

Language Model Reasoning (II)

Mark Lu, Yasaman Ansari, Nick Song

Sep 12, 2024

Agenda

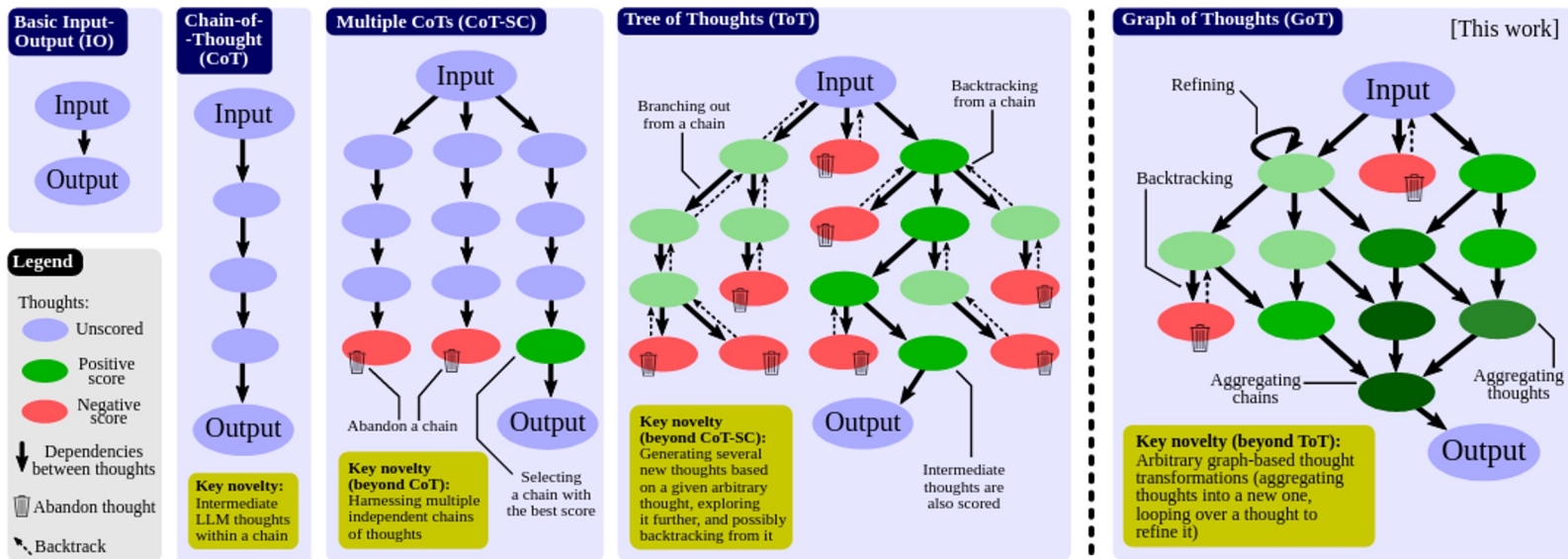
- Large Language Models Can Self-Improve
- Progressive-Hint Prompting Improves Reasoning in Large Language Models
- Large Language Models are Better Reasoners with Self-Verification
- Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models

Large Language Models Can Self-Improve

Huang et al.

Recap

- Chain-of-Thought prompting (Wei et al., 2022b)
- Self-consistency (Wang et al., 2022b)



Related Work - Self-Improvement Framework

- Traditional Self-Training
 - Assigns pseudo-labels to unlabeled data using trained classifier
 - Iterative process: Generate pseudo-labels → Retrain → Repeat
- Innovative Approach of This Work
 - Combines Chain-of-Thought (CoT) prompting and self-consistency
 - Generates rationale-augmented answers
 - Provides both answers and reasoning processes
 - Directly applied to fine-tuning large language models

Method

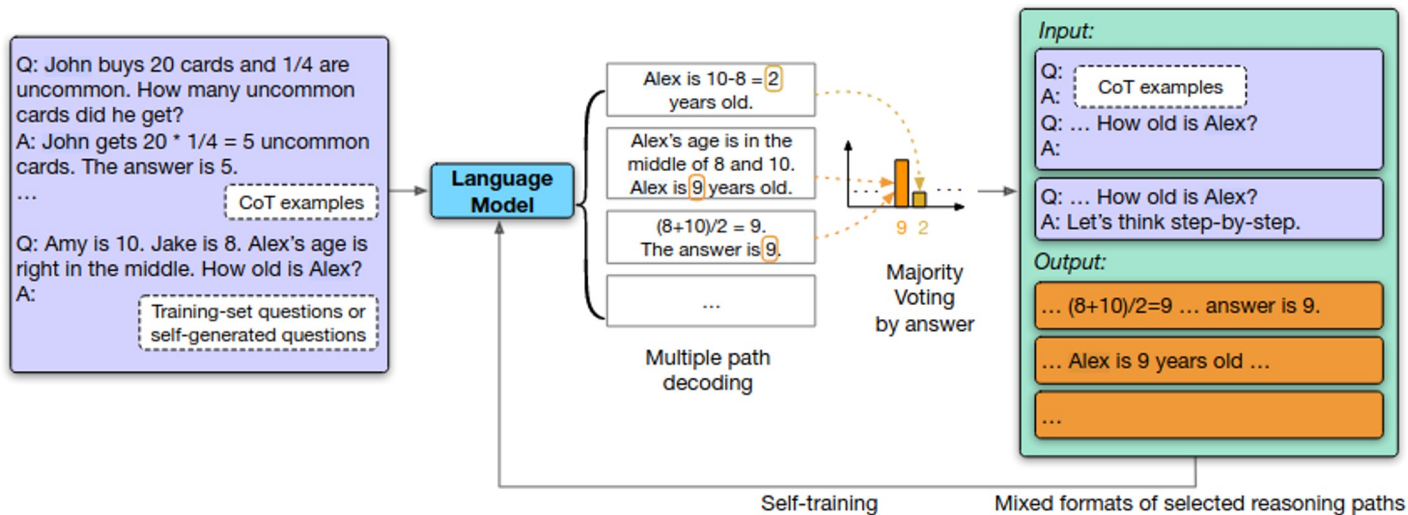


Figure 1: Overview of our method. With Chain-of-Thought (CoT) examples as demonstration (Wei et al., 2022b), the language model generates multiple CoT reasoning paths and answers (temperature $T > 0$) for each question. The most consistent answer is selected by majority voting (Wang et al., 2022b). The “high-confidence” CoT reasoning paths that lead to the majority answer are augmented by mixed formats as the final training samples to be fed back to the model for fine-tuning.

Method

Multiple Reasoning Paths

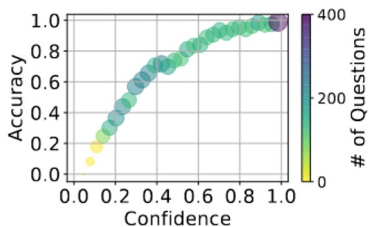


Figure 2: The relation of accuracy and confidence of the majority-voted answer after multiple path decoding on GSM8K training-set questions. Predicted confidence from self-consistency (Wang et al., 2022b) is well calibrated (Guo et al., 2017).

Question: Stefan goes to a restaurant with his family. They order an appetizer that costs \$10 and 4 entrees that are \$20 each. If they tip 20% of the total, what is the total amount of money that they spend?

Multiple Path Decoding:

Output 1: The appetizer costs \$10. The entrees cost \$20 each so in total $4 * \$20 = \80 . This means the total cost is $\$10 + \$80 = \$90$. They tip 20% of it, so the total amount they spend is $\$90 * 1.2 = \108 . The answer is 108.

Output 2: The appetizer costs \$10 and the entrees are \$20 each. There are 4 entrees so the sum is $\$20 * 4 = \80 . The waiter gets 20% of the total. 20% of \$80 is $\$80 * .2 = \16 . The answer is $\$80 + \$16 = \$96$. (*Incorrect reasoning path*)

Output 3: The appetizer costs \$10. The entrees cost $4 * \$20 = \80 . The tip is 20% of the total, so it is 20% of the \$90 they have spent. The tip is $0.2 * 90 = \$18$. The total they spent is $\$90 + \$18 = \$108$. The answer is 108.

Table 1: Examples of 3 self-generated CoT reasoning paths given a question. Output 1 and 3 are the most consistent reasoning paths based on majority voting and kept as self-training data.

Method

Training With Mixed Formats

- CoT prompting
- Standard prompting
- "Let's think step by step" (Kojima et al., 2022)
- Plain

Question: Amy is 10 years old. Jake is 8 years old. Alex's age is right in the middle. How old is Alex?

Selected Chain-of-Thought: Amy is 10 years old. Jake is 8 years old. Alex's age is in the middle of Amy and Jake, so Alex is $(8 + 10) / 2 = 9$ years old. The answer is 9.

Mixed-formats of training data:

Format 1: Input: *[CoT prompting examples]* + '\n' + *[Question]* + '\n' + 'A:'

Output: Amy is 10 years old. Jake is 8 years old. Alex's age is in the middle of Amy and Jake, so Alex is $(8 + 10) / 2 = 9$ years old. The answer is 9.

Format 2: Input: *[Standard prompting examples]* + '\n' + *[Question]* + '\n' + 'A:'

Output: The answer is 9.

Format 3: Input: *[Question]* + '\n' + 'A: Let's think step by step.'

Output: Amy is 10 years old. Jake is 8 years old. Alex's age is in the middle of Amy and Jake, so Alex is $(8 + 10) / 2 = 9$ years old. The answer is 9.

Format 4: Input: *[Question]* + '\n' + 'A:'

Output: The answer is 9.

Table 2: An example of how a reasoning path is augmented into four formats of training data with different prompts (in input) and answer styles (in output). Specifically, the *CoT prompting examples* used for each tasks are listed in Appendix A.2. The *Standard prompting examples* are the same question-answer pairs with *CoT prompting examples*, except that reasoning is removed.

Method

- Generating Questions
 - Select existing questions as input prompt
 - Let the LLM generate consecutive sequences as new questions
 - Only keep the questions that have a highly confident answer
- Generating Prompts
 - “A: Let’s think step by step.”
 - Let the LLM generate consecutive reasoning paths.

Experiment

- Datasets
 - Arithmetic reasoning:
 - GSM8K (Cobbe et al., 2021)
 - DROP (Dua et al., 2019)
 - Commonsense reasoning:
 - OpenBookQA (Mihaylov et al., 2018) dataset,
 - AI2 Reasoning Challenge (ARC) (Clark et al., 2018) sub-set (ARC-c).
 - Natural Language Inference:
 - Adversarial NLI (ANLI) (Mihaylov et al., 2018) subsets,
 - ANLI-A2 and ANLI-A3
- Models
 - PaLM-540B

Result - Main

	Prompting Method	GSM8K	DROP	ARC-c	OpenBookQA	ANLI-A2	ANLI-A3
	Previous SOTA	82.3 ^a	84.9 ^b	88.7 ^c	91.0 ^d	64.9 ^d	66.0 ^d
w/o LMSI	Standard-Prompting	17.9	60.0	87.1	84.4	55.8	55.8
	CoT-Prompting	56.5	70.6	85.2	86.4	58.9	60.6
	Self-Consistency	74.4	78.2	88.7	90.0	64.5	63.4
LMSI	Standard-Prompting	32.2	71.7	87.2	92.0	64.8	66.9
	CoT-Prompting	73.5	76.2	88.3	93.0	65.3	67.3
	Self-Consistency	82.1	83.0	89.8	94.4	66.5	67.9

Table 3: Accuracy results on six reasoning benchmarks. The previous SOTA results are from: (a) Li et al. (2022a), (b) Zhou et al. (2022b), (c) Wang et al. (2022b), (d) Wang et al. (2022a).

Result - Main

	Self-training data	AQUA	SVAMP	StrategyQA	ANLI-A1	RTE	MNLI-M/MM
w/o LMSI	-	35.8	79.0	75.3	68.8	79.1	72.0/74.0
LMSI	GSM8K + DROP + ...	39.0	82.8	77.8	79.2	80.1	81.8/82.2

Table 4: Comparison of CoT-prompting accuracy results on six Out-Of-Domain benchmarks with or without training on six In-Domain (GSM8K, DROP, ARC-c, OpenBookQA, ANLI-A2, ANLI-A3) training-set questions.

Result - Ablation study

	Results on GSM8K	
	Standard Prompting	CoT Prompting
w/o LMSI	17.9	56.5
LMSI w/o CoT formats	23.6	61.6
LMSI	32.2	73.5

Table 5: Ablation study: w/ or w/o CoT reasoning paths as training format on GSM8K dataset.

Result - Self-Improvements

	Questions used for Self-Training	Results on GSM8K	
		CoT-Prompting	Self-Consistency
w/o LMSI	-	56.5	74.4
LMSI	Generated Questions	66.2	78.1
LMSI	Training-set Questions	73.5	82.1

Table 6: Accuracy on GSM8K test set after self-training on self-generated or training set questions.

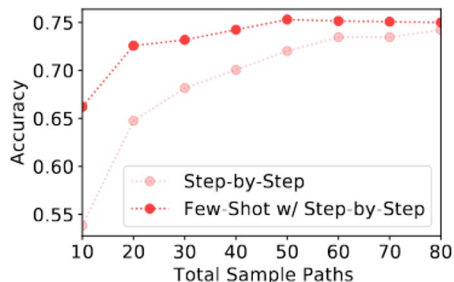


Figure 3: Accuracy results on GSM8K test set using PaLM-540B model with multi-path sampling and self-consistency (Wang et al., 2022b). “Step-by-Step” is the baseline performance of Kojima et al. (2022) plus self-consistency (Wang et al., 2022b), while our “Few-Shot w/ Step-by-Step” uses exemplars self-generated from Step-by-Step (greedy decoding) for few-shot prompting the LLM.

Result - Distillation

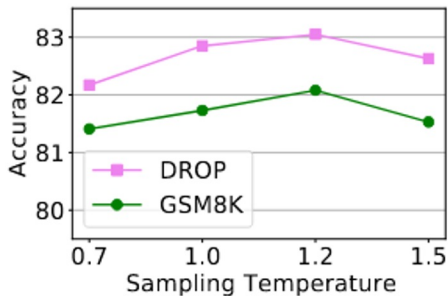
	Results on GSM8K		
	8 billion	62 billion	540 billion
w/o LMSI	5.0	29.7	56.5
Distilled from LMSI 540 billion	33.4	57.4	-

Table 7: Distillation from PaLM-540B model to small models. We see that distilled smaller models outperform models that are one-tier larger.

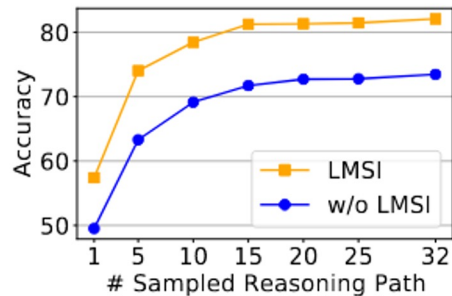
Result

Hyperparameter Study

- Temperature
- Number of Sampled Reasoning Paths



(a) Accuracy results of **LMSI** on GSM8K and DROP test set when different sampling temperatures are applied for Self-Consistency.



(b) Accuracy results with or without **LMSI** on GSM8K test set using different numbers of sampled reasoning path for Self-Consistency.

Figure 4: Hyperparameter study results.

Result

- Performance on Smaller Model UL2
 - PaLM-540B is about 27 times larger than UL2-20B in terms of parameter count.

	Prompting Method	GSM8K	DROP	ARC-c	OpenBookQA	ANLI-A2	ANLI-A3
w/o LMSI	CoT-Prompting	5.4/7.1	11.1/16.8	49.9	53.6	35.9	33.8
	Self-Consistency	6.4/9.9	16.8/26.5	54.9	54.0	37.4	36.8
LMSI	CoT-Prompting	6.1/8.6	11.4/17.1	50.9	53.8	35.4	34.4
	Self-Consistency	7.9/10.2	18.1/28.1	54.7	55.2	38.1	37.4

Table 8: Accuracy results on six reasoning benchmarks with **LMSI** on UL2. On GSM8K and DROP, we also include accuracy scores after an equation-correction postprocessing step.

Conclusions

- Improves the 540B-parameter LLM
 - 74.4%→82.1% on GSM8K
 - 78.2%→83.0% on DROP
 - 90.0%→94.4% on OpenBookQA
 - 63.4%→67.9% on ANLI-A3
- Out-of-domain (OOD) test tasks
- Future work
 - Large-scale generated data

Progressive-Hint Prompting Improves Reasoning in Large Language Models

Authors: Chuanyang Zheng, et al.

<https://arxiv.org/abs/2304.09797>

Problem Statement

- LLMs struggle with complex reasoning tasks.
- Existing methods like Chain-of-Thought (CoT) prompting help but have limitations.
- There's a need for more dynamic and iterative approaches.

Baseline Methods and Their Challenges:

- Standard LLM:
 - No specific prompting techniques used; model works in a single-pass manner.
- Chain-of-Thought (CoT):
 - Breaks problems into intermediate reasoning steps.
 - Goes through the problem only once, which isn't always enough for tough questions.
- Complex CoT:
 - Extends CoT with more complex prompts instructions: Still lacks iterative refinement.
- Self-Consistency:
 - Samples multiple paths for consistency Challenge: Computationally expensive.

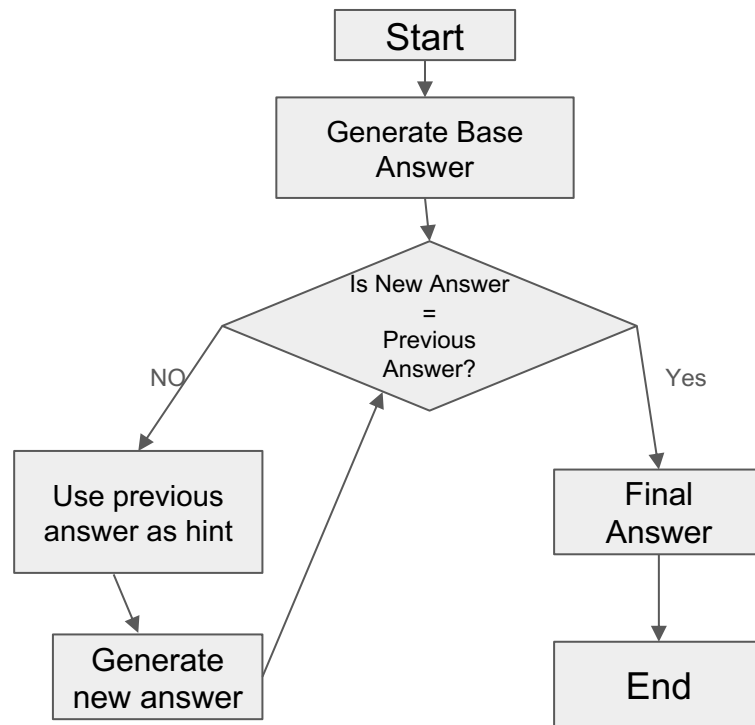
★ These shortcomings show the need for more dynamic methods

Introduction to Progressive-Hint Prompting (PHP)

- PHP guides the model to use previous answers as hints.
- Involves multiple interactions with the model.
- Helps the model refine its responses iteratively.
- mimics human problem-solving by revisiting and improving solutions through multiple interactions.

How PHP Works

- Step 1: Generate an initial answer.
- Step 2: Use this answer as a hint for the next attempt.
- Step 3: Repeat until the answer stabilizes.
- ★ PHP integrates easily with existing methods like CoT-> we can combine different techniques to help the LLM think even better.



How PHP Works – Example 1

Purple Box:
The input of
LLM.

Orange Box:
The output of
LLM.

Base Prompting

Q: There are 96 fourth-graders at Small Tree School. 43 of them are girls. On Friday, 5 fourth-grade girls and 4 fourth-grade boys were absent. How many fourth grade boys were at Small Tree School on Friday?

LLM

Base
Answer

A: There were 96 fourth-graders at Small Tree School. 43 of them were girls. On Friday, 5 girls and 4 boys were absent. So there were $43 - 5 = 38$ girls at school on Friday, and $96 - 38 = 58$ boys at school on Friday. The answer is **58**.

Progressive-Hint Prompting

Q: There are 96 fourth-graders at Small Tree School. 43 of them are girls. On Friday, 5 fourth-grade girls and 4 fourth-grade boys were absent. How many fourth grade boys were at Small Tree School on Friday? (Hint: The answer is near to **58**).

LLM

Second
Answer

A: We know the Answer Hints: **58**. With the Answer Hints: **58**, we will answer the question. There were 96 fourth-graders at Small Tree School, and 43 of them were girls. On Friday, 5 girls and 4 boys were absent. So there were $43 - 5 = 38$ girls and $96 - 43 = 53$ boys at Small Tree School on Friday. After subtracting the absent students, there were $38 - 4 = 34$ girls and $53 - 5 = 48$ boys. The answer is **48**.

Progressive-Hint Prompting

Q: There are 96 fourth-graders at Small Tree School. 43 of them are girls. On Friday, 5 fourth-grade girls and 4 fourth-grade boys were absent. How many fourth grade boys were at Small Tree School on Friday? (Hint: The answer is near to **58, 48**).

LLM

Third
Answer

A: We know the Answer Hints: **58, 48**. With the Answer Hints: **58, 48**, we will answer the question. There were 96 fourth-graders, and 43 of them were girls. On Friday, 5 girls and 4 boys were absent. So there were $43 - 5 = 38$ girls and $96 - 9 = 87$ students in total. Of those 87 students, 38 were girls, so there must have been $87 - 38 = 49$ boys. The answer is **49**.

Progressive-Hint Prompting

Q: There are 96 fourth-graders at Small Tree School. 43 of them are girls. On Friday, 5 fourth-grade girls and 4 fourth-grade boys were absent. How many fourth grade boys were at Small Tree School on Friday? (Hint: The answer is near to **58, 48, 49**).

LLM

Fourth
Answer

A: We know the Answer Hints: **58, 48, 49**. With the Answer Hints: **58, 48, 49**, we will answer the question. There were 96 fourth-graders at Small Tree School, and 43 of them were girls. On Friday, 5 girls and 4 boys were absent. So there were $43 - 5 = 38$ girls and $96 - 43 = 53$ boys originally. After the absences, there were $38 - 5 = 33$ girls and $53 - 4 = 49$ boys. The answer is **49**.

How PHP Works – Example 2

Table 1: Illustration of Progressive-Hint Prompting. **Blue Color:** The difference between Base CoT and PHP-CoT. **Red Color:** The handcrafted Hint in the designed prompt.

Base Prompting (e.g. CoT)

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?

A: There are 15 trees originally. Then there were 21 trees after the Grove workers planted some more. So there must have been $21 - 15 = 6$ trees that were planted. The answer is 6.

Progressive-Hint Prompting 1: Hint is the correct answer

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today? (Hint: The answer is near to 6).

A: We know the Answer Hints: 6. With the Answer Hints: 6, we will answer the question. There are 15 trees originally. Then there were 21 trees after the Grove workers planted some more. So there must have been $21 - 15 = 6$ trees that were planted. The answer is 6.

Progressive-Hint Prompting 2: Hint is the incorrect answer

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today? (Hint: The answer is near to 10, 8).

A: We know the Answer Hints: 10, 8. With the Answer Hints: 10, 8, we will answer the question. There are 15 trees originally. Then there were 21 trees after the Grove workers planted some more. So there must have been $21 - 15 = 6$ trees that were planted. The answer is 6.

Main Results Across Datasets

Table 2: PHP, when applied to different LLMs and prompting methods, can help to improve the performance. Meanwhile, PHP works better when the model and prompt are more powerful. The results are with greedy decoding.

	Prompt	PHP	Dataset					Average	
			AddSub	MultiArith	SingleEQ	SVAMP	GSM8K		AQuA
GPT-3.5 text-davinci-002	Standard [8]	✗	79.4	34.0	80.7	64.8	15.1	25.5	49.91
		✓	80.5 (+1.1)	31.8 (-2.2)	79.9 (-0.8)	64.2 (-0.6)	14.7 (-0.4)	25.5 (0.0)	49.43 (-0.48)
	CoT [8]	✗	85.8	89.1	89.7	72.9	49.5	44.4	71.89
		✓	86.8 (+1.0)	89.0 (-0.1)	90.1 (+0.4)	72.3 (-0.6)	51.1 (+1.6)	45.6 (+1.2)	72.48 (+0.59)
	Complex CoT [10]	✗	82.5	89.8	87.7	70.4	57.6	37.4	70.89
		✓	83.7 (+1.2)	90.1 (+0.3)	89.9 (+2.2)	74.6 (+4.2)	61.2 (+3.6)	37.0 (-0.4)	72.75 (+1.86)
GPT-3.5 text-davinci-003	Standard [8]	✗	89.1	36.3	83.8	68.7	15.9	28.3	53.68
		✓	89.1 (0.0)	36.0 (-0.3)	83.6 (-0.2)	68.7 (0.0)	16.0 (+0.1)	28.3 (0.0)	53.61 (-0.07)
	CoT [8]	✗	90.6	93.6	92.7	81.0	56.1	44.0	76.33
		✓	91.1 (+0.5)	94.0 (+0.4)	93.5 (+0.8)	81.3 (+0.3)	57.5 (+1.4)	44.4 (+0.4)	76.96 (+0.63)
	Complex CoT [10]	✗	86.3	94.8	91.5	77.4	67.0	48.8	77.63
		✓	88.1 (+1.8)	95.0 (+0.2)	94.0 (+2.5)	80.0 (+2.6)	71.6 (+4.6)	50.0 (+1.2)	79.78 (+2.15)

Performance with Different Base Answers

Table 3: Performance with different Base Answers. Initially, the base prompt provides base answers to the model and PHP generates the subsequent answers. The results are from text-davinci-003 with greedy decoding.

PHP	Base Prompt	Dataset						Average
		AddSub	MultiArith	SingleEQ	SVAMP	GSM8K	AQuA	
PHP-Standard	Standard [8]	89.1	36.0	83.6	68.7	16.0	28.3	53.61
	CoT [8]	92.4	80.5	92.1	78.5	50.2	42.5	72.70
	Complex CoT [10]	90.6	80.6	92.9	77.2	60.3	45.6	74.53
PHP-CoT	Standard [8]	90.8	92.5	90.7	80.2	52.3	40.9	74.56
	CoT [8]	91.1	94.0	93.5	81.3	57.5	44.4	76.96
	Complex CoT [10]	90.6	96.8	93.7	81.2	62.6	50.0	79.14
PHP-Complex CoT	Standard [8]	88.3	80.1	93.3	80.4	65.5	35.4	73.83
	CoT [8]	88.8	95.6	94.8	81.4	70.6	45.6	79.46
	Complex CoT [10]	88.1	95.0	94.0	80.0	71.6	50.0	79.78

Ablation Study Results

Table 4: Ablation Study. CoT-Merge: for the CoT base prompt and the PHP-CoT prompt, we employ the prompt that contains both base prompt and the PHP. **P1**: We know the Answer Hints A_1, \dots, A_p . **P2**: With the Answer Hints A_1, \dots, A_p , we will answer the question. According to the experiment results, we see that both the proposed P1 and P2 are necessary. Meanwhile, non-merge based method is better than merge based method when prompts are more powerful. The results are from text-davinci-003 with greedy decoding.

Method	P1	P2	Dataset						Average
			AddSub	MultiArith	SingleEQ	SVAMP	GSM8K	AQuA	
CoT-Merge	✓	✓	91.3	94.6	93.1	79.5	58.6	50.0	77.85
CoT [8]	✗	✗	91.1	93.5	93.3	80.0	58.1	44.8	76.80
	✓	✗	90.8	93.1	92.9	80.7	58.8	43.7	76.66
	✗	✓	91.3	93.8	93.5	80.5	58.2	46.4	77.28
	✓	✓	91.1	94.0	93.5	81.3	57.5	44.4	76.96
Complex CoT-Merge	✓	✓	88.8	94.3	94.6	78.1	70.2	46.8	78.80
Complex CoT [10]	✗	✗	87.8	93.3	93.7	78.0	68.3	50.3	78.56
	✓	✗	87.8	95.1	94.2	78.5	70.5	48.4	79.08
	✗	✓	88.3	94.3	94.6	79.1	69.3	46.8	78.73
	✓	✓	88.1	95.0	94.0	80.0	71.6	50.0	79.78

Hint Design Analysis

Table 5: Analysis of Hint Design (Shown in Figure 1). Correct: The hints of designed prompt are the same as the correct answers. Incorrect: The hints of the designed prompt are the incorrect answers. **Green**: The performance is better than without progressive-hint. **Red**: The performance is worse than without progressive-hint. The results are from text-davinci-003 with greedy decoding.

Method	Hint		Dataset						Average
	Correct	Incorrect	AddSub	MultiArith	SingleEQ	SVAMP	GSM8K	AQuA	
CoT [8]	✗	✗	90.6	93.6	92.7	81.0	56.1	44.0	76.33
	✓	✗	91.6	94.3	93.3	81.9	57.0	43.7	76.96
	✗	✓	91.1	93.5	93.1	79.7	57.9	45.2	76.74
	✓	✓	91.1	94.0	93.5	81.3	57.5	44.4	76.96
Complex CoT [10]	✗	✗	86.3	94.8	91.5	77.4	67.0	48.8	77.63
	✓	✗	88.3	94.0	93.8	77.8	68.6	46.4	78.14
	✗	✓	88.1	94.6	94.0	79.2	70.2	48.4	79.08
	✓	✓	88.1	95.0	94.0	80.0	71.6	50.0	79.78

Results with Self-Consistency

Table 6: The results after adding Self-Consistency (SC). **Number**: The interaction number between agent and LLM. The **best results of adding PHP** are highlighted with red color, and the **best results without PHP** are highlighted with green color. We find that PHP further improves performance, even adding self-consistency. Meanwhile, PHP may reduce the cost of self-consistency.

Prompt	SC	PHP	Dataset						Average	
			AddSub	MultiArith	SingleEQ	SVAMP	GSM8K	AQuA		
CoT [8]	5	✗	90.6	95.3	94.4	81.6	63.3	49.2	79.06	
	5	✓	90.8	96.6	94.8	83.5	66.3	49.6	80.26	
	5	Number	2.0075	2.0433	2.0098	2.1090	2.5458	2.0157	2.1218	
	10	✗	90.6	96.5	93.8	83.0	65.5	49.2	79.76	
	10	✓	90.8	97.1	93.8	83.5	67.5	50.0	80.45	
	10	Number	2.0075	2.0283	2.0059	2.0510	2.2145	2.0118	2.0531	
	20	✗	91.1	96.5	94.2	83.3	68.0	55.1	81.36	
	20	✓	91.6	96.5	94.4	83.7	68.6	55.1	81.64	
	20	Number	2.0050	2.0366	2.0098	2.0250	2.1144	2.0078	2.0330	
	40	✗	91.6	96.5	94.8	82.9	67.3	53.1	81.03	
	40	✓	91.6	96.6	95.0	83.7	68.4	53.1	81.39	
	40	Number	2.0050	2.0300	2.0050	2.0320	2.0530	2.0000	2.0208	
	Complex CoT [10]	5	✗	88.1	97.0	93.1	80.4	73.5	51.5	80.60
		5	✓	89.6	97.3	95.2	82.5	76.9	51.9	82.23
5		Number	2.0378	2.0166	2.0334	2.2370	2.5390	2.0118	2.1459	
10		✗	88.6	98.3	93.3	82.4	76.4	54.3	82.21	
10		✓	89.1	98.5	95.2	83.4	78.2	54.7	83.18	
10		Number	2.0177	2.0016	2.0295	2.059	2.1531	2.0078	2.0447	
20		✗	88.6	98.0	93.8	82.5	77.7	56.2	82.80	
20		✓	89.8	98.0	95.8	83.6	78.6	56.2	83.66	
20		Number	2.0253	2.0000	2.0196	2.0330	2.0401	2.0000	2.0196	
40		✗	88.3	98.5	94.8	83.9	78.1	58.6	83.70	
40		✓	88.6	98.5	95.8	84.7	79.0	58.6	84.20	
40		Number	2.0101	2.0000	2.0137	2.0210	2.0348	2.0039	2.0137	

Performance with Advanced Models

Table 7: Performance of Complex CoT with GPT-3.5-Turbo and GPT-4, employing greedy decoding. Number: The average interaction number with LLM.

	PHP	Dataset						Average
		AddSub	MultiArith	SingleEQ	SVAMP	GSM8K	AQuA	
Previous SOTA	✗	94.9 [27]	100 [25]	95.5 [29]	89.1 [30]	92.0 [17]	76.4 [31]	91.31
GPT-3.5 Turbo	✗	85.5	97.5	92.5	81.0	82.8	57.4	82.78
	✓	85.3	98.0	92.9	83.1	85.1	60.6	84.16
	Number	(-0.2)	(+0.5)	(+0.4)	(+2.1)	(+2.3)	(+3.2)	(+1.38)
	Number	2.1037	2.0133	2.0610	2.3570	2.3426	2.3228	2.2000
GPT-4	✗	89.3	97.8	93.1	90.5	94.9	77.5	90.51
	✓	89.6	98.1	93.1	91.9	95.5	79.9	91.34
	Number	(+0.3)	(+0.3)	(0.0)	(+1.4)	(+0.6)	(+2.4)	(+0.83)
	Number	2.0126	2.0033	2.0019	2.0700	2.0507	2.2913	2.0716

Table 8: Performance of Complex CoT with GPT-3.5-Turbo and GPT-4 on MATH dataset, employing greedy decoding. Number: The average interaction number with LLM. **Overall**: The results overall MATH subtopics [14].

	PHP	MATH Dataset							Overall
		InterAlgebra	Precalculus	Geometry	NumTheory	Probability	PreAlgebra	Algebra	
Previous SOTA[7]	✗	-	-	-	-	-	-	-	50.30
GPT-4 CoT[17]	✗	-	-	-	-	-	-	-	42.50
GPT-3.5-Turbo Complex CoT (Ours)	✗	14.6	16.8	22.3	33.4	29.7	53.8	49.1	34.12
	✓	17.1	16.1	25.4	35.1	33.7	57.7	51.1	36.50
	Number	(+2.5)	(-0.7)	(+3.1)	(+1.7)	(+4.0)	(+3.9)	(+2.0)	(+2.38)
	Number	4.2746	3.9625	4.3361	3.8166	3.7594	3.1526	3.0716	3.6673
GPT-4 Complex CoT (Ours)	✗	23.4	26.7	36.5	49.6	53.1	71.6	70.8	50.36
	✓	26.3	29.8	41.9	55.7	56.3	73.8	74.3	53.90
	Number	(+2.9)	(+3.1)	(+5.4)	(+6.1)	(+3.2)	(+2.2)	(+3.5)	(+3.54)
	Number	3.2414	3.2435	3.2233	3.1740	2.8122	2.3226	2.4726	2.8494

Key Findings- Summary of Performance Improvements

- consistently boosts LLM performance
 - Significant improvements across multiple benchmarks
 - More effective with advanced models and prompts
- State-of-the-Art Results
 - GSM8K: 95.5% (+3.5%)
 - SVAMP: 91.9% (+2.8%)
 - AQuA: 79.9% (+3.5%)
 - MATH: 53.9% (+3.6%)

Conclusion and Future Directions

PHP enhances reasoning in large language models (LLMs).

- Combines well with existing methods like Chain-of-Thought (CoT) and self-consistency.
- Significantly improves performance across complex reasoning benchmarks.

Opens new opportunities for AI applications.

- Potential uses in fields like education, research, and complex problem-solving.

Future work:

- Automating hint generation to improve efficiency.
- Exploring new types of hints beyond simple previous answers.

Large Language Models are Better Reasoners with Self- Verification

By Yixuan Weng, et al.

<https://arxiv.org/abs/2212.09561>

Background

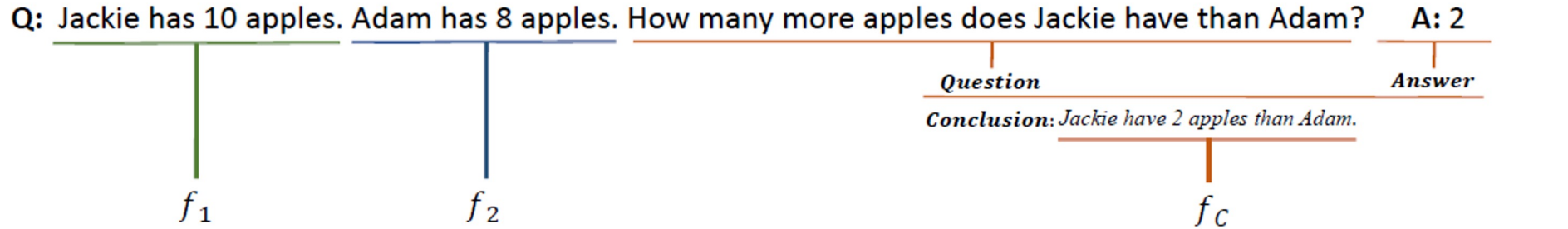
- Chain of thought (CoT) prompting for complex problems
- Multi-step prompting and multi-token prediction
- Sensitive to individual mistakes or error accumulation

Challenges

- Detecting and mitigating errors
- Previous method: Training a verifier (Shen et al., 2021; Li et al., 2022)
 - Human annotation and fine-tuned models
 - Limited use in other tasks and domains
 - Non transparent reasoning process
- A verifier that can avoid human annotations and additional training
- Self verification
 - Forward reasoning
 - Backward verification

Forward Reasoning

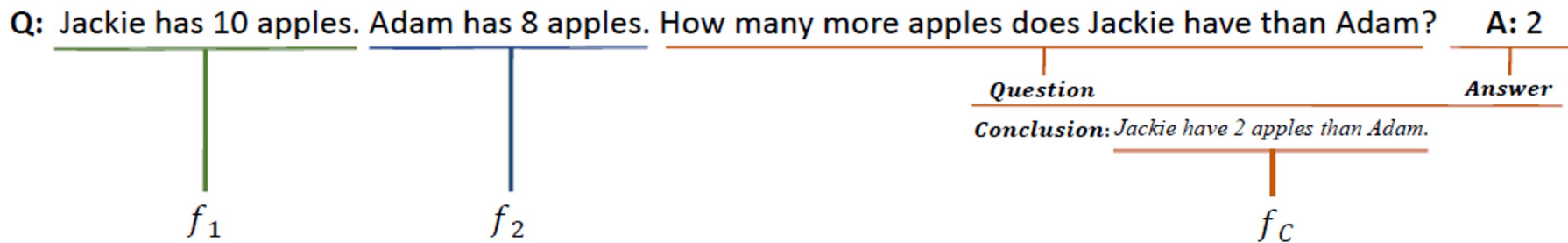
- Question + Candidate answers = Candidate conclusion



$$(f_1 \wedge f_2) \vdash f_c \quad \Rightarrow \quad \left\{ \begin{array}{l} (f_2 \wedge f_c) \vdash f_1 \quad [\text{Adam has 8 apples. Jackie have 2 apples than Adam.}] \rightarrow [\text{Jackie has 10 apples}] \\ (f_1 \wedge f_c) \vdash f_2 \quad [\text{Jackie has 10 apples. Jackie have 2 apples than Adam.}] \rightarrow [\text{Adam has 8 apples.}] \end{array} \right.$$

Backward Verification

- Mask the original condition in the question
- Predict what the original condition would have been based on the candidate conclusions
- Rank the candidate conclusions based how well do the predicted conditions align with the original conditions



$$(f_1 \wedge f_2) \vdash f_c \quad \Rightarrow \quad \left\{ \begin{array}{l} (f_2 \wedge f_c) \vdash f_1 \quad [\text{Adam has 8 apples. Jackie have 2 apples than Adam.}] \rightarrow [\text{Jackie has 10 apples}] \\ (f_1 \wedge f_c) \vdash f_2 \quad [\text{Jackie has 10 apples. Jackie have 2 apples than Adam.}] \rightarrow [\text{Adam has 8 apples.}] \end{array} \right.$$

Condition Masking

- True-False Item Verification - Reasoning QA task
- Condition Mask Verification - Condition filtering
 - “Dana worked 9 hours on Friday, 10 hours on Saturday, and 3 hours on Sunday. She earns \$13 per hour. How much money did Dana earn in weekend?”

Condition Masking

Q: "Shawn has 'X' toys. For Christmas, he got two toys each from his mom and dad. He has 9 toys now." **What is the answer of 'X'?**

A: Shawn started with X toys. If he got 2 toys each from his mom and dad, then that is 4 more toys. $X + 4 = 9$, $X = 9 - 4$, $X = 5$. **The answer is 5.**

(A) Conditional Masked Verification

Q: 'Shawn has five toys. For Christmas, he got two toys each from his mom and dad. He has 9 toys now.' **Do it is correct (True or False)?**

A: If Shawn now has 9 toys and his parents given him two each, then he originally had $9 - 2 - 2 = 5$, which is consistent with the theme. **The answer is True.**

(B) True-False Item Verification

Verification Score Calculation

$$\text{Score}_y = \begin{cases} \sum_{p=1}^P (\sum_{r=1}^R 1_{(\mathcal{L}\mathcal{L}\mathcal{M}_p(\mathcal{X}-f_r+f_y)=f_r)}) & \text{TFV} \\ \sum_{p=1}^P (1_{(\mathcal{L}\mathcal{L}\mathcal{M}_p(\mathcal{X}+f_y)}) & \text{CMV} \end{cases}$$

$$\text{Output} = \underset{k \in [0, K]}{\text{argmax}} (\text{Score}_k)$$

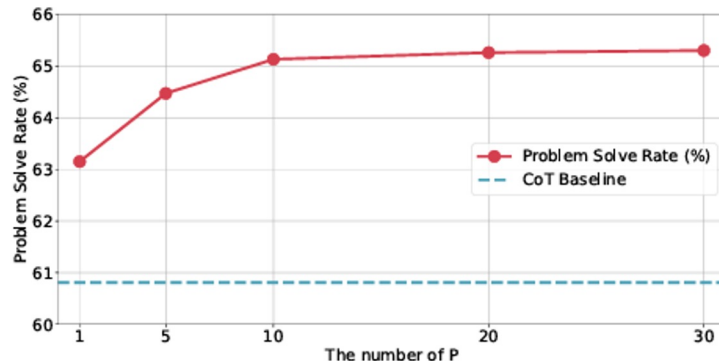


Figure 7: The computational resource of the proposed method on GSM8K.

Step1: Forward Reasoning

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?

A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been $21 - 15 = 6$. The answer is 6.
..... (CoT Prompt)

Q: Jackie has 10 apples. Adam has 8 apples. How many more apples does Jackie have than Adam?

A₁: Since Jackie has 10 apples and Adam has 8 apples, in total, Jackie has $10+8=18$ more apples than Adam, so the answer is 18.

A₂: Jackie has 10 apples, so Jackie has $10-8=2$ more apples than Adam, and the answer is 2.

A_k

Step2: Backward Verification

1) Rewritten Candidate Conclusion

Please change the questions and answers into complete declarative sentences [Q] The answer is [A]



f_c^1 Jackie has 18 apples more than Adam f_c^2 Jackie has 2 apples more than Adam f_c^k

2) Rewritten Condition

f_1 Jackie has 10 apples. \Rightarrow Regular(f_1 , $-\{d+\}, \{d^*\}$) \Rightarrow f_1 "Jackie has 'X' apples." Φ What is the answer of 'X' 10

Arithmetic Tasks: Condition Mask Verification

f_1 Jackie has 10 apples. \Rightarrow f_1 "Jackie has 10 apples." Φ Do it is correct (True or False)?

General Tasks: True-False Item Verification

3) Verification

...
Q₁: "Jackie has 'X' apples. Adam has 8 apples. Jackie has 18 apples more than Adam" What is the answer of 'X'.



A₁: Jackie has X apples, and Jackie's apples are 18 more than Adam's, so Jackie's apples should be $8+18=26$, the answer is 26.

A₁ Scores: 1

×
×
✓
×

...
Q₂: "Jackie has 'X' apples. Adam has 8 apples. Jackie has 2 apples more than Adam" What is the answer of 'X'.



A₂: Jackie has X apples, which is 2 more than Adam's 8 apples, so $8+2=10$, the answer is 10.

A₂ Scores: 4

✓
✓
✓
✓

Sorted by scores
Answer: 2

We take Condition Mask Verification as an example. For True-False Item Verification, we only need to count the number of True's to calculate the scores.

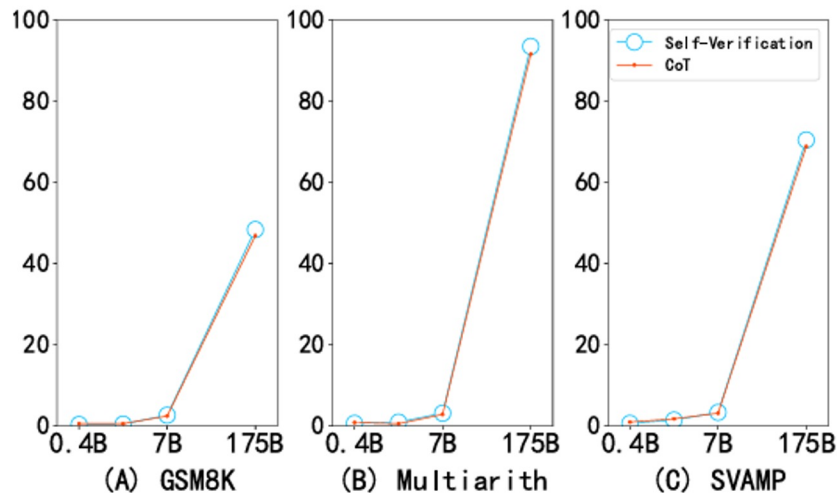
Results

Method		Arithmetic Tasks						General Tasks	
		GSM8K	SingleEq	AddSub	MultiArith	AQUA-RAT	SVAMP	CSQA	DU
Previous SOTA (Fine-tune)		35 ^a /57 ^b	32.5 ^c	94.9 ^d	60.5 ^e	37.9 ^f	57.4 ^g	91.2 ^h	-
9-12 year olds		60 ⁱ	-	-	-	-	-	-	-
GPT-3 Standard		19.7	86.8	90.9	44.0	29.5	69.9	82.3	49.0
GPT-3 (175B)	CoT	13.84	60.20	58.55	45.85	18.90	38.42	46.75	38.72
code-davinci-001	CoT+Self-Verification	13.92 _(+0.08)	60.61 _(+0.41)	59.07 _(+0.52)	46.19 _(+0.34)	27.04 _(+8.14)	38.96 _(+0.54)	47.68 _(+0.93)	39.03 _(+0.31)
Instruct-GPT (175B)	CoT	60.81	91.01	82.78	96.13	45.30	75.87	77.42	65.43
code-davinci-002	CoT+Self-Verification	65.14 _(+4.33)	93.40 _(+2.39)	86.33 _(+3.55)	99.15 _(+3.02)	47.95 _(+2.65)	76.99 _(+1.12)	77.83 _(+0.41)	66.57 _(+1.14)
Self-Consistency Decoding (Wang et al., 2023c) For Forward Reasoning									
GPT-3 (175B)	SC	23.40	70.25	68.65	79.82	25.60	54.58	54.92	49.26
code-davinci-001	SC+ Self-Verification	23.59 _(+0.19)	70.50 _(+0.25)	68.71 _(+0.06)	80.01 _(+0.19)	28.98 _(+3.38)	54.68 _(+0.1)	55.09 _(+0.17)	49.72 _(+0.46)
Instruct-GPT (175B)	SC	78.00	96.78	91.64	100.0	52.01	86.77	81.43	71.58
code-davinci-002	SC+Self-Verification	78.32 _(+0.32)	96.85 _(+0.07)	92.03 _(+0.39)	100.0 _(+0.0)	52.25 _(+0.24)	86.94 _(+0.17)	81.53 _(+0.1)	71.89 _(+0.31)
PAL (Gao et al., 2023) For Forward Reasoning									
GPT-3 (175B)	PAL	31.82	63.98	63.15	61.52	30.56	42.69	-	-
code-davinci-001	PAL+Self-Verification	32.87 _(+1.05)	65.45 _(+1.47)	64.15 _(+1.0)	61.76 _(+0.24)	30.90 _(+0.34)	42.78 _(+0.09)	-	-
Instruct-GPT (175B)	PAL	72.02	96.08	92.64	99.15	59.75	79.45	-	-
code-davinci-002	PAL+Self-Verification	72.89 _(+0.87)	96.52 _(+0.44)	93.78 _(+1.14)	99.87 _(+0.72)	60.21 _(+0.46)	80.24 _(+0.79)	-	-

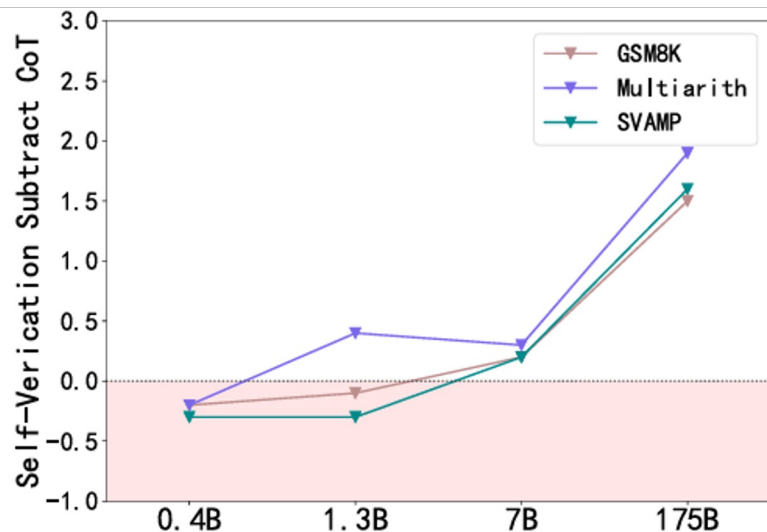
Results

- **GSM8K:**
 - high quality linguistically diverse grade school math word problems
- **SingleEq:**
 - one unknown arithmetic word problems for up-to-4 grade level students
- **AddSub**
 - addition and subtraction arithmetic word problems
- **MultiArith**
- **AQUA**
 - algebraic word problems with natural language rationales
- **SVAMP**
 - single-equation grade-school algebra word problems with multiple math operations over non negative rational numbers and one variable

Results



(a) Problem solve rate (%) in different size models. The text-ada-001 (0.4B), text-babbage-001 (1.3B), text-curie-001 (7B) and text-davinci-002 (175B) models are used respectively.



(b) Subtract the problem solve rate (%) of CoT from the problem solve rate (%) of self-verification in different size models. The pink area means that the use of self-verification has a negative impact.

Figure 3: The self-verification ability of models with different sizes.

Results

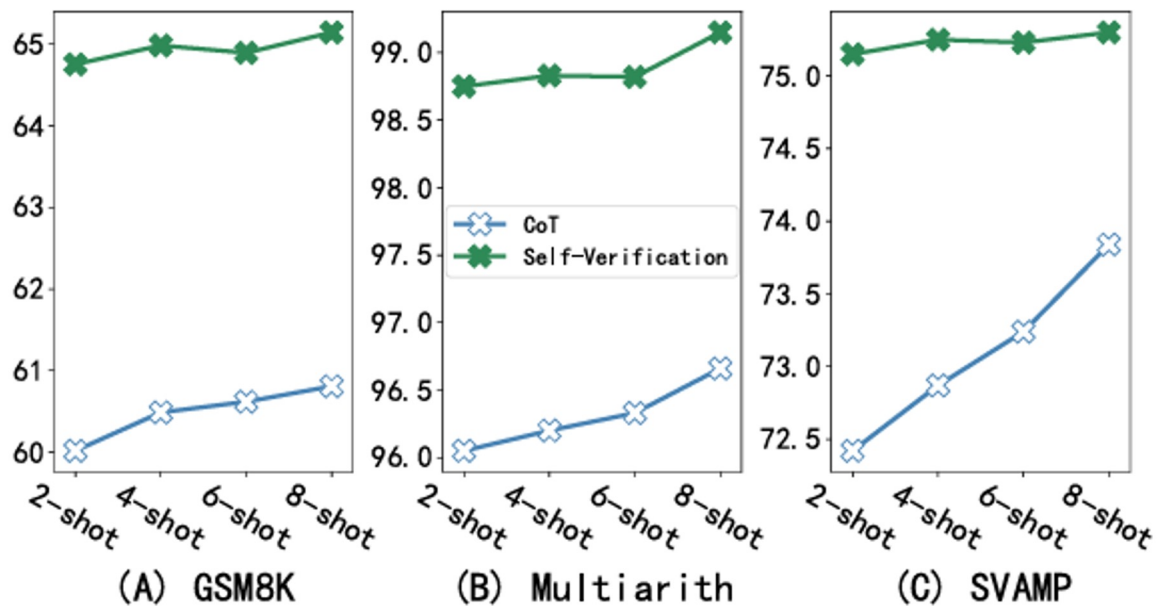
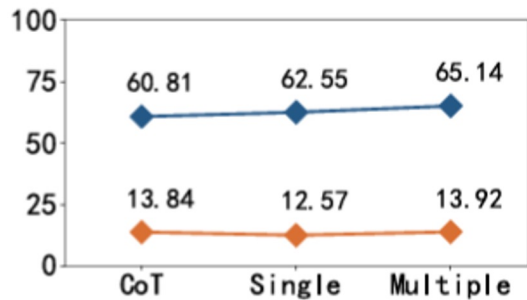
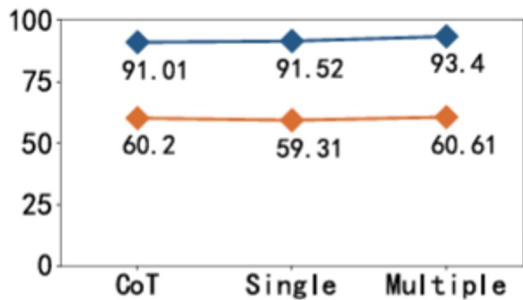


Figure 4: Problem solve rate (%) comparison of 2-shot to 8-shot prompts.

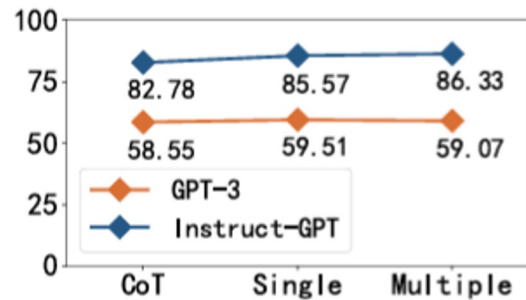
Result



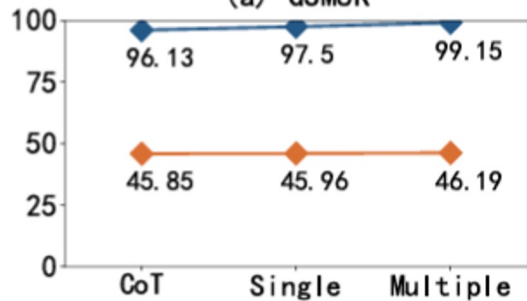
(a) GSM8K



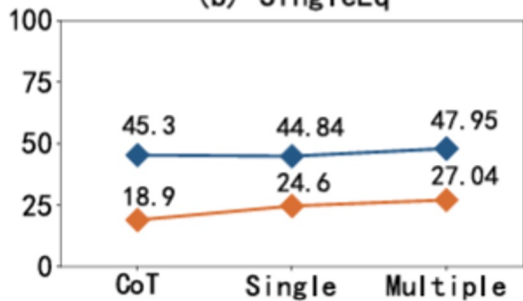
(b) SingleEq



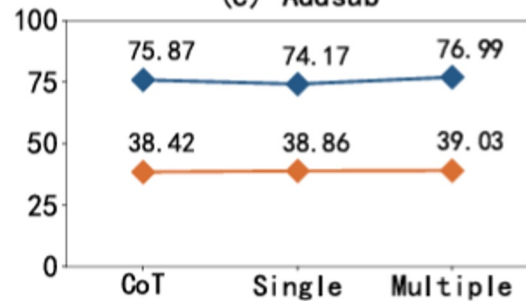
(c) Addsub



(d) MultiArith



(e) AQUA-RAT



(f) SVAMP

Results

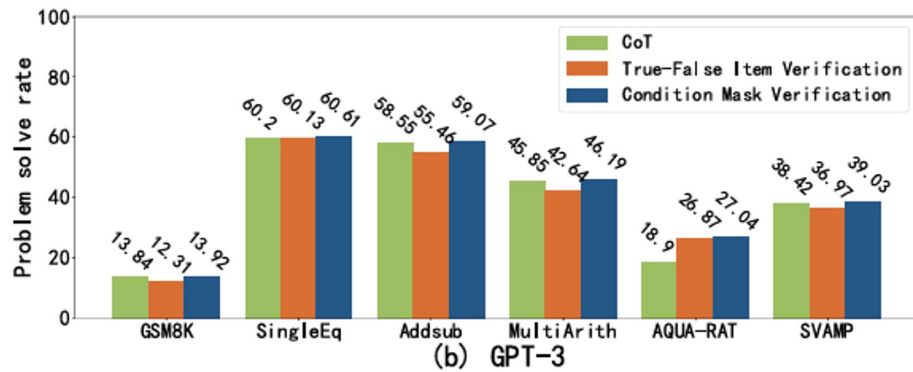
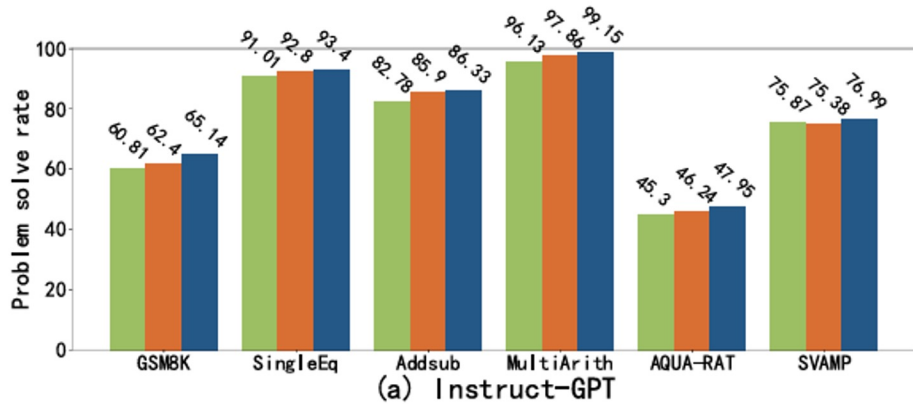


Figure 6: Comparison of problem solve rate (%) for the “CMV” and the “TFV” in arithmetic tasks.

Results

Dataset	Num.	Avg. Ans.	Ran. Acc.	CoT Acc.	Veri. Acc.
GSM8K	601/1306	2.80	35.7%	53.4%	58.9%
Addsub	77/377	2.13	46.9%	56.7%	74.0%
AQUA	123/219	2.71	36.9%	50.3%	51.2%
Multiarith	73/593	2.19	45.6%	71.2%	89.0%
SingleEq	51/501	2.31	43.3%	51.0%	74.5%
SVAMP	274/954	2.54	39.4%	51.5%	58.4%

Summary & Future Work

- LLMs have strong ability to self verify
- Enables enhancement of LLMs' reasoning abilities through a simple process of self-verification
- Limitations
 - Bias
 - Depend on LLMs' reasoning abilities. What about small scale models?
 - Rely on conclusion not reasoning process
 - Not suitable for evaluating the LLM's inference procedure
 - Increase computational cost

Plan-and-Solve Prompting: Improving Zero-Shot Chain-of- Thought Reasoning by Large Language Models

Lei Wang, et al.

<https://arxiv.org/abs/2305.04091>

Background

- Complex problems
 - Few-shot chain-of-thought (CoT) prompting(Wei et al.)
 - Zero-shot CoT(Kojima et al.)
 - Similar performance
- Zero-shot CoT
 - “Let’s think step by step”

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are $16 / 2 = 8$ golf balls. Half of the golf balls are blue. So there are $8 / 2 = 4$ blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

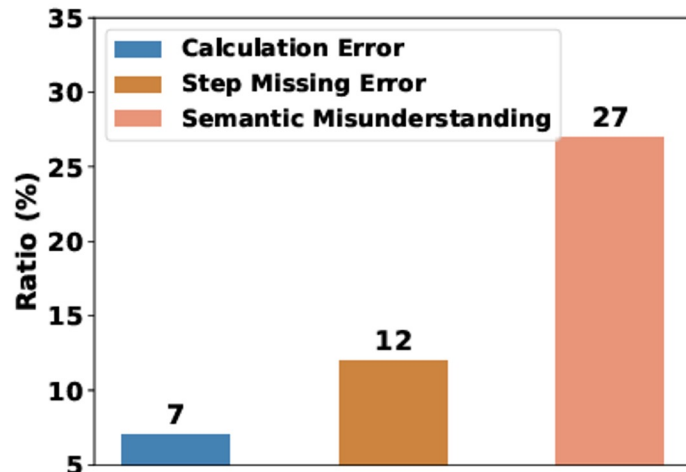
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: **Let's think step by step.**

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

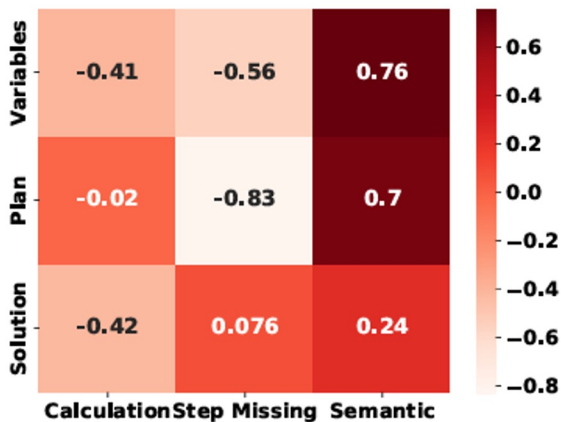
Challenges

- Zero-shot-CoT
 - Calculation errors
 - Missing Step errors
 - Semantic misunderstanding
- Plan-and-Solve (PS) Prompting
 - Step 1: Prompting for Reasoning Generation
 - Step 2: Prompting for Answer Extraction



Prompting for Reasoning Generation

- PS prompting
 - “Let’s first understand the problem and devise a plan to solve the problem. Then, let’s carry out the plan and solve the problem step by step”
- PS+ prompting
 - “extract relevant variables and their corresponding numerals”(calculation errors)
 - “calculate intermediate results (pay attention to calculation and commonsense)”(missing steps)



PS Prompting

Q: In a dance class of 20 students, 20% enrolled in contemporary dance, 25% of the remaining enrolled in jazz dance, and the rest enrolled in hip-hop dance. What percentage of the entire students enrolled in hip-hop dance?

A: *Let's think step by step.*

LLMs

First, 20% of the class enrolled in contemporary dance, so that means that 80% of the class is left.

Next, 25% of the remaining 80% enrolled in jazz dance, so that means that 25% of the original class enrolled in jazz dance. That leaves 55% of the original class.

Finally, the rest of the class (55%) enrolled in hip-hop dance. So, 55% of the entire class enrolled in hip-hop dance.



(a)

Q: In a dance class of 20 students, 20% enrolled in contemporary dance, 25% of the remaining enrolled in jazz dance, and the rest enrolled in hip-hop dance. What percentage of the entire students enrolled in hip-hop dance?

A: *Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step.*

LLMs

Plan:

Step 1: Calculate the total number of students who enrolled in contemporary and jazz dance.

Step 2: Calculate the total number of students who enrolled in hip-hop dance.

Step 3: Calculate the percentage of students who enrolled in hip-hop dance.

Solution:

Step 1: 20% of 20 students is 4 students. 25% of the remaining 16 students is 4 students. So, a total of 8 students enrolled in contemporary and jazz dance.

Step 2: The remaining 12 students enrolled in hip-hop dance.

Step 3: The percentage of students who enrolled in hip-hop dance is $12/20 = 60\%$.



(b)

Q: In a dance class of 20 students, 20% enrolled in contemporary dance, 25% of the remaining enrolled in jazz dance, and the rest enrolled in hip-hop dance. What percentage of the entire students enrolled in hip-hop dance?

A: *Let's first understand the problem and devise a plan to solve the problem.*

Then, let's carry out the plan and solve the problem step by step.

Plan:

Step 1: Calculate the total number of students who enrolled in contemporary and jazz dance.

Step 2: Calculate the total number of students who enrolled in hip-hop dance.

Step 3: Calculate the percentage of students who enrolled in hip-hop dance.

Solution:

Step 1: 20% of 20 students is 4 students. 25% of the remaining 16 students is 4 students. So, a total of 8 students enrolled in contemporary and jazz dance.

Step 2: The remaining 12 students enrolled in hip-hop dance.

Step 3: The percentage of students who enrolled in hip-hop dance is $12/20 = 60\%$.

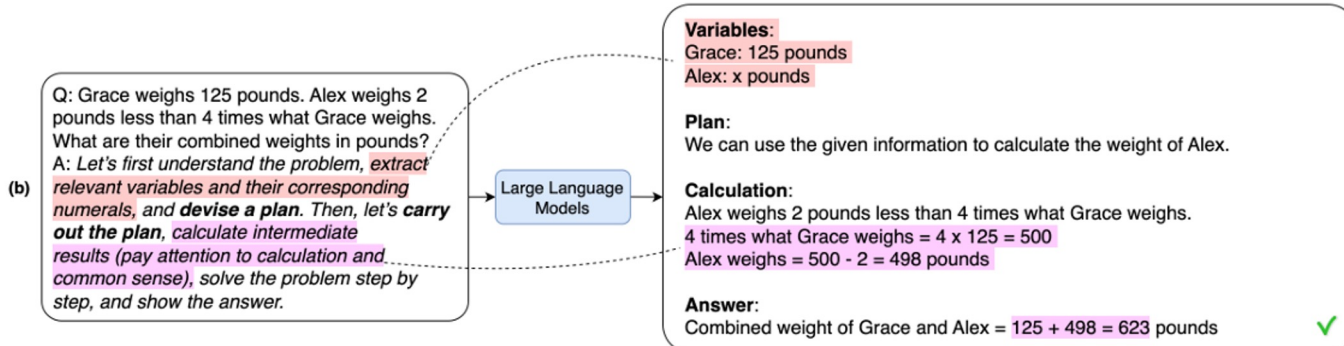
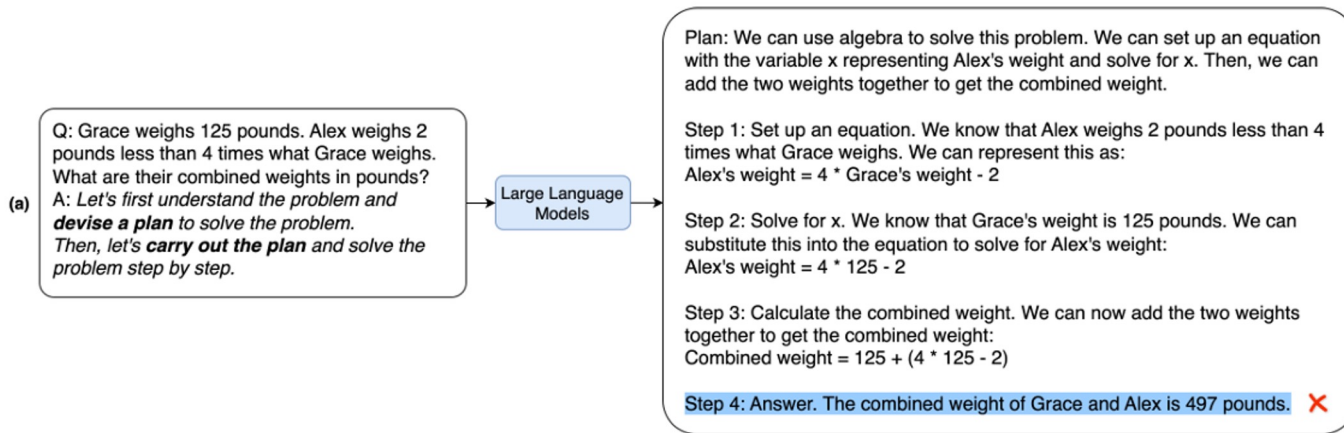
Therefore, the answer (arabic numerals) is

LLMs

60%

(c)

PS+ Prompting



Prompting for Answer Extraction

- Therefore, the answer (arabic numerals) is
- Desired form

Results and Observations

- Last Letter
 - “James Brown” → “sn”
- Coin Flip
 - whether a coin is still heads up after it is flipped or not flipped based on steps given in the questions

Table 2: Accuracy comparison on six math reasoning datasets. The best and second best results are boldfaced and underlined respectively.

Setting	Method (text-davinci-003)	MultiArith	GSM8K	AddSub	AQuA	SingleEq	SVAMP	Average
Zero-Shot	CoT	83.8	56.4	85.3	38.9	88.1	69.9	70.4
	PoT	92.2	57.0	85.1	<u>43.9</u>	<u>91.7</u>	70.8	<u>73.5</u>
	PS (ours)	87.2	<u>58.2</u>	<u>88.1</u>	42.5	89.2	<u>72.0</u>	<u>72.9</u>
	PS+ (ours)	<u>91.8</u>	59.3	92.2	46.0	94.7	75.7	76.7
Few-Shot	Manual-CoT	93.6	58.4	91.6	48.4	93.5	80.3	77.6
	Auto-CoT	95.5	57.1	90.8	41.7	92.1	78.1	75.9

Table 3: Accuracy on commonsense reasoning datasets.

Method	CSQA	StrategyQA
Few-Shot-CoT (Manual)	78.3	71.2
Zero-shot-CoT	65.2	63.8
Zero-shot-PS+ (ours)	71.9	65.4

Table 4: Accuracy on symbolic reasoning datasets.

Method	Last Letter	Coin Flip
Few-Shot-CoT (Manual)	70.6	100.0
Zero-shot-CoT	64.8	96.8
Zero-shot-PS+ (ours)	75.2	99.6

Results and Observations

No.	Trigger Sentence		GSM8K	SVAMP
1	Let's think step by step.	(*1)	56.4	69.9
2	<pre>import math import numpy as np # Question: example['question'] # Answer this question by implementing a solver() function. def solver(): # Let's write a Python program step by step, and then return the answer # Firstly, we need define the following variable:</pre>	(*2)	57.0	70.8
3	Extract variables and assign their corresponding numerals to these variables first and then solve the problem step by step.		50.5	69.5
4	Firstly, extract variables and their corresponding numerals. Then, calculate intermediate variables. Finally, solve the problem step by step.		54.8	70.8
5	Let's first understand the problem and devise a plan to solve the problem. Then, let's carry out the plan and solve the problem step by step.		58.2	72.0
6	Let's first understand the problem, extract relevant variables and their corresponding numerals, and make a plan. Then, let's carry out the plan, calculate intermediate variables (pay attention to correct numerical calculation and commonsense), solve the problem step by step, and show the answer.		59.3	75.7

Results and Observations

- 100 random problems from GSM8K

Method	Calculation	Missing	Semantic
Zero-shot-CoT	7%	12%	27%
Zero-shot-PS	7%	10%	26%
Zero-shot-PS+	5%	7%	27%

Summary & Future Work

- New zero-shot prompting method: PS and PS+ prompting
- PS+ prompting outperforms the previous zero-shot baselines across three types of reasoning problems
- Zero-shot PS+ prompting has the potential to outperform Few-shot manual-CoT prompting
- PS+ prompting can be used for non-reasoning tasks
- Refining plans
- Limitations:
 - Designing the prompt to guide LLMs to generate correct reasoning steps
 - Semantic misunderstanding errors still remain