SAAS: Solving Ability Amplification Strategy for Enhanced Mathematical Reasoning in Large Language Models

Hyeonwoo Kim[1*], **Gyoungjin Gim**[1*], **Yungi Kim**[1*]
Jihoo Kim[1], **Byungju Kim**[2], **Wonseok Lee**[2],
Chanjun Park[1†]

[1] Upstage AI, [2] Mathpresso Inc.
{choco_9966, gyoungjin.gim, eddie, jerry, chanjun.park}@upstage.ai
{peyton.kim, jack.lee}@mathpresso.com

Abstract

This study presents a novel learning approach designed to enhance both mathematical reasoning and problem-solving abilities of Large Language Models (LLMs). We focus on integrating the Chain-of-Thought (CoT) and the Program-of-Thought (PoT) learning, hypothesizing that *prioritizing the learning of mathematical reasoning ability is helpful for the amplification of problem-solving ability*. Thus, *the initial learning with CoT* is essential for solving challenging mathematical problems. To this end, we propose a sequential learning approach, named **SAAS** (Solving Ability Amplification Strategy), which strategically transitions from CoT learning to PoT learning. Our empirical study, involving an extensive performance comparison using several benchmarks, demonstrates that our **SAAS** achieves state-of-the-art (SOTA) performance. The results underscore the effectiveness of our sequential learning approach, marking a significant advancement in the field of mathematical reasoning in LLMs.

1 Introduction

The advent of Large Language Models (LLMs) has marked a significant breakthrough in various domains. However, despite their remarkable performance across these domains, a notable challenge persists in the realm of mathematical reasoning (Zhao et al., 2023; Lu et al., 2022b; Meadows and Freitas, 2022; Qian et al., 2022; Zhou et al., 2022; Lightman et al., 2023; Drori et al., 2021; Zhang et al., 2019). The ability of LLMs to comprehend, interpret, and manipulate mathematical concepts is not yet on par with their linguistic capabilities.

The significance of mathematical reasoning in LLMs involves more than just crunching numbers. It also encompasses the ability to engage in logical thinking, problem-solving, and complex decision-making, which are essential for understanding and generating human-like responses in the different situations (Lu et al., 2022b; Meadows and Freitas, 2022; Thawani et al., 2021). In other words, mathematical reasoning in LLMs is essential for a comprehensive understanding and manipulation of language in numerous scientific and practical applications. However, the current ability of LLMs in mathematical reasoning hinder their potential in the fields where numerical and logical comprehension are paramount such as coding. Thus, it's critical challenge to enhance the ability of LLMs in mathematical reasoning.

In this study, we explore a learning approach for enhancing both mathematical reasoning ability and problem-solving ability in LLMs, focusing on learning with both the Chain-of-Thought (CoT) (Wei et al., 2022b) and the Program-of-Thought (PoT) (Chen et al., 2022; Gao et al., 2023a). The CoT rationale (Figure 1-(a)) consists of a series of intermediate reasoning steps. Although it enhances the reasoning ability of LLMs, it leads to arithmetic calculation errors when dealing with large numbers (Chen et al., 2022), resulting a low problem-solving ability. To address this issue, Chen et al. (2022) proposed the PoT rationale (Figure 1-(b)), which expresses the reasoning steps as code and delegate computation steps to an code interpreter. It requires the reasoning steps to be expressed *accurately* as code. Therefore, we hypothesize that *prioritizing the learning of mathematical reasoning ability is helpful for the amplification of problem-solving ability*. In other words, *the initial learning with CoT* is essential for solving challenging mathematical problems, since it improves the mathematical reasoning ability (Magister et al., 2022; Shridhar et al., 2023; Jie et al., 2023; Liang et al., 2023).

Our research is motivated by an analysis of existing models (Gou et al., 2023; Yue et al., 2023).

*Equal Contribution † Corresponding Author

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 186–198
November 12-16, 2024 ©2024 Association for Computational Linguistics

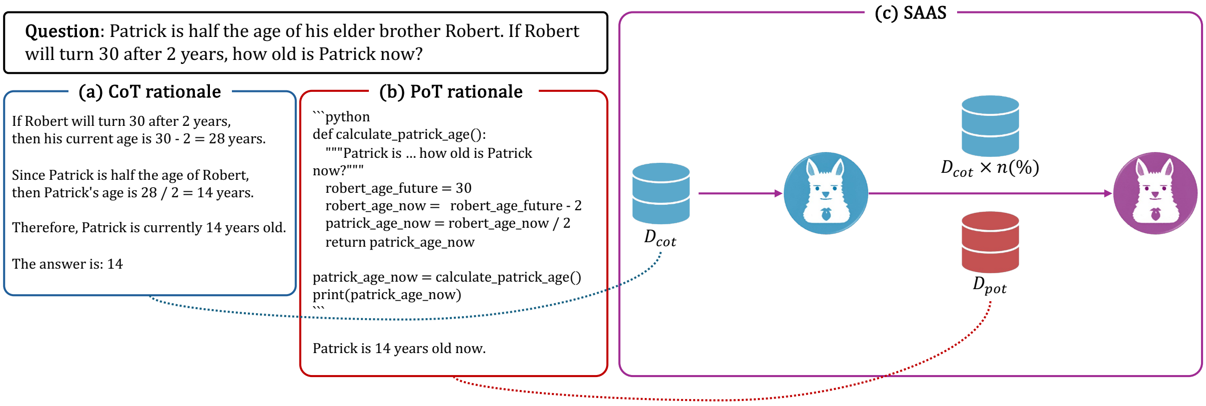

Figure 1: Overview of **SAAS** (Solving Ability Amplification Strategy) with two core strategies: i) sequential learning strategy; ii) cognitive retention strategy.

ToRA (Gou et al., 2023) tried to learn reasoning ability as well as PoT by adding reasoning step into the PoT rationale. Similarly, MAmmoTH (Yue et al., 2023) tried to learn both CoT and PoT by using both CoT rationale and PoT rationale as training data simultaneously. However, we conjecture that they do not fully utilize the advantages of learning with both CoT and PoT. This is because they did not consider *the sequence of CoT learning and PoT learning*, resulting a less effective learning.

In this work, we introduce a sequential learning approach, named **SAAS** (Solving Ability Amplification Strategy), to effectively utilize the strengths of CoT learning and PoT learning. This approach transitions from CoT learning to PoT learning, focusing on *enhancing problem-solving ability in PoT learning based on logical skills established in CoT learning*. This pedagogical strategy ensures that the competencies developed during CoT learning positively influence the PoT learning phase, leading to an overall improvement in solving challenging mathematical problems.

We validate the rationality and effectiveness of our **SAAS** via extensive experiments on the reputable benchmarks (Cobbe et al., 2021; Hendrycks et al., 2021; Gao et al., 2023b; Patel et al., 2021; Miao et al., 2021; Lu et al., 2022a; Koncel-Kedziorski et al., 2016). Most importantly, **SAAS** achieved state-of-the-art with remarkable performance. Through this, in this paper, we present a *novel and effective perspective* (*i.e.*, our hypothesis) within the field of mathematics.

2 SAAS: Solving Ability Amplification Strategy

In this paper, we hypothesize that learning about the problem-solving ability is more effective after logical skills are well established. To explore this, we propose the sequential learning approach, named **SAAS** (Solving Ability Amplification Strategy), which transitions from CoT learning to PoT learning as shown in Figure 1. Our **SAAS** is motivated by the pedagogical strategy of human that first learns logical skills and then develops problem-solving abilities by solving numerous problems (Glaser, 1984). In the following subsections, we describe CoT learning and PoT learning in details.

2.1 Chain-of-Thought Learning

It has been shown in various domains that CoT learning, which trains LLMs with data composed of CoT rationales, improves reasoning ability (Jie et al., 2023; Liang et al., 2023). Thus, we first fine-tune the LLM via CoT learning for improving mathematical reasoning ability. The primary objective in this phase is to optimize the model parameters for logically interpreting and responding to mathematical problems.

To achieve this, we employ a widely used optimization approach (Yu et al., 2023; Gou et al., 2023) that seeks to find the optimal parameters, denoted as θ^*_{cot}, which minimize the negative log-likelihood. This is expressed mathematically as:

$$\arg\min_{\theta} -\frac{1}{|D_{cot}|} \sum_{(x_{cot}, y_{cot}) \in D_{cot}} \log p_{\theta}(y_{cot}|x_{cot}), \quad (1)$$

where θ represents the learnable parameters of the LLM. The dataset D_{cot} consists of (x_{cot}, y_{cot}) pairs, where x_{cot} denotes a mathematical question, and y_{cot} is the desired CoT rationale for that question.

This optimization process is designed to ensure that the model learns to generate CoT rationales that are logically consistent throughout the reasoning process. This is particularly important in the field of mathematics, since the rationale behind each step is as critical as the final answer. By minimizing the negative log-likelihood, we effectively guide the model to generate step-by-step explanations that mirror human problem-solving approaches, thus enhancing its overall reasoning capability.

This phase sets the foundation for the subsequent PoT learning phase, where the model's enhanced reasoning ability, developed through CoT training, is further refined and applied to more complex problem-solving scenarios.

2.2 Program-of-Thought Learning

Although the LLM optimized with parameters θ^*_{cot} demonstrates improved logical skills, it still exhibits limitations in problem-solving ability, particularly in computational accuracy (Chen et al., 2022), which will be empirically validated in section 3.2.4. To amplify this problem-solving ability, building upon the mathematical reasoning established in the CoT learning phase, we further fine-tune the LLM with θ^*_{cot} as its starting point using data composed of PoT rationales.

To accomplish this, we construct a dataset $D_{pot+cot}$ that consists of both PoT and CoT rationales. Notably, we integrate CoT rationales alongside PoT rationales in this dataset. This is because we observed that focusing exclusively on PoT rationales during this phase leads to a deterioration in mathematical reasoning ability in our experiments, as detailed in Table 3. To mitigate this *cognitive forgetting*, we introduce a *cognitive retention strategy*. This strategy involves randomly sampling CoT rationales and incorporating them into the PoT learning phase. Such a mixed approach (*i.e.*, congnitive retention strategy) ensures that the LLM retains its previously acquired reasoning skills while adapting to the new learning format.

The objective in this phase is to find the final optimal parameters θ^* of the LLM, which involves minimizing the following negative log-likelihood:

$$\arg\min_{\theta^*_{cot}} -\frac{1}{|D_{pot+cot}|} \sum_{(x,y)\in D_{pot+cot}} \log p_{\theta^*_{cot}}(y|x), \quad (2)$$

where x represents a mathematical question, and y is the desired output, which could be either a PoT rationale or a CoT rationale, for the given question x. This approach aims to harmonize the strengths of both CoT and PoT learning, thereby equipping the LLM with enhanced computational accuracy and problem-solving abilities while maintaining its proficiency in logical reasoning.

Seed Dataset	Rationale	Models	Size
MetaMathQA	CoT	GPT, WizardMath	465K
MATH, GSM8K	CoT	WizardMath	300K
QANDA	CoT	WizardMath	120K
MetaMathQA	PoT	ToRA	60K
MATH, GSM8K	PoT	ToRA	226K
MathInstruct	PoT	ToRA	38K
QANDA	PoT	ToRA	12K

Table 1: Summary of synthetic datasets

3 Experiments

In this section, we conduct extensive experiments to answer the following key research questions (RQs):

- **RQ1**: Does SAAS quantitatively outperform its competitors for solving challenging mathematical problems?
- **RQ2**: Are two core strategies of SAAS (sequential learning, cognitive retention strategy) effective in improving the accuracy?
- **RQ3**: Is SAAS effective in solving not only basic but also challenging mathematical problems?
- **RQ4**: Does sequential learning that transitions from CoT learning to PoT learning help improve both the mathematical reasoning and computational accuracy?

3.1 Experimental Settings

3.1.1 Dataset Details

In this paper, we synthesize GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MetaMathQA (Yu et al., 2023), MathInstruct (Yue et al., 2023), and QANDA. The QANDA dataset was gathered manually through direct interaction with the application[1]. The overall procedure of synthetic data generation is illustrated in Figure 2.

Specifically, we synthesize these datasets into Chain-of-Thought (CoT) and Program-of-Thought (PoT) rationales via various models (GPT, WizardMath (Luo et al., 2023), ToRA (Gou et al., 2023)).

[1]https://mathpresso.com/en

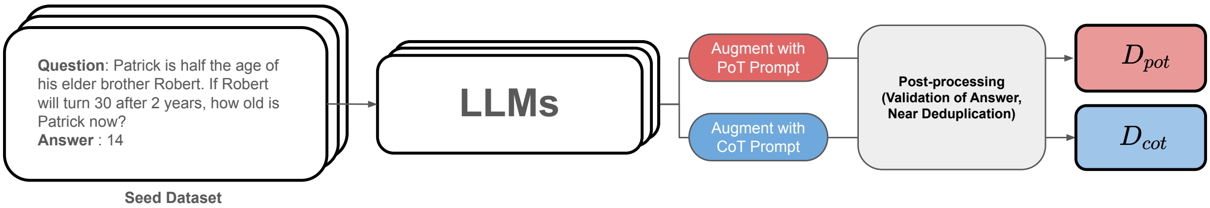

Figure 2: Overall procedure of the synthetic data generation.

To generate diverse synthetic data, we adjust some hyperparameters such as temperature and top_p. Then, we select only the correct responses and eliminate similar ones among these correct responses as in Wang et al. (2022). The detailed descriptions of seed datasets are described in Appendix B. Table 1 provides the summary of our synthetic datasets for fine-tuning.

3.1.2 Training Details

We used the CodeLLaMA 13B model (Roziere et al., 2023) as our base model and fine-tuned it with our synthetic datasets by setting the batch size to 128. We set learning rate to $2e-5$ and use cosine scheduler with warm-up period (1 epoch). For efficient model training, we used DeepSpeed ZeRO Stage3 (Rajbhandari et al., 2020).

3.1.3 Model Details

To evaluate the effectiveness of our SAAS in RQ1, we compared it with several state-of-the-art competitors. These competitors are divided into two groups: general models and mathematics domain-specific models. The general models include GPT-4 (Achiam et al., 2023), ChatGPT (gpt-3.5-turbo)(OpenAI, 2023), Claude-2(Anthropic, 2023), PaLM-2 (Anil et al., 2023), LLaMA-2 (Touvron et al., 2023), Platypus-2 (Lee et al., 2023), CodeLLaMA (Roziere et al., 2023), and SOLAR-1 (Kim et al., 2023). The mathematics domain-specific models consist of WizardMath (Luo et al., 2023), MetaMath (Yu et al., 2023), MulggleMath (Li et al., 2023a), Toolformer (Schick et al., 2023), MathCoder (Wang et al., 2023), MammoTH (Yue et al., 2023), and ToRA (Gou et al., 2023).

As in Gou et al. (2023), we report CoT prompting results by default, and include PAL (Gao et al., 2023a) prompting results for selected models. Within the category of mathematics domain-specific models, WizardMath, MetaMath, and MuggleMath exclusively employ CoT learning for fine-tuning. Conversely, ToRA utilizes solely PoT learning, whereas MathCoder and MammoTh integrate a combination of CoT and PoT learning methodologies for fine-tuning. Also, Toolformer is trained to utilize calculators.

3.1.4 Evaluation Details

We evaluated the model's performance and its ability to generalize mathematical reasoning using both in-domain and out-of-domain data. For in-domain evaluation, we use the test set of MATH and GSM8K dataset. For out-of-domain evaluation, we utilized the following various datasets, which are used in the previous studies (Gou et al., 2023; Yue et al., 2023) and publicly available: GSM-Hard (Gao et al., 2023b), SVAMP (Patel et al., 2021), ASDIV (Miao et al., 2021), TabMWP (Lu et al., 2022a), and MAWPS (Koncel-Kedziorski et al., 2016) that consists of SingleEQ, SingleOP, AddSub, and MultiArith. These datasets ensure a comprehensive analysis of the model's applicability across various mathematical contexts.

3.2 Results and Analysis

We highlight the best and the second-best results in each column (*i.e.*, dataset) of the following tables in bold and underline, respectively.

3.2.1 RQ1: Comparison with Competitors

To demonstrate the superiority of our **SAAS** over competitors, we compare the accuracies of all competitors and **SAAS**. In this experiment, we utilize LLaMA-2 7B, CodeLLaMA 7B, SOLAR-1 10.7B, LLaMA-2 13B, CodeLLaMA 13B, CodeLLaMA 34B, and Llemma-34B as our base models.[2]

Table 2 shows the results. We summarize our empirical findings as follows. First, we observed that mathematics domain-specific models outperforms general models *with similar size* in almost cases. This indicates a requisite for domain-specific models to address complex mathematical problems effectively. Second, among mathematics domain-specific competitors, ToRA, which utilizes solely

[2]For experiment on the 70B model, we could not proceed it due to hardware constraint.

Model	Size	GSM8K	MATH	GSM-Hard	SVAMP	TabMWP	ASDiv	MAWPS	Avg.
General Models									
GPT-4	-	92.0	45.2	64.7	93.1	67.1	91.3	97.6	78.3
GPT-4 (PAL)	-	94.2	51.8	77.6	94.8	95.9	92.6	97.7	86.4
ChatGPT	-	80.8	35.5	55.9	83.0	69.1	87.3	94.6	72.3
ChatGPT (PAL)	-	78.6	38.7	67.6	77.8	79.9	81.0	89.4	73.3
Claude-2	-	85.2	32.5	-	-	-	-	-	-
PaLM-2	540B	80.7	34.3	-	-	-	-	-	-
LLaMa-2	7B	13.3	4.1	7.8	38.0	31.1	50.7	60.9	29.4
Platypus-2	7B	14.4	5.4	8.6	36.7	26.5	47.9	58.4	28.3
CodeLLaMa (PAL)	7B	34.0	16.6	33.6	59.0	47.3	61.4	79.6	47.4
SOLAR-1	10.7B	25.8	8.0	17.1	59.3	33.6	55.1	68.4	38.1
LLaMa-2	13B	24.3	6.3	13.6	43.1	39.5	56.3	70.4	36.2
Platypus-2	13B	23.7	7.1	14.3	50.7	45.3	55.1	69.6	38.0
CodeLLaMa (PAL)	13B	39.9	19.9	39.0	62.4	59.5	65.3	86.0	53.1
CodeLLaMa (PAL)	34B	53.3	23.9	49.4	71.0	63.1	72.4	91.5	60.7
LLaMa-2	70B	57.8	14.4	36.0	73.6	57.5	76.0	92.4	58.2
Platypus-2	70B	45.9	15.0	24.6	74.3	47.3	72.7	91.1	53.0
Mathematics Domain-Specific Models									
WizardMath	7B	54.9	10.7	20.6	57.3	38.1	59.1	73.7	44.9
MetaMath	7B	66.5	19.8	-	-	-	-	-	-
MuggleMATH	7B	68.4	-	-	-	-	-	-	-
Toolformer	7B	-	-	-	29.4	-	40.4	44.0	-
MathCoder	7B	64.2	23.3	-	-	-	-	-	-
MathCoder-CODE	7B	67.8	30.2	-	-	-	-	-	-
MAmmoTH	7B	53.6	31.5	-	-	-	-	-	-
MAmmoTH-CODE	7B	59.4	33.4	-	-	-	-	-	-
ToRA	7B	68.8	40.1	54.6	68.2	42.4	73.9	88.8	62.4
SAAS	7B	74.3	43.2	**58.3**	**74.3**	49.6	77.3	93.6	67.2
ToRA-CODE	7B	72.6	44.6	56.0	70.4	51.6	78.7	91.3	66.5
SAAS-CODE	7B	**74.8**	**45.2**	58.1	73.6	**64.0**	**80.4**	**93.8**	**70.0**
SAAS	10.7B	**82.0**	50.1	**64.9**	**85.0**	**72.5**	**87.5**	**95.7**	**76.8**
WizardMath	13B	63.9	14.0	28.4	64.3	46.7	65.8	79.7	51.8
MetaMath	13B	72.3	22.4	-	-	-	-	-	-
MuggleMATH	13B	74.0	-	-	-	-	-	-	-
MathCoder	13B	72.6	29.9	-	-	-	-	-	-
MathCoder-CODE	13B	74.1	35.9	-	-	-	-	-	-
MAmmoTH	13B	62.0	34.2	-	-	-	-	-	-
MAmmoTH-CODE	13B	64.7	36.3	-	-	-	-	-	-
ToRA	13B	72.7	43.0	57.3	72.9	47.2	77.2	91.3	65.9
SAAS	13B	76.6	46.2	61.6	77.8	58.2	80.5	94.3	70.7
ToRA-CODE	13B	75.8	48.1	60.5	75.7	65.4	81.4	92.5	71.3
SAAS-CODE	13B	79.4	**50.6**	61.6	80.6	68.2	84.5	95.4	74.3
MathCoder-CODE	34B	81.7	45.2	-	-	-	-	-	-
MAmmoTH-CODE	34B	72.7	43.6	-	-	-	-	-	-
ToRA-CODE	34B	80.7	50.8	63.7	80.5	70.5	84.2	93.3	74.8
SAAS-CODE	34B	82.9	52.3	64.1	82.8	73.9	85.4	95.2	76.6
SAAS-LLEMA	34B	**85.4**	**54.7**	**67.0**	**85.2**	**80.2**	**87.6**	**96.6**	**79.5**
WizardMath	70B	81.6	22.7	50.3	80.0	49.8	76.2	86.2	63.8
MetaMath	70B	82.3	26.6	-	-	-	-	-	-
MuggleMATH	70B	82.3	-	-	-	-	-	-	-
MathCoder	70B	83.9	45.1	-	-	-	-	-	-
ToRA	70B	84.3	49.7	67.2	82.7	74.0	86.8	93.8	76.9

Table 2: Accuracies of competitors and our **SAAS** on the mathematical benchmark datasets. Our **SAAS** models are shown in purple color.

PoT learning, *consistently* outperforms all others with similar size, including MathCoder and MammoTH, which integrate a combination of CoT learning and PoT learning methodologies. This implies that simply combining CoT and PoT learning does not effectively solve complex mathematical problems. Therefore, a strategic and careful approach is imperative in the combination of CoT and PoT

Strategy	GSM8K	MATH
Chain-of-Thought (CoT)	69.7	26.9
Program-of-Thought (PoT)	76.8	47.7
Combination of CoT and PoT	79.0	49.2
SAAS	**79.4**	**50.6**
without cognitive retention strategy	79.0	49.6
Reverse **SAAS**	76.8	47.1
without cognitive retention strategy	69.4	27.6

Table 3: Accuracies of different learning strategies. All improvements are statistically significant with p-value ≤ 0.001.

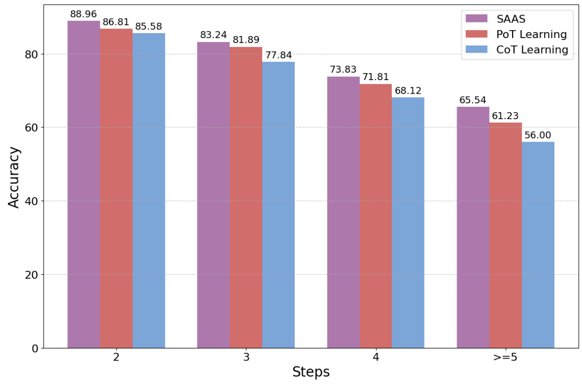

Figure 3: Accuracies on GSM8K with respect to the number of required reasoning steps.

learning. Third and most importantly, our **SAAS** *consistently* and *significantly* outperforms all competitors with similar size. Specifically, on ∼7B size, 7B∼13B size, 13B∼34B size, and 34B∼70B size, **SAAS** outperforms the best competitors (*i.e.*, ToRA-CODE and ToRA) by up to 5.26%, 7.71%, and 6.28% in terms of average score. Note that although we could not fine-tune 70B model, **SAAS** with 10.7B showed similar performance to ToRA with 70B. Furthermore, **SAAS-Llema** demonstrated superior performance than ToRA with 70B. This remarkable performance of **SAAS** underscore the effectiveness of our sequential learning approach.

3.2.2 RQ2: Effectiveness of Sequential Learning and Cognitive Retention Strategy

To further explore what factors contribute to the improvement of our **SAAS**, we conduct comparative experiments on diverse learning strategies, as shown in Table 3. Specifically, we compare CoT learning, PoT learning, CoT+PoT learning, **SAAS** that transtions from CoT learning to PoT learning, and reverse **SAAS** that transtions from PoT learning to CoT learning. In addition, we compare (reverse) **SAAS** *without cognitive retention strategy* to validate the effectiveness of this strategy. From Table 3, our empirical findings are summarized as follows:

i) **Effectiveness of the hybrid learning**: Combining of CoT and PoT learning significantly outperforms both CoT learning and PoT learning. This is because CoT learning, which enhances mathematical reasoning ability, and PoT learning, which improves problem-solving ability, play a complementary role;

ii) **Effectiveness of the sequential learning**: Our **SAAS** without cognitive retention strategy *slightly* outperforms combining of CoT and PoT learning in MATH only. We conjecture that the absence of significant improvement, despite sequential learning, can be attributed to the deterioration of mathematical reasoning abilities during the PoT learning phase (*i.e.*, cognitive forgetting). Furthermore, reverse **SAAS** without cognitive retention strategy shows a lower accuracy than combining of CoT and PoT learning. This result indicates that the order of the learning sequences in sequential learning is vital for mathematical reasoning and problem-solving abilities;

iii) **Effectiveness of the cognitive retention strategy**: To mitigate the cognitive forgetting, in Section 2.2, we proposed the cognitive retention strategy, which includes some data samples from first phase in the second phase. (Reverse) **SAAS** outperforms (reverse) **SAAS** without cognitive retention strategy, verifying the effectiveness of the cognitive retention strategy.

3.2.3 RQ3: Further Analysis of the Capabilities of SAAS

To analyze the capabilities of **SAAS** depending on the difficuly of mathemtical problem, we quantitatively assess the break-down accuracies for problems with respect to the reasoning steps as in Shi et al. (2023). Specifically, we segmented the GSM8K dataset into 4 categories based on the number of reasoning steps required to arrive at an answer. Then, we quantified accuracies of CoT learning, PoT learning, and SAAS across each designated category.

As illustrated in Figure 3, **SAAS** outperforms both CoT learning and PoT learning for all categories. Moreover, as the reasoning steps in a

Question: Benjamin is trying to break a combination lock. This particular type of lock has 5 digits from 0 to 9, and Benjamin just happens to know that there can be no repeated digits in the code for this type of lock. How many valid codes are there?

Answer: 30240

CoT Learning

To find the number of valid codes, we need to consider the number of choices for each digit.
For the first digit, there are 9 choices (0-9, excluding 0).
For the second digit, there are 8 choices (0-9, excluding the digit already chosen).
For the third digit, there are 7 choices (0-9, excluding the digits already chosen).
For the fourth digit, there are 6 choices (0-9, excluding the digits already chosen).
For the fifth digit, there are 5 choices (0-9, excluding the digits already chosen).
Therefore, the total number of valid codes is 9 * 8 * 7 *6 * 5 = 1260.
The answer is: 1260

PoT Learning

```python
from sympy import factorial

def valid_codes():
    """Benjamin is trying to break a combination lock. This particular type of lock has 5 digits from 0 to 9, and Benjamin just happens to know that there can be no repeated digits in the code for this type of lock. How many valid codes are there?"""
    # There are 10 choices for each of the 5 digits
    num_codes = 10**5

    return num_codes

result = valid_codes()
print(result)
```
```output
100000
```
The number of valid codes is 100000.

SAAS

```python
from sympy import factorial

def valid_codes():
    """Benjamin is trying to break a combination lock. This particular type of lock has 5 digits from 0 to 9, and Benjamin just happens to know that there can be no repeated digits in the code for this type of lock. How many valid codes are there?"""
    # There are 10 choices for each of the 5 digits, and the choices are independent,
    # so we use the product of the number of choices for each digit
    valid_codes = factorial(10) / factorial(10 - 5)

    return valid_codes

result = valid_codes()
print(result)
```
```output
30240
```
The number of valid codes is 30240.

Figure 4: Responses of different learning approaches for a given question-answer pair.

mathematical problem extend (*i.e.*, the difficulty increases), especially the step 5 or above, the difference between our **SAAS** and other strategies becomes more pronounced. This result supports our hypothesis that prioritizing the learning of mathematical reasoning ability via CoT learning is helpful for the amplification of *challenging* problem-solving ability.

3.2.4 RQ4: Case Study

To demonstrate that our **SAAS** is effective in terms of both mathematical reasoning and computational accuracy, we conduct a case study showing the responses of CoT learning, PoT learning, and **SAAS** for a given question-answer pair. Figure 4 shows the visualization results, where the colored words indicate incorrect responses and the words with no color mark indicate correct responses.

As depicted in Figure 4, CoT learning approach exhibited inaccuracies in arithmetic computations as well as deficiencies in mathematical reasoning. Conversely, PoT approach demonstrated precise calculations yet exhibited a critical deficiency in mathematical reasoning. As we expected, our **SAAS** exhibited precise computational accuracy along with enhanced mathematical reasoning capabilities (See the more detailed comments than the comments of PoT learning). Through this case study, we demonstrated the following three observations: i) only CoT learning approach leads to arithmetic calculation errors; ii) only PoT learning approach may result in a deficit of mathematical reasoning; iii) sequential learning that transitions from CoT to PoT learning help improve computational accuracy as well as mathematical reasoning.

4 Conclusion

In this paper, we demonstrated the following two important points in the sense of solving challenging mathematical problems: (1) prioritizing the learning of mathematical reasoning ability via Chain-of-Thought (CoT) learning is helpful for the amplification of problem-solving ability during Program-of-Thought (PoT) learning; (2) for effective sequential learning, it is necessary to employ *a cognitive retention strategy* that incorporates some data samples from the initial phase into the subsequent phase. In light of this, we proposed a novel sequential learning approach, named **SAAS** (Solving Ability Amplification Strategy), which progresses from CoT learning to PoT learning with cognitive retention strategy. Through extensive experiments with the reputable benchmarks, we demonstrated that **SAAS** consistently and significantly outperforms all competitor, marking a significant advancement in the field of mathematical reasoning in LLMs.

Acknowledgements

This work was supported by the 2023 KT ICT AI2XL Laboratory R&D Fund" project funded by KT.

Limitations

This study, while advancing the field of computational linguistics through the use of Large Language Models (LLMs), encounters several limitations that are important to acknowledge.

Firstly, the intricate nature of LLMs can sometimes lead to unpredictability in their outputs. This unpredictability can be particularly challenging when dealing with mathematical reasoning, where precision and accuracy are paramount, making it difficult to utilize LLMs in applications in the field of mathematics.

Furthermore, despite advancements via our study, LLMs still have limitations in their understanding and application of advanced mathematical concepts. While they can perform well on structured problems, their ability to handle abstract and complex mathematical reasoning is still an area of ongoing research and development.

Additionally, the reliance on synthetic data for training these models also presents a limitation. While synthetic datasets are useful for mitigating the scarcity of real-world data, it may not always accurately capture real-world scenarios, leading to potential gaps in the model's performance when applied to practical, real-world tasks.

Finally, ethical considerations, particularly around the potential misuse of AI, remain a concern. Ensuring that LLMs are used responsibly and do not perpetuate biases is an ongoing challenge in the field.

In summary, while our study leverages the capabilities of LLMs to enhance mathematical reasoning in computational linguistics, it is important to recognize the limitations related to unpredictability of LLMs, understanding of advanced mathematical concepts, reliance on synthetic data, and ethical considerations. These limitations highlight the need for continued research and development in the field to address these challenges effectively.

Ethics Statement

In this research, we have diligently adhered to the highest ethical standards of scientific inquiry and data management, ensuring the integrity and reliability of our findings. The design and execution of our experiments were grounded in fairness and objectivity, without favoring any particular outcome. This commitment was reflected in our meticulous planning and consistent application of methodologies across various datasets.

We also placed a strong emphasis on data privacy and security, handling all data, especially synthetic data generated for our models, in compliance with relevant data protection laws and guidelines. We confirmed that all the data used in our experiments were free of licensing issues. Our approach to data was characterized by strict anonymization protocols and its use was confined strictly to research purposes. We have strived for transparency in our research process, documenting all methodologies, data sources, and analysis techniques clearly, which underpins our commitment to the reproducibility of scientific research. This allows other researchers to verify our results and build upon our work, contributing to the collective knowledge in the field.

Recognizing the broader impacts of AI and LLMs on society, our research was conducted with a profound sense of responsibility. We were mindful of the ethical implications of AI development and aimed to create models that are effective yet ethically aligned, avoiding any form of biased, discriminatory, or harmful applications of these technologies. We believe our research makes a positive contribution to the field of computational linguistics and AI, particularly in enhancing the mathematical reasoning capabilities of Large Language Models in a manner that is ethically sound and socially responsible.

Our work underscores our commitment to conducting scientifically rigorous and ethically responsible research, maintaining the highest standards of integrity in AI and computational linguistics.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng

Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Anthropic. 2023. Model card and evaluations for claude models. URL `https://www-files.anthropic.com/production/images/Model-Card-Claude-2.pdf`.

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 arXiv:2310.10631*.

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

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. *arXiv preprint arXiv:2305.12524*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

Iddo Drori, Sunny Tran, Roman Wang, Newman Cheng, Kevin Liu, Leonard Tang, Elizabeth Ke, Nikhil Singh, Taylor L Patti, Jayson Lynch, et al. 2021. A neural network solves and generates mathematics problems by program synthesis: Calculus, differential equations, linear algebra, and more. *CoRR, abs/2112.15594*.

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

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

Robert Glaser. 1984. Education and thinking: The role of knowledge. *American psychologist*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yujiu Yang, Minlie Huang, Nan Duan, Weizhu Chen, et al. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

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

Zhanming Jie, Trung Quoc Luong, Xinbo Zhang, Xiaoran Jin, and Hang Li. 2023. Design of chain-of-thought in math problem solving. *arXiv preprint arXiv:2309.11054*.

Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, et al. 2023. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

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

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023b. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*.

Zhenwen Liang, Dian Yu, Xiaoman Pan, Wenlin Yao, Qingkai Zeng, Xiangliang Zhang, and Dong Yu. 2023. Mint: Boosting generalization in mathematical reasoning via multi-view fine-tuning. *arXiv preprint arXiv:2307.07951*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2022a. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*.

Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. 2022b. A survey of deep learning for mathematical reasoning. *arXiv preprint arXiv:2212.10535*.

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

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

Jordan Meadows and André Freitas. 2022. A survey in mathematical language processing. *arXiv preprint arXiv:2205.15231*.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2021. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022. Numglue: A suite of fundamental yet challenging mathematical reasoning tasks. *arXiv preprint arXiv:2204.05660*.

OpenAI. 2023. Chat-gpt. URL https://openai.com/blog/chatgpt.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Jing Qian, Hong Wang, Zekun Li, Shiyang Li, and Xifeng Yan. 2022. Limitations of language models in arithmetic and symbolic induction. *arXiv preprint arXiv:2208.05051*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Timo Schick, Jane Dwivedi-Yu, R Dessı, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools (2023). *arXiv preprint arXiv:2302.04761*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Avijit Thawani, Jay Pujara, Pedro A Szekely, and Filip Ilievski. 2021. Representing numbers in nlp: a survey and a vision. *arXiv preprint arXiv:2103.13136*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

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

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712*.

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

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

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

Dongxiang Zhang, Lei Wang, Luming Zhang, Bing Tian Dai, and Heng Tao Shen. 2019. The gap of semantic parsing: A survey on automatic math word problem solvers. *IEEE transactions on pattern analysis and machine intelligence*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Hattie Zhou, Azade Nova, Hugo Larochelle, Aaron Courville, Behnam Neyshabur, and Hanie Sedghi. 2022. Teaching algorithmic reasoning via in-context learning. *arXiv preprint arXiv:2211.09066*.

A Related Work and Background

The field of Large Language Models (LLMs) has witnessed substantial advancements, yet the integration of mathematical reasoning within these models remains a challenging frontier. Existing researches in LLMs primarily focus on the natural language understanding and generation (Wei et al., 2022a; Yang et al., 2023), with limited exploration in mathematical problem-solving. The complexity of mathematical problems, which requires not only numerical computation but also logical inference and the understanding of abstract concepts, still remains a notable challenge for LLMs (Zhao et al., 2023; Lu et al., 2022b; Meadows and Freitas, 2022; Qian et al., 2022; Zhou et al., 2022; Lightman et al., 2023; Drori et al., 2021; Zhang et al., 2019). To address this challenge, many researches are being conducted via the following approaches: 1) prompting approach, 2) fine-tuning approach, and 3) continued pretraining approach.

Prompting Approach Recent studies are based on the prompting methods for mathematical reasoning without additional training. Recently, the concepts of Chain of Thoughts (CoT) (Wei et al., 2022b) and Program of Thoughts (PoT) (Chen et al., 2022; Gao et al., 2023a) have emerged as promising approaches to enhance mathematical reasoning in LLMs. The CoT involves breaking down complex reasoning problems into a series of intermediate reasoning steps. This approach has shown promise in improving the accuracy and reliability of LLMs in mathematical problem-solving, by mimicking the human thought process of step-by-step reasoning. However, it is not ideal for solving complex mathematical problems (Chen et al., 2022). To address this issue, the PoT introduces a more algorithmic perspective. Specifically, it expresses the reasoning steps as code and delegate computation steps to an code interpreter. This approach allows the LLMs to effectively deal with problems that require a combination of mathematical operations and logical reasoning, by structuring the problem-solving process in a programmatic manner.

Fine-tuning Approach More recently, many works (Luo et al., 2023; Yue et al., 2023; Yu et al., 2023; Gou et al., 2023) focus on the fine-tuning LLMs for mathematical reasoning tasks. WizardMath (Luo et al., 2023) proposed Reinforcement Learning from Evol-Instruct Feedback (RLEIF), which integrates supervised fine-tuning (SFT) and proximal policy optimization (PPO) for mathematical reasoning. MAmmoTH (Yue et al., 2023) introduces a new hybrid instruction-tuning dataset called MathInstruct[3], which consists of CoT rationale and PoT rationale. MetaMath (Yu et al., 2023) proposed a new instruction-tuning dataset named MetaMathQA[4], which is augmented by question bootstrapping methods. ToRA (Gou et al., 2023) suggested a series of tool-integrated reasoning agents, which is fine-tuned on the tool-use trajectories (PoT rationale) datasets generated by prompting GPT-4.

Continued Pretraining Approach Some researches (Lewkowycz et al., 2022; Azerbayev et al., 2023) continually pretrain a base model to specialize in the mathematical reasoning. Minerva (Lewkowycz et al., 2022) is a large language model pretrained on general natural language data and further trained on the scientific and mathematical data. Llemma (Azerbayev et al., 2023) was also obtained through continued pretraining Code Llama (Roziere et al., 2023) on their own collected data named Proof-Pile-2[5].

In this paper, we focus on the fine-tuning approach by integrating the CoT and PoT learning. Motivated by Dong et al. (2023) that showed that the abilities of LLMs can be improved depending on the SFT strategy, we analyze how much performance can be improved depending on the SFT strategy from the perspective of solving challenging mathematical problems.

B Detailed Descriptions of Seed Datasets

The detailed description of each seed dataset is as follows:

i) **GSM8K** (Cobbe et al., 2021): It focuses on elementary-level math problems to evaluate abilities that handle logical reasoning and parse and interpret math questions presented in natural language;

ii) **MATH** (Hendrycks et al., 2021): It includes a wide range of math problems, ranging from elementary arithmetic to advanced topics such as

[3]https://huggingface.co/datasets/TIGER-Lab/MathInstruct

[4]https://huggingface.co/datasets/meta-math/MetaMathQA

[5]https://huggingface.co/datasets/EleutherAI/proof-pile-2

algebra, calculus, and geometry, which are challenging more than GSM8K;

iii) **MetaMathQA** (Yu et al., 2023): It is a dataset augmented through rephrasing question, forward-backward reasoning (Jiang et al., 2023), self-verification, and answer augmentation based on GSM8K and MATH;

iv) **MathInstruct** (Yue et al., 2023): It consists of a mix of 13 types of CoT and PoT mathematical rationales from various mathematical fields. Specifically, CoT type data consist of GSM8K, GSM8K-RFT (Yuan et al., 2023), AQuA-RAT (Ling et al., 2017), MATH, THeoremQA (Chen et al., 2023) Camel-Math (Li et al., 2023b) and College-Math. Otherwise, PoT type data consist of GSM8K, AQuA-RAT, TheoremQA, MathQA (Amini et al., 2019) and NumGLUE (Mishra et al., 2022);

v) **QANDA**: It consists of a diverse collection of real-world mathematical questions and detailed solutions, catering to a broad spectrum of mathematical concepts and difficulty levels.