distillation task and propose a novel method LoRID, to the best of our knowledge, which is among the first to draw inspiration from the human beings teaching and learning pattern.
- We introduce knowledge during data augmentation and propose multi-LoRA interaction during model distillation, which improves the student’s reasoning abilities.
- Experimental results show that with the interaction between System 1 and System 2, LoRID outperforms previous state-of-the-art approaches and can be easily and effectively integrated into any CoT distillation method.

2 Related Work

Mathematical reasoning tasks like GSM8K [Cobbe *et al.*, 2021] and MATH [Hendrycks *et al.*, 2021] are among the most challenging problems in LLMs. To solve them, recent works [Wei *et al.*, 2022; Kojima *et al.*, 2022] show that it is possible to elicit reasoning abilities by prompting LLMs to perform Chain-of-Thought (CoT) reasoning, i.e., generate a series of intermediate steps, but it reduces the accuracy of models with less than 10 billion parameters. Thus, most current methods [Li *et al.*, 2024a; Tang *et al.*, 2024] mainly use mainstream closed-source LLMs to generate diverse and high-quality enhanced data. MuMath [You *et al.*, 2024] and MetaMath [Yu *et al.*, 2024b] bootstrap the questions in both forward and backward reasoning directions. They require LLMs to produce a large volume of reasoning data, which raises both augmentation and training costs.

Another research trajectory [Yin *et al.*, 2024; Yue *et al.*, 2024] highlights the synergy between LLMs and external tools. ToRA [Gou *et al.*, 2024] interleaves Python code blocks and natural language reasoning parts in multiple rounds of the same solution, which provides a more flexible combination of CoT and Program-of-Thought (PoT). Although using a compiler to output the final answer helps reduce computational errors, it requires the student model to repeatedly generate code until it compiles correctly. Furthermore, if the SLM is only pre-trained on natural language texts, rather than programming languages, it will be difficult to enable the model to master coding capabilities based solely on supervised fine-tuning. Therefore, this paper does not consider the use of external tools.

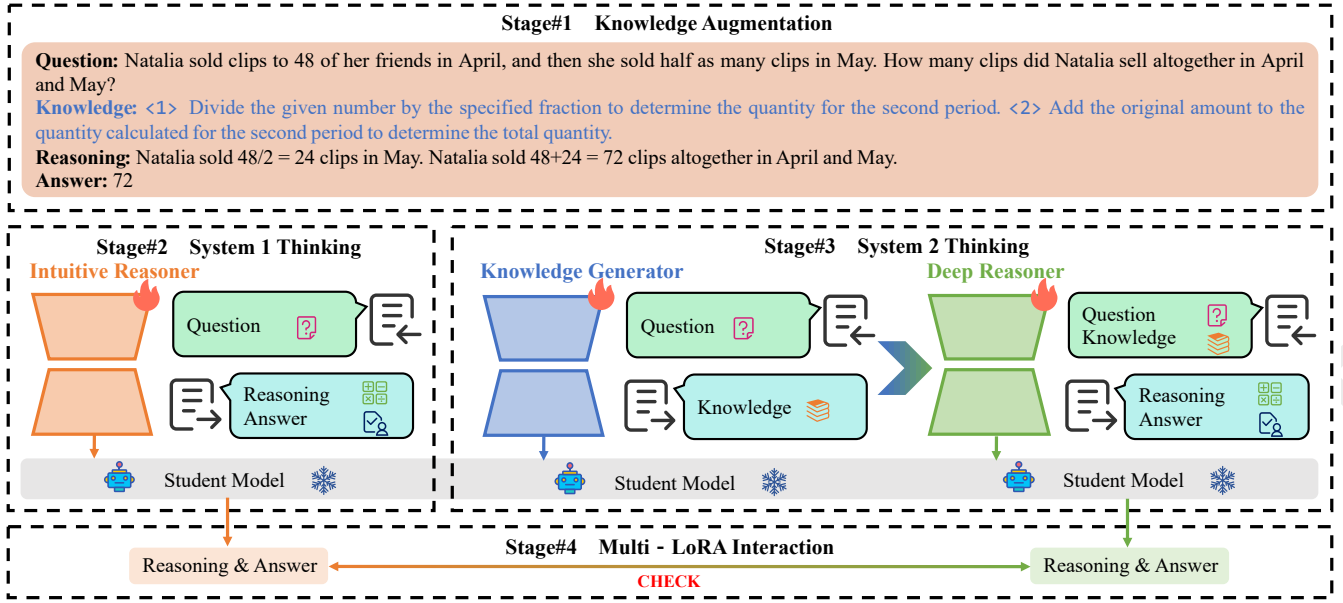


Figure 2: Overview of our proposed LoRID framework.

3 Methodology

3.1 Preliminary

A mathematical reasoning problem can be denoted as $\mathcal{D} = \{(q_i, r_i, a_i)\}_{i=1}^n \subseteq \mathcal{Q} \times \mathcal{R} \times \mathcal{A}$, where each sample includes question q_i , reasoning r_i , and answer a_i . Our task is to train a student model $f(q_i; \theta) \rightarrow [r_i \oplus a_i]$ with parameters θ to minimize the prediction loss \mathcal{L} , which can be formulated as:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^n \ell(f(q_i; \theta), [r_i \oplus a_i]). \quad (1)$$

where ℓ is the cross entropy loss between predicted tokens and target tokens, and n is the amount of data. Then we compare the answer a_i with \hat{a}_i generated by models to evaluate their mathematical reasoning ability.

3.2 Framework

The framework of LoRID is shown in Figure 2, which mainly includes four stages: knowledge augmentation, System 1 thinking, System 2 thinking, and multi-LoRA interaction. In stage 1, we use a closed-source LLM as a teacher to generate knowledge-enhanced mathematical reasoning datasets. The question and reasoning are provided as prompt inputs to inspire LLMs to output the knowledge for problem-solving. In stage 2, similar to most other methods, we train a LoRA block to generate a series of reasoning steps (i.e., CoT) for intuitive reasoning. In Stage 3, we separately train a Knowledge Generator to imitate the process of students acquiring knowledge, and a Deep Reasoner to apply that knowledge to solve mathematical problems. In stage 4, since the three LoRA blocks mentioned above are trained on the same student model, they can be plug-and-play during inference, allowing them to interact in a parameter-efficient manner. By comparing the responses from System 1 and System 2, we determine whether

further inference is required. This is similar to how students in human society need to rely not only on intuition but also on deep thinking to reason.

3.3 Knowledge Augmentation

The human learning process can be divided into two steps: (1) acquiring knowledge to solve a specific type of problem, and (2) practicing with exercises to flexibly apply that knowledge. However, the current CoT distillation paradigm [Magister *et al.*, 2023] only generates a large number of data using an LLM and then directly fine-tunes student models on these problems, which deviates from the way humans learn. Thus, motivated by this, we aim to explicitly extract knowledge from the teacher model.

Consider a standard sample d_i consisting of a question q_i , its correct reasoning r_i and answer a_i . As shown in Figure 3, we use zero-shot [Wang *et al.*, 2019] instructions I to prompt a teacher model to generate the general knowledge k_i required to solve this problem. Since most language models undergo extensive pre-training on raw text, our knowledge representation is also in the form of natural language. For any dataset \mathcal{D} , the entire process can be formulated as follows:

$$f_{\text{LLM}}(\mathcal{P}) \rightarrow \mathcal{K}. \quad (2)$$

where $\mathcal{P} = \{(I, q_i, r_i, a_i)\}_{i=1}^n$ denotes a prompt set and $\mathcal{K} = \{k_i\}_{i=1}^n$ denotes a knowledge set.

3.4 System 1 Thinking

In system 1, similar to other approaches [You *et al.*, 2024; Yu *et al.*, 2024b], we train an Intuitive Reasoner (IR) specifically designed for mathematical reasoning. The input consists of a question q_i , and the output is the concatenation of reasoning r_i and answer a_i . We train a student model $f(q_i; \theta_{W_{\text{IR}}}) \rightarrow$

Instruction

Given a math problem, it is overly complicated and lacks clear knowledge for students to grasp. Your task is to summarize the general solving rules to assist students.

Important notes:

1. The rules summarized are general and not tailored to specific questions.
2. The rules summarized should not contain any semantic information from the questions.
3. The rules summarized should map complete reasoning steps for solving the questions.

Input

Question: Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?

Reasoning: Natalia sold $48 / 2 = 24$ clips in May. Natalia sold $48 + 24 = 72$ clips altogether in April and May.

Answer: 72

Output

Divide the given number by the specified fraction to determine the quantity for the second period. Add the original amount to the quantity calculated for the second period to determine the total quantity.

Figure 3: The format of the knowledge generation prompt.

$[r_i \oplus a_i]$ to minimize the prediction loss \mathcal{L}_{IR} :

$$\mathcal{L}_{\text{IR}} = \frac{1}{n} \sum_{i=1}^n \ell(f(q_i; \theta_{W_{\text{IR}}}), [r_i \oplus a_i]) \quad (3)$$

$$W_{\text{IR}} := W_{\text{init}} + A_{\text{IR}} B_{\text{IR}} \quad (4)$$

where $W_{\text{init}} \in \mathbb{R}^{d \times k}$ denotes a pre-trained weight matrix of the student model, $A_{\text{IR}} \in \mathbb{R}^{d \times r}$ and $B_{\text{IR}} \in \mathbb{R}^{r \times k}$ are LoRA parameters of Intuitive Reasoner, and the rank $r \ll \min(d, k)$. In this phase, the student model directly learns the problem-solving skills necessary for later comparison with the output from the Deep Reasoner.

3.5 System 2 Thinking

In System 2, inspired by human learning, a first-grade student can only attempt to answer a fifth-grade math problem based on his existing knowledge. However, due to the lack of more advanced knowledge, solving the problem correctly becomes challenging. Thus, acquiring additional knowledge is essential for effective problem-solving.

For Knowledge Generator (KG), we take the question q_i as input and the knowledge k_i as output, training a student model $f(q_i; \theta_{W_{\text{KG}}}) \rightarrow k_i$ to minimize the prediction loss \mathcal{L}_{KG} :

$$\mathcal{L}_{\text{KG}} = \frac{1}{n} \sum_{i=1}^n \ell(f(q_i; \theta_{W_{\text{KG}}}), k_i) \quad (5)$$

$$W_{\text{KG}} := W_{\text{init}} + A_{\text{KG}} B_{\text{KG}} \quad (6)$$

where $A_{\text{KG}} \in \mathbb{R}^{d \times r}$ and $B_{\text{KG}} \in \mathbb{R}^{r \times k}$ are LoRA parameters of the Knowledge Generator. During this phase, the student model learns the essential knowledge required for solving problems from the teacher. Since the semantic complexity of the problem has been simplified, knowledge exhibits less diversity than reasoning, making it easier for students to grasp general rules. Without explicit knowledge, students would struggle to generalize from a large number of problems and may rely more on rote memorization.

It is widely known that acquiring knowledge enhances problem-solving abilities, but practice is also necessary for students to fully internalize this knowledge. For Deep Reasoner (DR), we concatenate the question q_i and knowledge k_i as the input, with reasoning r_i and answer a_i as the output. The student model $f([q_i \oplus k_i]; \theta_{W_{\text{DR}}}) \rightarrow [r_i \oplus a_i]$ is trained to minimize the prediction loss \mathcal{L}_{DR} :

$$\mathcal{L}_{\text{DR}} = \frac{1}{n} \sum_{i=1}^n \ell(f([q_i \oplus k_i]; \theta_{W_{\text{DR}}}), [r_i \oplus a_i]) \quad (7)$$

$$W_{\text{DR}} := W_{\text{init}} + A_{\text{DR}} B_{\text{DR}} \quad (8)$$

where $A_{\text{DR}} \in \mathbb{R}^{d \times r}$ and $B_{\text{DR}} \in \mathbb{R}^{r \times k}$ are LoRA parameters of Deep Reasoner. In the training phase, knowledge is generated by closed-source LLMs, while in the inference phase, it is provided by the Knowledge Generator.

3.6 Multi-LoRA Interaction

Just as in student learning, some mathematical problems can be solved using System 1, while others require System 2, which involves first learning the necessary knowledge and then solving the problems. Inspired by this process, integrating System 1 and System 2 is beneficial for the reasoning of student models. Since the three LoRA blocks, Intuitive Reasoner, Knowledge Generator, and Deep Reasoner, are fine-tuned on the same model, their plug-and-play advantage facilitates parameter-efficient interactive inference.

In terms of implementation, we store the answers produced by Intuitive Reasoner and Deep Reasoner in \mathcal{A}_{IR} and \mathcal{A}_{DR} , respectively, during each iteration. When both sets have the identical answer \hat{a}_i , the result is considered final, and inference stops; otherwise, the process continues. To manage inference costs, we set a threshold t to limit the number of iterations. Unlike existing methods, we do not require the Intuitive Reasoner or Deep Reasoner to produce highly accurate outputs in a single iteration; rather, we only require the final solution to be correct. This idea reduces the need for extensive training data and computational time. Similarly, humans cannot solve problems on the first attempt and typically require multiple trials and errors to find the correct answer.

4 Experiments

4.1 Experimental Setup

Datasets

We use two popular mathematical reasoning benchmarks: (1) GSM8K [Cobbe *et al.*, 2021] consists of high-quality grade school math word problems, containing 7,473 training samples and 1,319 test samples; and (2) MATH [Hendrycks *et al.*, 2021] dataset consists of high school competition problems covering seven subjects, and contains 7,500 and 5,000 samples for training and testing, respectively. Problems in GSM8K require between 2 and 8 steps to get an answer, while MATH is much more challenging.

For each sample in datasets, we call GPT-4o [Achiam *et al.*, 2023] to generate the knowledge sequence required to solve the problem. To increase the amount of data, we directly use subsets obtained by MetaMathQA [Yu *et al.*, 2024b] based on answer augmentation and question rephrasing. Since the

Dataset	Training	#GSM8K	#MATH
MuggleMath [Li <i>et al.</i> , 2024a]	System 1	152,589	147,787
MuMath [You <i>et al.</i> , 2024]	System 1	384,261	366,244
MetaMath [Yu <i>et al.</i> , 2024b]	System 1	240,000	155,000
RFT [Yuan <i>et al.</i> , 2023]	System 1	103,638	-
Ours	System 2	160,000	125,000

Table 1: Statistics of datasets for training System 1 and System 2.

two augmentations do not significantly modify the reasoning steps, data from the same original problem can share the knowledge we generate. Thus, we obtain 7,473 pieces of knowledge for GSM8K and 7,500 pieces of knowledge for MATH. The statistics of datasets are shown in Table 1.

Baselines

We compare LoRID with some strong baselines, which are divided into three groups. (1) **Closed-source models**, we compare GPT-4o, GPT-o1-mini, Claude 3.5 Sonnet, Gemini 1.5-Pro, and DeepSeek-V3 with in-context learning. (2) **Open-source models with tools**, we provide 8 baseline methods for comparison: ToRA [Gou *et al.*, 2024], MAMmoTH [Yue *et al.*, 2024], MathCoder [Wang *et al.*, 2024a], R³ [Xi *et al.*, 2024], MathGenieLM [Lu *et al.*, 2024], MuMath-Code [Yin *et al.*, 2024], OpenMath [Toshniwal *et al.*, 2024], and AlphaMath [Chen *et al.*, 2024b], which all require the help of code compilers to output the answer. (3) **Open-source models without tools**, we make comparisons with the following state-of-the-art baselines including RFT [Yuan *et al.*, 2023], MetaMath [Yu *et al.*, 2024b], QDMR [Huang *et al.*, 2024], AutoPRM [Chen *et al.*, 2024c], MuMath [You *et al.*, 2024], MathScale [Tang *et al.*, 2024], R³, MFT [Chen *et al.*, 2024a], Math-Shepherd [Wang *et al.*, 2024b], MuggleMath [Li *et al.*, 2024a], DPO-ST [Wang *et al.*, 2024c], AlphaMath, RefAug [Zhang *et al.*, 2024b], Self-Refine [Ranaldi and Freitas, 2024], and DART-Math [Tong *et al.*, 2024]. Additionally, we conduct experiments on three general models, LLaMA-2-7B, Mistral-7B, and LLaMA-3-8B [Grattafiori *et al.*, 2024], as well as two math-specialized models, Qwen2.5-Math-7B [Yang *et al.*, 2024] and DeepSeekMath-7B [Shao *et al.*, 2024]. However, methods like OVM [Yu *et al.*, 2024a], which require combining up to 100 outputs to achieve more accurate results, are not included in our comparison.

Settings

All experiments are conducted on the $8 \times$ NVIDIA A100 GPUs. We set the rank and α of LoRA to 512 and 1024 respectively. We employ the AdamW [Loshchilov and Hutter, 2017] optimizer with a cosine learning rate schedule spanning a total of 5 epochs of training. The maximum learning rate is set at $5e-5$ and there is a 3% linear warmup. Considering the diversity of generation, we set the top-p and temperature during inference to 0.90 and 1.50. The inference iteration threshold t for multi-LoRA interaction is set to 20.

4.2 Main Results

We conduct comparative experiments to evaluate the performance of each method in the mathematical reasoning task. Table 2 shows the accuracy results of all models on the GSM8K and MATH datasets.

Method	Base model	#params	GSM8K	MATH
<i>Closed-source models</i>				
ICL	GPT-4o	-	92.9	76.6
ICL	GPT-o1-mini	-	94.8	90.0
ICL	Claude 3.5 Sonnet	-	96.4	71.1
ICL	Gemini 1.5-Pro	-	91.7	58.5
ICL	DeepSeek-V3	671B	89.3	61.6
<i>Open-source models with tools</i>				
ToRA	LLaMA-2	7B	68.8	40.1
MAMmoTH	LLaMA-2	7B	53.6	31.5
MathCoder	LLaMA-2	7B	64.2	23.3
R ³	LLaMA-2	7B	68.9	-
MathGenieLM	LLaMA-2	7B	71.7	33
MuMath-Code	LLaMA-2	7B	83.8	48.8
MAMmoTH	Mistral	7B	75.0	40.0
MathGenieLM	Mistral	7B	80.5	45.1
OpenMath	Mistral	7B	80.2	44.5
AlphaMath	DeepSeekMath	7B	84.1	66.3
<i>Open-source models without tools</i>				
ICL	LLaMA-2	7B	14.6	2.5
SFT	LLaMA-2	7B	41.6	7.2
RFT	LLaMA-2	7B	51.2	-
MetaMath	LLaMA-2	7B	66.5	19.8
QDMR	LLaMA-2	7B	30.4	-
AutoPRM	LLaMA-2	7B	70.8	23.6
MuMath	LLaMA-2	7B	<u>76.2</u>	23.3
MathScale	LLaMA-2	7B	66.3	31.1
R ³	LLaMA-2	7B	50.5	-
MFT	LLaMA-2	7B	69.0	20.8
Math-Shepherd	LLaMA-2	7B	73.2	21.6
MuggleMath	LLaMA-2	7B	69.8	23.1
DPO-ST	LLaMA-2	7B	54.7	-
Ours	LLaMA-2	7B	78.5	<u>25.2</u>
ICL	LLaMA-3	8B	58.4	17.0
SFT	LLaMA-3	8B	60.9	18.1
DPO-ST	LLaMA-3	8B	68.8	-
AlphaMath	LLaMA-3	8B	<u>71.8</u>	41.9
Ours	LLaMA-3	8B	87.9	44.7
ICL	Mistral	7B	15.5	10.1
SFT	Mistral	7B	50.3	13.4
MetaMath	Mistral	7B	77.7	28.2
MathScale	Mistral	7B	74.8	<u>35.2</u>
MFT	Mistral	7B	79.5	29.0
Math-Shepherd	Mistral	7B	<u>81.8</u>	33.0
RefAug	Mistral	7B	78.9	30.1
Self-Refine	Mistral	7B	71.6	-
Ours	Mistral	7B	84.2	38.7
ICL	Qwen2.5-Math	7B	57.7	<u>52.1</u>
SFT	Qwen2.5-Math	7B	79.4	49.1
Ours	Qwen2.5-Math	7B	91.7	61.2
ICL	DeepSeekMath	7B	65.7	33.4
SFT	DeepSeekMath	7B	67.2	30.9
DART-Math	DeepSeekMath	7B	<u>88.2</u>	<u>52.9</u>
Ours	DeepSeekMath	7B	90.0	54.8

Table 2: Accuracy results (%) of the compared methods on GSM8K and MATH datasets (ICL: In-context learning, SFT: Supervised finetuning on the training set of GSM8K or MATH). Results of baselines are retrieved from original papers. The **bold** scores indicate the best results and underlined scores indicate the second best results.

First, compared to the open-source models without tools, LoRID outperforms all other baselines across all datasets, except MathScale. Specifically, on the GSM8K dataset, it achieves accuracy improvements of 2.3%, 16.1%, 2.4%,

12.3%, and 1.8% over the second-best methods when deployed on the LLaMA-2-7B, LLaMA-3-8B, Mistral-7B, Qwen2.5-Math-7B, and DeepSeekMath-7B base models respectively. This demonstrates that our method benefits from the interaction between System 1 and System 2, and is more effective than others based solely on CoT data augmentation. Furthermore, considering that our method is implemented based on LoRA blocks, we can choose better data augmentation methods to train System 1 and flexibly try different LoRA combinations, so there is still potential for performance improvement. Although MathScale (31.1%) has higher accuracy than our method (25.2%) on the MATH dataset, their training data size is approximately 2 million, much more than we require.

Second, LoRID achieves significant performance improvements on different base models, including general models such as LLaMA-3-8B and math-specialized models such as DeepSeekMath-7B. On the GSM8K dataset, our method outperforms the zero-shot context learning method by 63.9%, 29.5%, 68.7%, 34.0%, and 24.3% on the LLaMA-2-7B, LLaMA-3-8B, Mistral-7B, Qwen2.5-Math-7B, and DeepSeekMath-7B base models respectively. Additionally, compared to closed-source LLMs with hundreds of billions of parameters, the open-source models trained based on our method are already close in mathematical reasoning capabilities (e.g., 91.7% on Qwen2.5-Math-7B vs. 92.9% on GPT-4o). It shows that imitating the way teachers impart knowledge in CoT distillation is effective, and the student model even surpasses the teacher in some capabilities.

Finally, the experimental results demonstrate that LoRID has consistent improvements on both the GSM8K dataset, which emphasizes natural language understanding, and the MATH dataset, which focuses on mathematical calculations. However, taking Mistral-7B as an example, on the GSM8K dataset, the accuracy of our method is 3.7% higher than the best open-source models with tools, but 6.4% lower on the MATH dataset. This indicates that for datasets (e.g., MATH) involving complex calculations, tool-based methods have certain advantages due to leveraging the capabilities of external tools. For datasets (e.g., GSM8K) that emphasize knowledge reasoning but involve simple calculations, their performance is inferior to the method we proposed, suggesting that their reasoning abilities remain insufficient.

4.3 Ablation Results

We conduct ablation experiments on the GSM8K and MATH datasets, where System 1 is trained on the MuggleMath, MuMath, MetaMath, and RFT augmented datasets, and System 2 is trained on the knowledge-enhanced reasoning dataset we constructed. As shown in Table 3, it is noticed that LoRID outperforms the methods trained with only CoT (w/o System 2) by 4.4-25.0% on LLaMA-2-7B and 3.6-22.4% on Mistral-7B across all datasets. This indicates that for some problems, students need to first learn the knowledge and then apply it to answer (i.e., System 2). Additionally, LoRID achieves higher accuracy than methods that do not use System 1, which demonstrates that the integration of both systems is necessary. The LoRA blocks, trained on the same student model, provide the foundation for implementing this interaction. Fi-

Method		LLaMA-2-7B		Mistral-7B	
		GSM8K	MATH	GSM8K	MATH
MuggleMath	LoRID	0.785	0.252	0.832	0.387
	w/o System 1	0.597	0.148	0.667	0.217
	w/o System 2	0.741	0.201	0.789	0.351
MuMath	LoRID	0.783	0.231	0.842	0.352
	w/o System 1	0.597	0.148	0.667	0.217
	w/o System 2	0.700	0.151	0.773	0.259
MetaMath	LoRID	0.726	0.203	0.785	0.316
	w/o System 1	0.597	0.148	0.667	0.217
	w/o System 2	0.647	0.124	0.679	0.221
RFT	LoRID	0.682	-	0.743	-
	w/o System 1	0.597	-	0.667	-
	w/o System 2	0.432	-	0.519	-

Table 3: Ablation results on the LLaMA-2-7B and Mistral-7B base student model. Since RFT has not augmented data for the MATH dataset, there are no related experimental results.

nally, taking the GSM8K dataset as an example, the performance of Mistral-7B in the LoRID, System 1, and System 2 is improved by 5.7%, 7.0%, and 6.0% compared with LLaMA-2-7B. The performance gain brought by the base model itself is consistent across each module of our method.

4.4 Discussions

Analysis of Scaling Laws

We use LLaMA-2-7B and Mistral-7B as our base model to study the scaling laws of LoRID. In the experiment, System 1 is trained with augmented data from MuggleMath, MuMath, MetaMath, and RFT, with data sizes set to 7.5k, 40k, 80k, and 140k, respectively. The training data sizes for the Knowledge Generator and Deep Reasoner in System 2 are also consistent with those of System 1. Table 4 shows that, with the same number of training samples, our approach outperforms those that rely solely on System 1 for reasoning, after incorporating System 2. Using 40k samples, LoRID consistently achieves better results than other methods that use 140k samples. Additionally, we observe that as the data size increases, the performance of our method shows an upward trend in most cases, but eventually reaches a plateau, which is consistent with the findings of most other works [Li *et al.*, 2024a]. When the sample sizes are 7.5k, 40k, 80k, and 140k, our method achieves an average accuracy improvement of 11.8%, 11.0%, 9.1%, and 5.6% compared to the baselines, respectively. This suggests that LoRID may have greater potential for application in low-resource settings.

Analysis of Problem Difficulty

We investigate the effectiveness of LoRID on problems with varying difficulties, with experiments conducted on Mistral-7B, while System 1 is trained on the MetaMath augmented dataset. The GSM8K is categorized by the number of reasoning steps, while the MATH has five levels of difficulty, ranging from low to high. In Figure 4, across all levels of problem difficulty, our method improves the reasoning accuracy by an average of 10.6% and 11.8% compared to System 1 and System 2, respectively. This indicates that the integration of two thinking modes enabled by LoRA blocks contributes to the

Method	LLaMA-2-7B				Mistral-7B			
	7.5k	40k	80k	140k	7.5k	40k	80k	140k
MuggleMath w/ S-1	0.562	0.689	0.719	0.731	0.738	0.776	0.793	0.808
MuggleMath w/ S-1&2	0.637	0.742	0.762	0.778	0.798	0.832	0.829	0.821
MuMath w/ S-1	0.470	0.596	0.653	0.694	0.661	0.719	0.762	0.776
MuMath w/ S-1&2	0.600	0.699	0.744	0.763	0.770	0.819	0.810	0.812
MetaMath w/ S-1	0.462	0.590	0.616	0.636	0.632	0.703	0.716	0.696
MetaMath w/ S-1&2	0.592	0.691	0.699	0.731	0.748	0.795	0.777	0.772
RFT w/ S-1	0.419	0.443	0.486	-	0.541	0.576	0.559	-
RFT w/ S-1&2	0.566	0.638	0.663	-	0.720	0.757	0.745	-

Table 4: Performance of LoRID using different sizes of training data on the GSM8K dataset (S-1: System 1, S-2: System 2). Since the augmented dataset of RFT is less than 140k, there are no relevant experimental results.

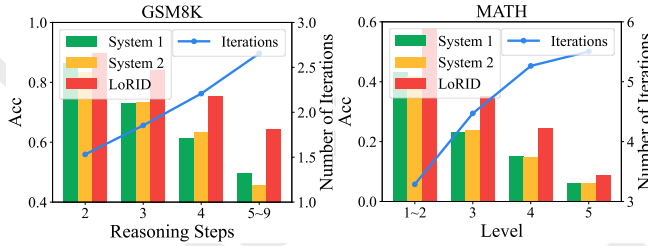


Figure 4: Performance of LoRID on different problem difficulties.

improvement of the student model’s mathematical reasoning ability. Furthermore, we observe that more difficult problems require more iterations. This aligns with the common sense: students tend to engage in multiple rounds of self-reflection and correction when facing hard problems.

Analysis of Inference Cost

We analyze the performance of LoRID and Self-consistent CoT (SC-CoT) [Wang *et al.*, 2023], then further demonstrate our feasibility. In Table 5, our method achieves an accuracy improvement of 9.2% and 12.4% compared to System 1 and System 2, respectively, which perform inference once. We trade a small increase in inference time for improved accuracy, a concept that has recently been adopted by some large models, such as OpenAI o1 [Jaech *et al.*, 2024]. Furthermore, LoRID even achieves comparable performance to SC-CoT ($k=10$) on both models, which shows that the interaction between System 1 and System 2 is more efficient in terms of inference overhead than the interaction within a single system alone (e.g., two Intuitive Reasoners). We speculate that System 1 is better suited for handling problems based on experience and intuition, while System 2 is more suitable for problems grounded in reasoning and logic. The preferences of the two systems for solving problems differ, and this will be explored in future work.

Case Study

We conduct a case study to verify that System 2 can compensate for the errors caused by intuition in System 1. As shown in Table 6, System 1 makes consecutive errors in two steps due to a lack of deep understanding of the problem and logical analysis. In System 2, our Knowledge Generator correctly

Method		LLaMA-2-7B		Mistral-7B	
		Acc \uparrow	# Iter \downarrow	Acc \uparrow	# Iter \downarrow
System 1	SC-CoT (k=1)	0.649	1	0.679	1
	SC-CoT (k=5)	0.718	5	0.757	5
	SC-CoT (k=10)	0.739	10	0.782	10
System 2	SC-CoT (k=1)	0.597	1	0.667	1
	SC-CoT (k=5)	0.667	5	0.732	5
	SC-CoT (k=10)	0.704	10	0.754	10
LoRID		0.727	2.3	0.785	2.1

Table 5: Performance of LoRID and Self-consistent CoT on the GSM8K dataset.

Q	Last Friday, 13 of the 82 teachers at Rydell Elementary School were sick. There were 9 substitute teachers called in to help. How many teachers were at school that day?
GT	There were $82-13=69$ regular teachers at school. If we add the substitute teachers, we find there were $69+9=78$ teachers at school that day.
S-1	There were 13 sick teachers and 9 substitute teachers, so there were a total of $13+9=22$ teachers not available. Out of the 82 total teachers, 22 were not available, so there were $82-22=60$ teachers at school that day. ✗
S-2	Subtract the number of incomplete items from the total to find the complete items. Add the number of additional items to the remaining items to find the total. There were $82-13=69$ teachers at Rydell Elementary School that day. In addition to the 69 teachers, there were 9 substitute teachers. So, the total number of teachers at school that day was $69+9=78$. ✓

Table 6: Case study of LoRID on the GSM8K dataset (Q: Question, GT: Ground Truth).

outputs the steps of subtracting first and then adding, allowing Deep Reasoner to obtain the correct answer based on this. Similarly, System 1 can also reduce errors in System 2 caused by incorrect associations of knowledge or wrong application of it. Due to limited space, further details are not elaborated.

5 Conclusion

In this work, we propose a novel method LoRID with multi-LoRA interaction, which improves the mathematical reasoning performance of student language models like human beings teaching and learning pattern. LoRID explicitly extracts the knowledge of teacher models in the data augmentation stage, and fully utilizes the consistency of System 1 and System 2 in the model distillation stage. Experimental results show that LoRID outperforms the state-of-the-art methods and can be effectively integrated into any CoT distillation model. In the future, we will explore the following directions: (1) We will apply the idea of interaction between knowledge and reasoning during the training phase to reduce the inference overhead of models, such as introducing reinforcement learning [Rafailov *et al.*, 2023]. (2) We will use external tools (e.g., compilers) in our approach so that the knowledge generator, reasoning generator, and code generator can verify each other and reduce computational errors to a certain degree.

Acknowledgments

We thank the reviewers for their insightful comments. This work was supported by National Science Foundation of China (Grant Nos.62376057). All opinions are of the authors and do not reflect the view of sponsors.

References

- [Achiam *et al.*, 2023] Josh Achiam, Steven Adler, Sandhini Agarwal, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [Brown *et al.*, 2020] Tom B. Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In *NeurIPS*, 2020.
- [Chaslot *et al.*, 2008] Guillaume Chaslot, Sander Bakkes, Istvan Szita, et al. Monte-carlo tree search: A new framework for game ai. In *AAAI*, 2008.
- [Chen *et al.*, 2021] Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.
- [Chen *et al.*, 2024a] Changyu Chen, Xiting Wang, Ting-En Lin, et al. Masked thought: Simply masking partial reasoning steps can improve mathematical reasoning learning of language models. In *ACL*, 2024.
- [Chen *et al.*, 2024b] Guoxin Chen, Minpeng Liao, Chengxi Li, et al. Alphamath almost zero: process supervision without process. In *NeurIPS*, 2024.
- [Chen *et al.*, 2024c] Zhaorun Chen, Zhuokai Zhao, Zhihong Zhu, et al. Autoprmm: Automating procedural supervision for multi-step reasoning via controllable question decomposition. In *NAACL*, 2024.
- [Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [Gou *et al.*, 2024] Zhibin Gou, Zhihong Shao, Yeyun Gong, et al. ToRA: A tool-integrated reasoning agent for mathematical problem solving. In *ICLR*, 2024.
- [Grattafiori *et al.*, 2024] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.
- [Hendrycks *et al.*, 2021] Dan Hendrycks, Collin Burns, Saurav Kadavath, et al. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021.
- [Hu *et al.*, 2022] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.
- [Hu *et al.*, 2024] Jun Hu, Wenwen Xia, Xiaolu Zhang, et al. Enhancing sequential recommendation via llm-based semantic embedding learning. In *WWW*, 2024.
- [Huang *et al.*, 2024] Jinfeng Huang, Qiaoqiao She, Wenbin Jiang, et al. Qdmmr-based planning-and-solving prompting for complex reasoning tasks. In *COLING*, 2024.
- [Jaech *et al.*, 2024] Aaron Jaech, Adam Kalai, Adam Lerer, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.
- [Jiang *et al.*, 2023] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [Kahneman, 2011] Daniel Kahneman. Thinking, fast and slow. *Farrar, Straus and Giroux*, 2011.
- [Kojima *et al.*, 2022] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, et al. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.
- [Li *et al.*, 2024a] Chengpeng Li, Zheng Yuan, Hongyi Yuan, et al. Mugglemath: Assessing the impact of query and response augmentation on math reasoning. In *ACL*, 2024.
- [Li *et al.*, 2024b] Zenan Li, Zhi Zhou, Yuan Yao, et al. Neuro-symbolic data generation for math reasoning. In *NeurIPS*, 2024.
- [Longpre *et al.*, 2023] Shayne Longpre, Le Hou, Tu Vu, et al. The flan collection: Designing data and methods for effective instruction tuning. In *ICML*, 2023.
- [Loshchilov and Hutter, 2017] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [Lu *et al.*, 2024] Zimu Lu, Aojun Zhou, Houxing Ren, et al. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. In *ACL*, 2024.
- [Luo *et al.*, 2023] Haipeng Luo, Qingfeng Sun, Can Xu, et al. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.
- [Magister *et al.*, 2023] Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, et al. Teaching small language models to reason. In *ACL*, 2023.
- [Rafailov *et al.*, 2023] Rafael Rafailov, Archit Sharma, Eric Mitchell, et al. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.
- [Ranaldi and Freitas, 2024] Leonardo Ranaldi and Andr e Freitas. Self-refine instruction-tuning for aligning reasoning in language models. In *EMNLP*, 2024.
- [Shao *et al.*, 2024] Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [Tang *et al.*, 2023] Xiaojuan Tang, Zilong Zheng, Jiaqi Li, et al. Large language models are in-context semantic reasoners rather than symbolic reasoners. *arXiv preprint arXiv:2305.14825*, 2023.
- [Tang *et al.*, 2024] Zhengyang Tang, Xingxing Zhang, Benyou Wang, et al. Mathscale: Scaling instruction tuning for mathematical reasoning. In *ICML*, 2024.
- [Team *et al.*, 2023] Gemini Team, Rohan Anil, Sebastian Borgeaud, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [Tong *et al.*, 2024] Yuxuan Tong, Xiwen Zhang, Rui Wang, et al. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. In *NeurIPS*, 2024.

- [Toshniwal *et al.*, 2024] Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, et al. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. In *NeurIPS*, 2024.
- [Touvron *et al.*, 2023] Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [Wang *et al.*, 2019] Wei Wang, Vincent W. Zheng, Han Yu, et al. A survey of zero-shot learning: Settings, methods, and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2):1–37, 2019.
- [Wang *et al.*, 2023] Xuezhi Wang, Jason Wei, Dale Schuurmans, et al. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023.
- [Wang *et al.*, 2024a] Ke Wang, Houxing Ren, Aojun Zhou, et al. Mathcoder: Seamless code integration in LLMs for enhanced mathematical reasoning. In *ICLR*, 2024.
- [Wang *et al.*, 2024b] Peiyi Wang, Lei Li, Zhihong Shao, et al. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *ACL*, 2024.
- [Wang *et al.*, 2024c] Tianduo Wang, Shichen Li, and Wei Lu. Self-training with direct preference optimization improves chain-of-thought reasoning. In *ACL*, 2024.
- [Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.
- [Wu *et al.*, 2024] Xuansheng Wu, Wenlin Yao, Jianshu Chen, et al. From language modeling to instruction following: Understanding the behavior shift in LLMs after instruction tuning. In *NAACL*, 2024.
- [Xi *et al.*, 2024] Zhiheng Xi, Wenxiang Chen, Boyang Hong, et al. Training large language models for reasoning through reverse curriculum reinforcement learning. In *ICML*, 2024.
- [Yang *et al.*, 2024] An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [Yin *et al.*, 2024] Shuo Yin, Weihao You, Zhilong Ji, et al. Mumath-code: Combining tool-use large language models with multi-perspective data augmentation for mathematical reasoning. *arXiv preprint arXiv:2405.07551*, 2024.
- [You *et al.*, 2024] Weihao You, Shuo Yin, Xudong Zhao, et al. MuMath: Multi-perspective data augmentation for mathematical reasoning in large language models. In *NAACL*, 2024.
- [Yu *et al.*, 2024a] Fei Yu, Anningzhe Gao, and Benyou Wang. OVM, outcome-supervised value models for planning in mathematical reasoning. In *NAACL*, 2024.
- [Yu *et al.*, 2024b] Longhui Yu, Weisen Jiang, Han Shi, et al. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*, 2024.
- [Yuan *et al.*, 2023] Zheng Yuan, Hongyi Yuan, Chengpeng Li, et al. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.
- [Yue *et al.*, 2024] Xiang Yue, Xingwei Qu, Ge Zhang, et al. Mammoth: Building math generalist models through hybrid instruction tuning. In *ICLR*, 2024.
- [Zhang *et al.*, 2024a] Di Zhang, Xiaoshui Huang, Dongzhan Zhou, et al. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*, 2024.
- [Zhang *et al.*, 2024b] Zhihan Zhang, Tao Ge, Zhenwen Liang, et al. Learn beyond the answer: Training language models with reflection for mathematical reasoning. In *EMNLP*, 2024.
- [Zhou *et al.*, 2024] Aojun Zhou, Ke Wang, Zimu Lu, et al. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. In *ICLR*, 2024.