DOI: 10.19884/j.1672-5220.202406009

Select-and-Answer Prompting: Facilitating LLMs for Improving Zero-Shot Reasoning

WANG Yufang[1], TANG Xuesong[1,2*], HAO Kuangrong[1,2]
1. College of Information Science and Technology, Donghua University, Shanghai 201620, China
2. Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education, Donghua University, Shanghai 201620, China

Abstract: Large language models (LLMs) have demonstrated remarkable generalization abilities across multiple tasks in natural language processing (NLP). For multi-step reasoning tasks, chain-of-thought (CoT) prompting facilitates step-by-step thinking, leading to improved performance. However, despite significant advancements in LLMs, current CoT prompting performs suboptimally on smaller-scale models that have fewer parameters. Additionally, the common paradigm of few-shot CoT prompting relies on a set of manual demonstrations, with performance contingent on the quality of these annotations and varying with task-specific requirements. To address these limitations, we propose a select-and-answer prompting method (SAP) to enhance language model performance on reasoning tasks without the need for manual demonstrations. This method comprises two primary steps: guiding the model to conduct preliminary analysis and generate several candidate answers based on the prompting; allowing the model to provide final answers derived from these candidate answers. The proposed prompting strategy is evaluated across two language models of varying sizes and six datasets. On ChatGLM-6B, SAP consistently outperforms few-shot CoT across all datasets. For GPT-3.5, SAP achieves comparable performance to few-shot CoT and outperforms zero-shot CoT in most cases. These experimental results indicate that SAP can significantly improve the accuracy of language models in reasoning tasks.

Keywords: zero-shot learning; large language model (LLM); reasoning problem; chain-of-thought (CoT) prompting
CLC number: TP3-05 **Document code**: A
Article ID: 1672-5220(2025)05-0513-10

Open Science Identity (OSID)

0 Introduction

Large language models (LLMs)[1] have achieved remarkable success across a wide range of natural language processing tasks. Compared to previous large pre-trained language models (PLMs)[2], LLMs offer stronger performance in many areas. When confronted with complex tasks that demand multi-step reasoning, PLMs exhibit limited reasoning abilities[3], although PLMs have parameters reaching the scale of billions. The size of these parameters makes fine-tuning extremely expensive. Furthermore, the parameters and architectures of most PLMs are not released as open source[4]. Thus, LLMs are often applied to solve multi-step reasoning problems.

Various approaches have been proposed to improve the reasoning capabilities of LLMs. Previous efforts include supervised fine-tuning methods, which involve fine-tuning LLMs with large amounts of training data[5]. Studies have also explored iterative refinement of answers and knowledge enhancement through retrieval from external knowledge bases[6]. Wei et al.[7] introduced chain-of-thought (CoT) prompting. This technique comprises two main methods. The first is zero-shot CoT, which involves adding a single prompt after the question, such as "Let's think step by step", to facilitate LLMs' reasoning process generation. This approach does not require input-output demonstrations. The second method, known as few-shot CoT, provides LLMs with step-by-step reasoning examples instead of standard question-answer pairs. In practice, few-shot CoT has achieved stronger performance than zero-shot CoT[8]. The differences between zero-shot CoT and few-shot CoT are shown in Fig. 1. However, these reasoning examples depend on human annotation, requiring consistent formatting for different types of tasks like arithmetic and commonsense reasoning. Additionally, CoT prompting proves effective only in models with hundreds of billions of parameters; neither few-shot CoT nor zero-shot CoT benefits smaller models[9].

To address the reliance on human annotation and improve the reasoning abilities of smaller models, we propose a select-and-answer prompting method (SAP). The core motivation behind SAP is to enhance the

Received date: 2024-06-21
Foundation item: National Natural Science Foundation of China (No. 62176052)
* Correspondence should be addressed to TANG Xuesong, email: tangxs@dhu.edu.cn
Citation: WANG Y F, TANG X S, HAO K R. Select-and-answer prompting: facilitating LLMs for improving zero-shot reasoning [J]. *Journal of Donghua University (English Edition)*, 2025, 42(5): 513-522.

performance of LLMs in reasoning tasks without requiring extensive manual intervention. SAP introduces a two-step process that integrates reasoning with answer extraction in a structured manner, enabling models to handle complex reasoning tasks more efficiently. Thus, SAP bridges the gap between the need for human involvement in CoT prompting and the limitations of smaller models in handling reasoning tasks.

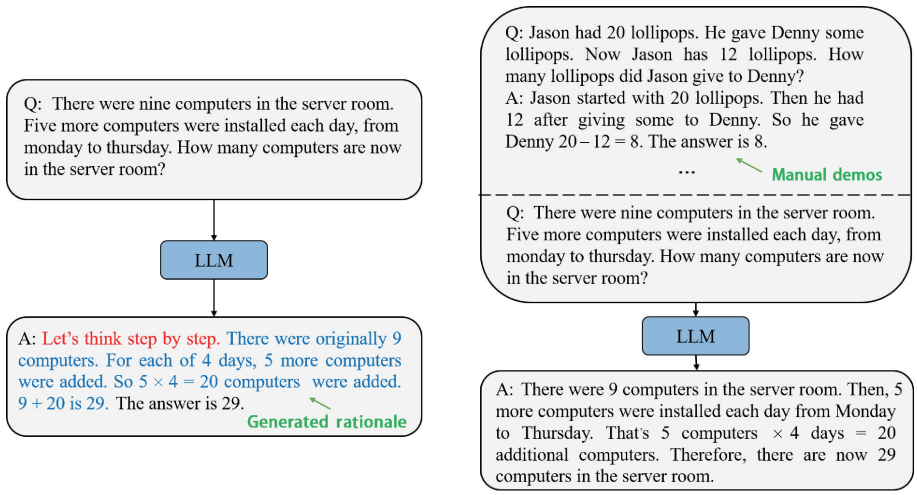

Fig. 1 Sample inputs and outputs of common CoT prompting methods for LLMs: (a) zero-shot CoT; (b) few-shot CoT

SAP consists of two primary steps. 1) The "select" prompt uses a modified version of zero-shot CoT, directing LLMs to perform a preliminary analysis of the problem and generate multiple candidate answers. The instruction prompt has been extended from "Let's think step by step" to include the directive "give several candidate answers", establishing a clear format for LLMs' outputs. 2) The "answer" prompt combines the original question with the generated candidates, enabling the model to select the most accurate answer from these candidates. However, the two-step process still appears somewhat redundant and could be further optimized in future research.

SAP offers three key advantages.

1) It eliminates the need for human-annotated examples, making it more scalable across different reasoning tasks.

2) It simplifies the reasoning process by structuring candidate answers, which in turn enables the efficient extraction of correct responses.

3) It significantly enhances reasoning performance without requiring billions of parameters, making it suitable for smaller LLMs.

We evaluate SAP on six benchmark datasets across three reasoning tasks: arithmetic reasoning (GSM8K[10], AddSub[11], SingleEq and SVAMP[12]), commonsense reasoning (CommonsenseQA[13]), and symbolic reasoning (last letter concatenation). Our experiments demonstrate that SAP matches or outperforms few-shot CoT in GPT-3.5 and significantly outperforms it in ChatGLM-6B[14] across all tasks. This suggests that SAP can effectively transfer prompt-based reasoning capabilities to smaller models, improving both efficiency and accuracy.

1 Related Work

1.1 Reasoning task

Solving multi-step reasoning tasks has been an active research area in the past few years. Researchers have proposed benchmarks for many reasoning tasks. These include commonsense reasoning (requiring common sense knowledge) and multi-hop reasoning[15] (requiring context understanding and multi-step inferences). Other types are arithmetic reasoning[16], involving mathematical concepts, and logical reasoning, based on rule comprehension and logical judgment. To enhance the precision of language models in reasoning tasks, researchers have pursued various strategies. One approach involves fine-tuning LLMs to directly generate the ultimate answer. For instance, Geva et al.[17] trained an NLP model by utilizing an interpretation produced by a fine-tuned GPT model. Experimental findings indicate that this trained model exhibits improved accuracy, particularly on CSQA datasets. A more advanced and widely adopted method is the introduction of CoT prompting, which guides the model through a step-by-step reasoning process to arrive at the final answer, rather than generating the result directly.

1.2 In-context learning (ICL)

ICL is pivotal to LLM reasoning[18]. It enhances output accuracy by introducing examples into the input, allowing the LLM to cope with a variety of tasks[19]. Research centered on ICL has outlined various avenues for refining LLM performance. Rubin et al.[20] proposed the ability to retrieve demonstrations relevant to the test

instance, dynamically providing relevant training examples for a given test input. Mishra et al. [21] suggested expanding examples by refining information, such as embedding task instructions, thereby enriching the learning process. Additionally, adjusting the output probabilities of the LLM, rather than directly calculating the likelihood probability of the target label, is also an effective method. However, the research by Liu et al.[22] indicates that the effectiveness of ICL can be significantly influenced by the context of the selected examples. Specifically, fluctuations in performance are closely related to the format, order, and wording of prompts. In contrast, Min et al.[23] suggest that using incorrect labels in examples only slightly reduces performance, raising questions about the strict paradigm of input-output mapping.

1.3 CoT prompting

The proposal of CoT has ushered LLMs into a new stage of reasoning ability. CoT prompting represents a gradient-free technique, guiding LLMs to generate a series of coherent intermediate reasoning steps that ultimately lead to the solution of a problem[24]. Kojima et al.[25] paved the way for generating reasoning steps without the need for manual demonstration. They posited that LLMs are adept zero-shot reasoners, with the most prominent method involving appending the prompt "Let's think step by step" after a reasoning question. The CoT reasoning, often denoted as few-shot CoT, revolves around the key idea of inserting multi-step reasoning paths before generating the final answer to achieve the desired outcome and stimulate emergent reasoning abilities[26]. Various methods were proposed to enhance CoT prompting, including representing the reasoning process using programming language, complex structures such as trees and graphs, as well as task decomposition and combining different prompts. In least-to-most prompting[27], complex problems are reduced to sub-problems, and then the sub-problems are solved sequentially. The other trend is to vote over multiple reasoning paths for a test question[28]. Zelikman et al.[29] suggested using prompts to enhance the fundamental principles of train carriages. Lu et al.[30] combined CoTs and error analysis to propose a new prompt method, EAPrompt. Jiang et al.[31] proposed LongLLMLingua, which improved the perception ability of LLMs in long context scenarios through prompt compression. However, such approaches typically require large LLMs with more than 100 billion (B) parameters, making them challenging to directly apply to small LLMs.

2 Methods

2.1 Overview

To overcome these limitations and to improve the reasoning performance of smaller models, we propose SAP, which is a new type of zero-shot prompting method and consists of two main steps. In Step 1, appropriate prompting statements are used to elicit reasoning in the LLM, generating possible candidate answers. In Step 2, the candidate answers obtained from the previous step, together with the prompting statements, are used to guide the LLM in reanalyzing the original question and evaluating the candidate answers.

2.2 Two-step reasoning strategy

In the process of generating candidate answers, our primary goal is to improve the accuracy of the LLM analysis to ensure that the resulting candidate answers contain the correct answers. At the same time, we need to clearly define tasks for the LLM so that it can provide answers in a specific format to facilitate subsequent steps. When constructing prompt statements, we should follow two criteria. First, we ensure that the prompts do not affect the analysis of the original question and allow the LLM to understand the subtask that comes up with multiple candidate answers. Second, we need to provide a clear template so that candidate answers can be provided in a fixed format. We refer to the method proposed in zero-shot CoT, where specific prompt-triggering sentences are used to convert input data into prompts with simple templates. Meanwhile, the construction of the template should consider inducing the LLM to generate the intermediate inference process to improve the reliability of the output candidate answers, as shown in Algorithm 1.

Algorithm 1: SAP

Given: question Q; prompt p_{generate} for generating candidate answers; LLM for generating candidate answers $\text{LLM}_{\text{generate}}$; LLM for deriving final answers $\text{LLM}_{\text{answer}}$; answer prompt p_{answer}.

Step 1: generate candidate answers

Input: Q, p_{generate}, and $\text{LLM}_{\text{generate}}$.

Process: Use $\text{LLM}_{\text{generate}}$ with p_{generate} to generate several candidate answers $[a_1, a_2, \cdots, a_n]$, along with their corresponding rationales, based on Q.

Output: rationale and $[a_1, a_2, \cdots, a_n]$.

Step 2: obtain final answers

Input: Q, p_{answer}, $[a_1, a_2, \cdots, a_n]$ and $\text{LLM}_{\text{answer}}$.

Process: Use $\text{LLM}_{\text{answer}}$ with p_{answer} to analyze $[a_1, a_2, \cdots, a_n]$ and derive the final answer A.

Output: A.

Specifically, we use a simple template to modify the input question x into prompt x' in Step 1: "Q: $[X]$. A: $[T]$", where $[X]$ is the input slot for the input question, and $[T]$ is the handcrafted trigger sentence slot for triggering the LLM to generate a post-ordered inference process. In this task, the reasoning process involves initial analysis of the question and the presentation of candidate answers. In zero-shot CoT, the

instructions in the input slot [T] include the trigger instruction "Let's think step by step". As shown in Fig. 2, SAP follows the above template. To implement this, we manually design a set of trigger prompts as follows: "Q: [X]. A: Let's analyze the problem step by step and give several candidate answers at the end of the answer, starting with: Possible candidate answers".

Referring to zero-shot CoT, SAP defaults to generating output with a greedy decoding strategy (1 output chain). In addition, when considering the design of the trigger sentence, we give detailed thought to the instructions. Specifically, in the trigger sentence, referring to zero-shot CoT, we use the sentence "Let's analyze the problem step by step" to require the LLM to perform analytical reasoning in as much detail as possible, enhancing the LLM's ability to generate relevant and important reasoning steps. To ensure that the output of the LLM follows a fixed format, we emphasize the sentence "at the end of the answer, starting with: Possible candidate answers" to explicitly instruct the LLM not to ignore tasks that output possible candidate answers. Adding important trigger sentences increases the credibility of the generated candidate answers. As shown in Fig. 2 (a), at the end of Step 1, the LLM generates inference text that contains the answer. In general, the strategy of adding specific descriptions to trigger sentences provides a new way to improve the zero-shot performance of complex reasoning.

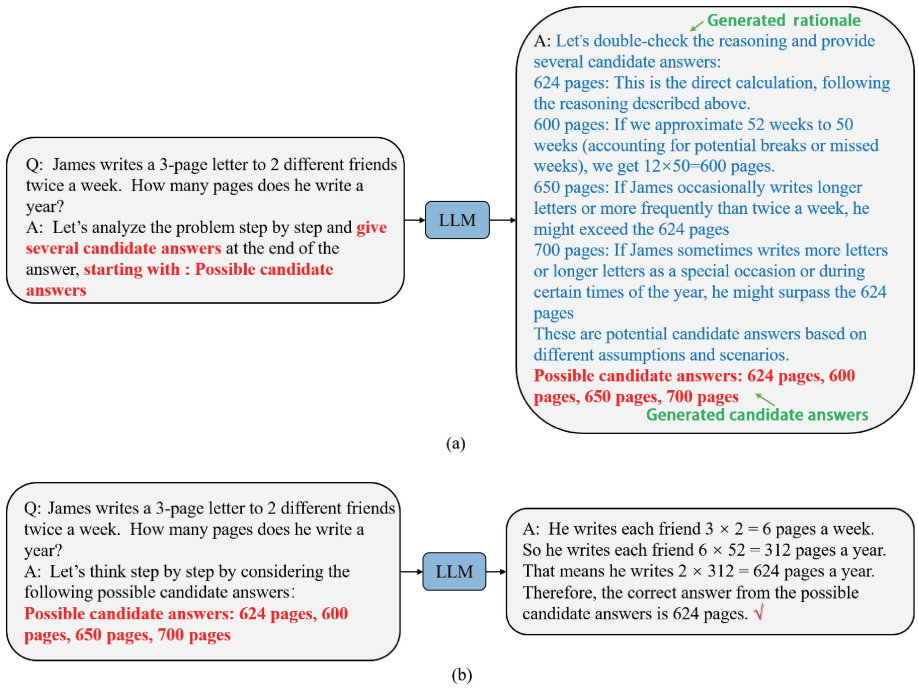

Fig. 2 Example inputs and outputs of GPT-3.5 with SAP: (a) reasoning for generating candidate answers; (b) final answer based on selected candidate options

We designed another prompt trigger sentence for the second reasoning step, enabling the LLM to consider the candidate answer generated in Step 1 in the reasoning of the next step. In addition to inducing the LLM to consider candidate answers during reasoning, the prompt also includes the answers generated in the first step. Since we designed a fixed prompt format in Step 1: "Possible candidate answers", we can extract the desired instruction information from the first prompt answer in Step 2. This allows the LLM to return the final answer by considering the candidate answers, as shown in Fig. 2 (b).

For models with hundreds of billions of parameters, such as GPT-3.5, we used the above approach to simply build templates. For a smaller model, such as ChatGLM-6B, we initially adopted an LLM to generate the first step inference responses that might improve the performance of the smaller model. To achieve this, we selected a small subset of the dataset and followed a manually constructed trigger prompt to an LLM (GPT-3.5 in this paper) to generate the answer to the question in Step 1. This includes intermediate reasoning steps and possible candidate answers in a fixed format. We ensured the reasonableness of the chain of reasoning steps by checking whether the candidate answers generated by the large model contained ground truth answers. With this approach, we obtained an enhanced dataset. In this dataset, a subset of the problem is paired with a candidate that leads to correct reasoning and a high-confidence answer. Therefore, we can refine the reasoning power into smaller models by fine-tuning the generated intermediate steps.

In Step 2, models with hundreds of billions of parameters are processed in the same way as smaller LLMs. Specifically, the trigger sentence for reasoning in

Step 2, along with the reasoning result from Step 1, is input into the unfine-tuned LLM to get the final answer.

3 Experimental Setup

3.1 Datasets

SAP is evaluated on six baseline datasets from three types of reasoning problems. Arithmetic reasoning includes the following datasets: 1) GSM8K, a high-quality multilingual dataset of primary-level math word problems created by human authors; 2) SVAMP, derived by extending an existing dataset; 3) AddSub, a dataset covering the addition and subtraction arithmetic word problem; 4) SingleEq, a dataset containing a single equation elementary algebra word problem that performs multiple mathematical operations on non-rational numbers and a variable. For commonsense reasoning, the assessment set includes CSQA, requiring the application of different types of commonsense knowledge to obtain correct answers on multiple-choice questions. For symbolic reasoning, the evaluation covers the last letter concatenation dataset (named Last Letters), which is required to solve problems connecting the last letter of a word in a name, e. g., "James Brown" to "sn". The datasets used in the experiment are summarized in Table 1.

Table 1 Details of datasets used in experiment

Dataset	Answer	Samples	Avg words	Data split (filename)
GSM8K	Number	1 319	46. 9	Test. jsonl
SVAMP	Number	600	31. 8	SVAMP. json
AddSub	Number	395	31. 5	AddSub. json
SingleEq	Number	508	27. 4	Questions. json
CSQA	Option	1 221	27. 8	dev_rand_split. jsonl
Last Letters	String	500	15. 0	–

Notes: "Answer" indicates the expected format of the model's output; "Samples" denotes the number of test instances in each dataset; "Avg words" indicates the average number of words per question; "Data split (filename)" specifies the exact file used for evaluation, ensuring reproducibility.

3.2 Baselines and implementations

We compare our approach to three baseline approaches: zero-shot, zero-shot CoT, and few-shot CoT. The zero-shot baseline connects the test question with the prompt "Yes" as LLM input, generating an answer to a given question without the need for an intermediate step. Among them, the zero-shot CoT appends "Let's think step by step" to the prompt. Few-shot CoT specifically uses the manual-CoT method, which creates eight hand-crafted examples for demonstration.

We use GPT-3. 5 (GPT-3. 5-turbo version) as the baseline model. As a backbone language model, it is one of the most widely used LLMs and provides a standard application programming interface (API) for easy integration. We chose this LLM because it has the strongest CoT inference performance among public LLMs. It is easy to evaluate our approach on large models and provide intermediate reasoning steps needed to fine-tune smaller models. For a smaller LLM, we choose ChatGLM-6B of version 2, because the model's parameters are public and downloadable, and the model is large enough to generate the rationale for non-trivial mass. In our experiment with the greedy decoding strategy, the temperature is set to 0. For few-shot CoT, the number of examples k used in arithmetic reasoning data is set to 8. For the commonsense reasoning class data set, k is set to 7. In symbolic reasoning data sets, k is set to 4.

4 Results and Discussion

4.1 Discussions on main results

The accuracy of SAP, the existing zero-shot CoT and few-shot CoT on six reasoning datasets is listed in Table 2.

1) SAP enhances performance effectively and robustly by bootstrapping candidate answers across various settings of difficulty and tasks in most datasets. For instance, using GPT-3. 5, we achieved an accuracy of 79. 3% on GSM8K and 84. 0% on SVAMP. Across most datasets, SAP consistently matches or surpasses the accuracy compared to the few-shot CoT that requires manual demonstration. On the commonsense reasoning dataset CSQA, the accuracy of SAP (76. 3%) outperforms that of zero-shot (60. 9%) and zero-shot CoT (67. 2%), and it slightly trails behind few-shot CoT (78. 2%). Nonetheless, SAP offers more flexibility and stronger task adaptability: no manual construction of examples is required for each dataset.

2) SAP demonstrates significant improvements on smaller LLMs like ChatGLM-6B. For example, with ChatGLM-6B, the accuracy of SAP on GSM8K is 38. 0%, much higher than that of zero-shot (10. 7%). Additionally, although the SAP performance of the smaller ChatGLM-6B model is not as good as that of GPT-3. 5, the improvements brought by SAP on smaller LLMs significantly exceed those on larger LLMs.

Table 2 Accuracies of methods on six datasets

Model	Method	Accuracy/%					
		GSM8K	SVAMP	CSQA	SingleEq	AddSub	Last Letters
GPT-3.5	Zero-shot	68.4	71.1	60.9	74.0	73.6	54.8
	Zero-shot CoT	73.2	78.8	67.2	83.6	78.2	60.3
	Few-shot CoT	78.9	82.4	**78.2**	86.1	**87.3**	64.2
	SAP (ours)	**79.3**	**84.0**	76.3	**88.4**	83.1	**71.6**
ChatGLM-6B	Zero-shot	10.7	30.3	24.8	68.7	54.8	18.3
	Zero-shot CoT	28.0	34.3	28.9	70.2	58.2	25.1
	Few-shot CoT	32.4	33.1	31.2	66.5	60.9	28.9
	SAP (ours)	**38.0**	**40.5**	**39.1**	**72.1**	**69.2**	**36.7**

4.2 Self-consistency analysis

Self-consistency reduces the randomness of LLMs' outputs by generating N inference results and determining the final answer through majority voting. With self-consistency, consistent and improved results are typically expected from the method. Our goal is to investigate the impact of self-consistency on SAP, and whether they are mutually compatible. We evaluated the self-consistency of SAP and few-shot CoT on the GSM8K and CSQA datasets. Experiments were conducted with varying the number of sampling paths at a fixed temperature of 0.7. The results for few-shot CoT and SAP were obtained under identical conditions, and the results of all samples were aggregated through majority voting to derive final answers.

One limitation of self-consistency is that it incurs higher computational costs. We attempt a small number of sampling paths (five in this experiment) as a starting point to achieve most of the benefits without incurring excessive costs, as performance tends to saturate after a certain number of sampling paths in most cases (Fig. 3). Our research findings suggest that for ChatGLM-6B, SAP consistently outperforms few-shot CoT in most cases, while for GPT-3.5, SAP achieves comparable or superior performance to few-shot CoT. The results indicate a significant improvement in performance with our approach on smaller LLMs.

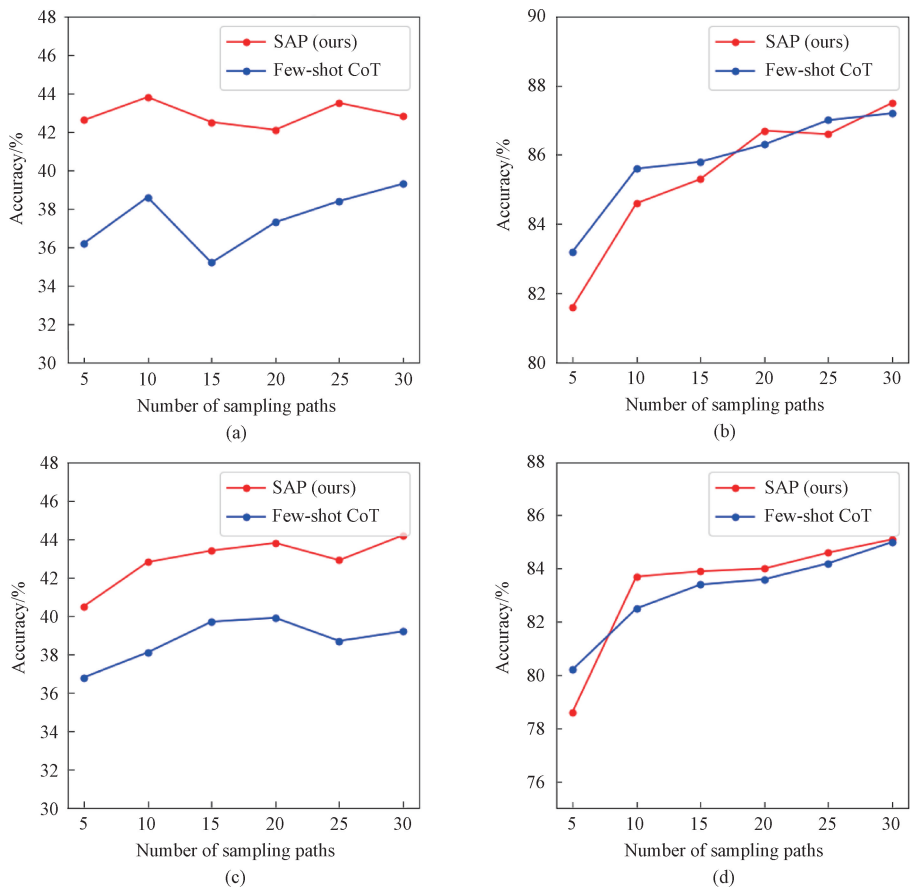

Fig. 3 Self-consistency analysis with varying number of sampling paths: (a) tested on GSM8K for ChatGLM-6B; (b) tested on GSM8K for GPT-3.5; (c) tested on CSQA for ChatGLM-6B; (d) tested on CSQA for GPT-3.5

4.3 Performance of prompts

Table 3 shows the performance comparison of five different input prompts, specifically the answer accuracy on the datasets GSM8K and CSQA, when the five input prompts are used in the first step. All these input prompts are variants of trigger sentences used in Step 1 of the SAP prompt strategy, and greedy decoding is employed. We observe that the accuracy is lower when the prompt sentence does not involve the task of guiding the model step-by-step through prompts and explicitly providing candidate answers. However, when we add more sentences regarding the intermediate thinking steps and the format of the task of providing candidate answers, the prompts perform the task better. These results indicate that when prompts include more detailed instructions for guiding LLMs, LLMs can generate high-quality reasoning text.

Table 3 Performance comparison of prompts with GPT-3.5 on GSM8K and CSQA

No.	Prompt sentence	Accuracy/%	
		GSM8K	CSQA
1	Understand the problem and propose several candidate answers.	70.5	62.5
2	Please analyze the problem step by step and give several candidate answers.	72.8	63.7
3	Let's think step by step and give several candidate answers.	74.3	64.8
4	Firstly, understand the problem, then give several candidate answers at the end of the answer, starting with: Possible candidate answers.	75.2	66.1
5	Let's analyze the problem step by step and give several candidate answers at the end of the answer, starting with: Possible candidate answers.	76.7	68.2

4.4 Effects of prompt selection

We also conducted ablation experiments and analyses to explore the key factors contributing to the significant improvements of SAP compared to the baseline, with the complete results shown in Fig. 4. Unlike previously proposed CoT methods, our main innovation lies in presenting an efficient answer selection strategy. This method guides the model to first select and then respond through a two-step prompt selection. Therefore, for the ablation experiments on the GSM8K dataset, we used few-shot CoT as the baseline and compared it with SAP without the prompt selection module (PSM) and the complete SAP. Additionally, we compared the reasoning path and direct answers of the self-consistency method, with the number of paths set to 10. The experimental results in Table 2 show that SAP's performance improvement is more significant on smaller LLMs than on larger ones, which is why we select ChatGLM-6B as the experimental model for the ablation experiments. Table 3 shows that relying solely on the second-step prompt or using only sampling decoding cannot achieve optimal performance. In other words, combining the first-step prompt selection with the reasoning path can lead to the best results.

Figure 4 demonstrates that SAP without PSM and the complete SAP can enhance performance compared to few-shot CoT. Additionally, the complete SAP outperforms all other methods, indicating the effectiveness of the PSM. Relying solely on the second-step prompt or using only sampling decoding does not yield optimal performance.

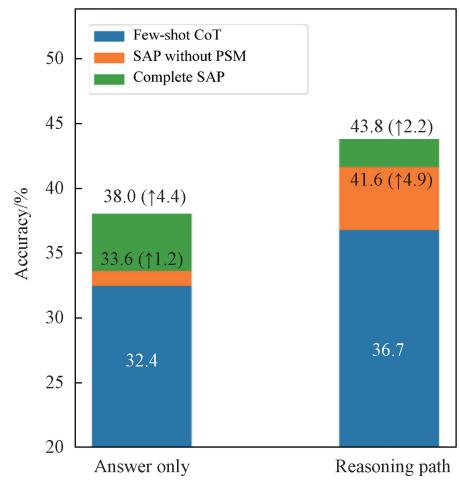

Fig. 4 Ablation study on GSM8K dataset with ChatGLM-6B

5 Conclusions

Previous research has demonstrated that LLMs exhibit reasoning abilities when guided by CoT prompting. However, the manual construction of CoT prompts relies heavily on expert craftsmanship. To alleviate this manual design burden, we propose the select-and-answer prompting method, SAP. It guides LLMs to conduct preliminary analysis, generate several candidate answers for the task, and then complete the original task based on these candidate answers. Evaluation on six datasets, covering three reasoning tasks, reveals that SAP outperforms previous zero-shot

baselines and consistently matches or exceeds the performance of few-shot CoT on multiple datasets. Our findings suggest that SAP can generate several high-confidence candidate answers, and SAP has the potential to outperform few-shot CoT prompting that requires manual demonstration, which could spark further developments of new CoT prompting methods.

References

[1] BROWN T B, MANN B, RYDER N, et al. Language models are few-shot learners[C]// Conference on Neural Information Processing Systems, NIPS '20: 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada. New York: NIPS, 2020: 1877-1901.

[2] LU P, QIU L, YU W H, et al. A survey of deep learning for mathematical reasoning [C]// Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2023: 14605-14631.

[3] SUN T, SHAO Y, QIAN H, et al. Black-box tuning for language-model-as-a-service [C]// Proceedings of the 39th International Conference on Machine Learning. New York: PMLR, 2022: 20841-20855.

[4] BAO K Q, ZHANG J Z, ZHANG Y, et al. Tallrec: an effective and efficient tuning framework to align large language model with recommendation[C]//Proceedings of the 17th ACM Conference on Recommender Systems. New York: ACM, 2023: 1007-1014.

[5] HSIEH C Y, LI C L, YEH C K, et al. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes [C]//Findings of the Association for Computational Linguistics: ACL 2023. Stroudsburg: ACL, 2023: 8003-8017.

[6] GAO Y F, XIONG Y, GAO X Y, et al. Retrieval-augmented generation for large language models: a survey [EB/OL]. (2023-12-18) [2024-06-02]. https://arxiv.org/abs/2312.10997v5.

[7] WEI J, WANG X Z, SCHUURMANS D, et al. Chain-of-thought prompting elicits reasoning in large language models[C]//Proceedings of the 36th Conference on Neural Information Processing Systems. New York: NIPS, 2022: 24824-24837.

[8] STOLFO A, JIN Z J, SHRIDHAR K, et al. A causal framework to quantify the robustness of mathematical reasoning with language models [C]//Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2023: 545-561.

[9] HENDRYCKS D, BURNS C, KADAVATH S, et al. Measuring mathematical problem solving with the MATH dataset[EB/OL]. (2021-03-05) [2024-06-02]. https://arxiv.org/abs/2103.03874.

[10] COBBE K, KOSARAJU V, BAVARIAN M, et al. Training verifiers to solve math word problems [EB/OL]. (2021-10-27) [2024-06-02]. https://arxiv.org/abs/2110.14168.

[11] HOSSEINI M J, HAJISHIRZI H, ETZIONI O, et al. Learning to solve arithmetic word problems with verb categorization[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Stroudsburg: ACL, 2014: 523-533.

[12] PATEL A, BHATTAMISHRA S, GOYAL N. Are NLP models really able to solve simple math word problems? [C]//Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online. Stroudsburg: ACL, 2021: 2080-2094.

[13] TALMOR A, HERZIG J, LOURIE N, et al. CommonsenseQA: a question answering challenge targeting commonsense knowledge [C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg: ACL, 2019: 4149-4158.

[14] DU Z X, QIAN Y J, LIU X, et al. GLM: general language model pretraining with autoregressive blank infilling [C]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2022: 320-335.

[15] JI H Z, KE P, HUANG S H, et al. Language generation with multi-hop reasoning on commonsense knowledge graph [C]// Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online. Stroudsburg: ACL, 2020: 725-736.

[16] LING W, YOGATAMA D, DYER C, et al. Program induction by rationale generation: learning to solve and explain algebraic word problems[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2017: 158-167.

[17] GEVA M, KHASHABI D, SEGAL E, et al. Did Aristotle use a laptop? A question answering benchmark with implicit reasoning strategies[J]. *Transactions of the Association for Computational Linguistics*, 2021, 9: 346-361.

[18] RADFORD A, WU J, CHILD R, et al.

Language models are unsupervised multitask learners [EB/OL]. (2019-02-14) [2024-06-02]. https://openai.com/research/language-unsupervised.

[19] GUU K, LEE K, TUNG Z, et al. Retrieval augmented language model pre-training [C]// Proceedings of the 37th International Conference on Machine Learning. New York: PMLR, 2020: 3929-3938.

[20] RUBIN O, HERZIG J, BERANT J. Learning to retrieve prompts for in-context learning [C]// Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg: ACL, 2022: 2655-2671.

[21] MISHRA S, KHASHABI D, BARAL C, et al. Cross-task generalization via natural language crowdsourcing instructions [C]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2022: 3470-3487.

[22] LIU H K, TAM D, MUQEETH M, et al. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning [C]// Proceedings of the 36th International Conference on Neural Information Processing Systems. New York: NIPS, 2022: 1950-1965.

[23] MIN S, LEWIS M, HAJISHIRZI H, et al. Noisy channel language model prompting for few-shot text classification [C]//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2022: 5316-5330.

[24] SUZGUN M, SCALES N, SCHÄRLI N, et al. Challenging BIG-bench tasks and whether chain-of-thought can solve them [C]//Findings of the Association for Computational Linguistics: ACL 2023. Stroudsburg: ACL, 2023: 13003-13051.

[25] KOJIMA T, GU S S, REID M, et al. Large language models are zero-shot reasoners [C]// Proceedings of the 36th International Conference on Neural Information Processing Systems. New York: NIPS, 2022: 22199-22213.

[26] JIN M, YU Q K, SHU D, et al. The impact of reasoning step length on large language models [C]//Findings of the Association for Computational Linguistics: ACL 2024. Stroudsburg: ACL, 2024: 1830-1842.

[27] ZHOU D, SCHÄRLI N, HOU L, et al. Least-to-most prompting enables complex reasoning in large language models [EB/OL]. (2022-05-21) [2024-06-02]. https://arxiv.org/abs/2205.10625.

[28] ZHU X Y, WANG J J, ZHANG L, et al. Solving math word problems via cooperative reasoning induced language models [C]// Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers. Stroudsburg: ACL, 2023: 4471-4485.

[29] ZELIKMAN E, WU Y, MU J, et al. Star: bootstrapping reasoning with reasoning [C]// Proceedings of the 36th International Conference on Neural Information Processing Systems. New York: NIPS, 2022: 15476-15488.

[30] LU Q Y, QIU B P, DING L, et al. Error analysis prompting enables human-like translation evaluation in large language models [C]// Findings of the Association for Computational Linguistics: ACL 2024. Stroudsburg: ACL, 2024: 8801-8816.

[31] JIANG H Q, WU Q H, LUO X F, et al. LongLLMLingua: accelerating and enhancing LLMs in long context scenarios via prompt compression [C]//Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Stroudsburg: ACL, 2024: 1658-1677.

选择与回答提示：引导大型语言模型提升零样本推理能力

王煜芳[1]，唐雪嵩[1,2*]，郝矿荣[1,2]
1. 东华大学 信息科学与技术学院，上海 201620
2. 东华大学 数字化纺织服装技术教育部工程研究中心，上海 201620

摘　要：大型语言模型（large language model，LLM）在自然语言处理（natural language processing，NLP）的多项任务中展示出显著的泛化能力。对于多步骤推理任务，思维链（chain-of-thought，CoT）提示有助于逐步思考，从而提升性能。然而，尽管 LLM 取得了重大进展，但目前的 CoT 提示技术在参数规模较小的模型上仍表现不佳。此外，常见的少样本思维链（few-shot CoT）提示依赖于一组人工编写的示例，其性能取决于这些示例的质量，并随任务的具体要求而变化。为了解决这些局限性，提出了一种名为“选择-回答提示”（select-and-answer prompting，SAP）的方法，以在无需人工编写示例的情况下增强语言模型在推理任务上的表现。该方法分为两步：引导模型进行初步分析，并基于提示生成多个候选答案；让模型从这些候选答案中选出最终答案。该文提出的提示策略针对 2 种不同规模的语言模型和 6 个数据集进行了评估。试验结果表明，在 ChatGLM-6B 上，SAP 的性能在全部数据集上均优于 few-shot CoT；在 GPT-3.5 上，SAP 的性能与 few-shot CoT 相当，且在大多数情况下优于 zero-shot CoT。这些结果表明，SAP 能显著提升语言模型在推理任务中的准确性。

关键词：零样本学习；大型语言模型；推理问题；思维链提示