Zero-Shot Chain-of-Thought Reasoning Guided by Swarm Intelligence Algorithms in Large Language Models

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have demonstrated remarkable performance across diverse tasks and exhibited impressive reasoning abilities by applying zero-shot Chain-of-Thought (CoT) prompting. However, due to the evolving nature of sentence prefixes during the pre-training phase, existing zero-shot CoT prompting methods that employ identical CoT prompting across all task instances may not be optimal. In this paper, we introduce a novel zero-shot prompting method that leverages swarm intelligence algorithms to dynamically generate diverse promptings for LLMs. Our approach involves initializing several CoT promptings, performing evolutionary operations based on LLMs to create a varied set, and utilizing the LLMs to select a suitable CoT prompting for a given problem. Additionally, a rewriting operation, guided by the selected CoT prompting, enhances the understanding of the LLMs about the problem. Extensive experiments conducted across ten reasoning datasets demonstrate the superior performance of our proposed method compared to current zero-shot CoT prompting methods on both black-box and open-source LLMs. Moreover, in-depth analytical experiments underscore the adaptability and effectiveness of our method in various reasoning tasks.

1 Introduction

The capacity for logical inference stands out as a defining characteristic of human intelligence, granting us the ability to engage in deduction, induction, and problem-solving. With the revolutionary advancement of pre-training (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2022, 2023), the rise of LLMs has firmly established itself as a cornerstone in the field of natural language processing (NLP), showcasing exceptional performance across a spectrum of NLP tasks. However, LLMs often face challenges in the nuanced domain of reasoning, prompting researchers to strategically leverage their embedded knowledge through the conditioning of LLMs on a limited set of illustrative examples, referred to as few-shot learning (Wei et al., 2022; Wang et al., 2023b), or through the provision of prompts for solving problems in the absence of illustrative examples, constituting a paradigm known as zero-shot learning (Kojima et al., 2022).

Current research mainly focuses on designing diverse prompting strategies to guide the reasoning processes of LLMs. For instance, Wei et al. (2022) propose the few-shot CoT prompting, involving the use of a limited number of manually demonstrated reasoning examples to enable LLMs to explicitly generate intermediate reasoning steps before predicting the final answer. Various approaches have been explored to eliminate the need for manually selected examples in few-shot CoT prompting. For instance, Kojima et al. (2022) introduce zero-shot CoT prompting by appending *"Let's think step by step"* to the target problem, PS+ prompting (Wang et al., 2023a) add *"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"* after the target problem, and RE2 prompting (Xu et al., 2023) add *"Read the question again"* combined with *"Let's think step by step"* to the target problem. However, these zero-shot CoT prompting methods employ uniform CoT prompting across all task instances. Given the ongoing evolution of sentence prefixes during the pre-training phase of extensive language models, using identical CoT prompting for all instances may introduce disruptions to predictive accuracy and potentially result in a degradation of overall performance. Consequently, a fundamental query emerges: Is it feasible to ascertain an appropriate CoT prompting for each instance within a discrete space?

Fortunately, swarm intelligence algorithms (SIA) (Mitchell, 1998; Hansen et al., 2003; Li and Tan, 2018) provide a solution. SIA represents a category of optimization algorithms inspired by the principles of natural evolution. Crossover, mutation, and selection steps in SIA can generate various CoT promptings. In this paper, we introduce a novel method guided by swarm intelligence algorithms named **E**volution **o**f **T**ought (**EoT**) prompting. The process begins by initializing several CoT promptings based on human design or auto-generation using large language models. Using LLMs as the optimizer within a swarm intelligence algorithm framework, we perform crossover and mutation operations on the initialized CoT promptings, generating a diverse set of new ones. Subsequently, we use LLMs to select a CoT prompting deemed suitable for the current problem. Furthermore, to deepen the understanding of LLMs of the current problem, a rewriting operation is performed on the selected CoT prompting. The LLMs engage in reasoning based on the rewritten problem. This strategy aims to capitalize on the diversity of CoT prompting generated through the SIA and problem rewriting to provide richer information that encourages the LLMs to attain a more profound understanding of the given problem.

To validate the effectiveness of our proposed zero-shot EoT prompting, we conduct a comprehensive series of experiments across ten datasets, covering arithmetic, commonsense, and symbolic reasoning. The experiments are carried out on black-box LLMs GPT-3.5-Turbo (OpenAI, 2022) and GPT-4 (OpenAI, 2023), as well as open-source LLM Llama-3-8B-Instruct[1]. Specifically, the results in mathematical reasoning indicate that our zero-shot EoT prompting outperforms existing zero-shot CoT prompting, with average improvements of 3.1% on GPT-3.5-Turbo. Its comparable performance to few-shot CoT prompting is particularly noteworthy, especially in arithmetic and symbolic reasoning. Additionally, extensive analytical experiments are conducted to gain a deeper understanding of the different components of zero-shot EoT prompting and the impact of various factors on EoT prompting.

2 Preliminaries

Zero-shot Chain-of-Thought Prompting In-context learning leverages a few demonstrations as a prompt and conducts inference without training the model parameters (Brown et al., 2020). Chain-of-thought (CoT) prompting (Wei et al., 2022) has been proposed as a type of in-context learning that decomposes the original problem into several small parts and achieves encouraging results on many complex reasoning tasks in large language models. Moreover, the zero-shot chain-of thought prompting (Kojima et al., 2022) has shown impressive effectiveness on various tasks in large language models by attaching a sentence before the reasoning process. For standard zero-shot CoT prompting, given the reasoning question $\mathcal{Q}$, zero-shot CoT specific instructions $\mathcal{T}$ like *"Let's think step by step."*, we formalize this simple yet fundamental solving paradigm as:

$$P(\mathcal{A}|\mathcal{T},\mathcal{Q}) = P(\mathcal{A}|\mathcal{T},\mathcal{Q},\mathcal{C})P(\mathcal{C}|\mathcal{T},\mathcal{Q}) \quad (1)$$

where $\mathcal{C}$ denotes a sampled rationale in natural language and $\mathcal{A}$ is the generated answer. As such, LLMs can perform complex reasoning by decomposing the problem into sequential or modular steps.

3 Method

Overview. We introduce our proposed zero-shot EoT prompting. EoT utilizes the large language model as an evolutionary optimizer, performing crossover and mutation operations on several given promptings to generate diverse promptings. Subsequently, EoT empowers LLMs to autonomously select the most suitable or optimal prompting from the generated set as the final prompting. Finally, EoT employs the chosen prompt to instruct LLMs in rewriting given problems, generating an intermediate reasoning process, and predicting the final answer for the input problem. Similar to zero-shot CoT prompting, our EoT prompting incorporates an answer extraction prompting, such as *"Therefore, the answer (arabic numerals) is"* to extract the answer for evaluation.

3.1 Prompt Generation Through Swarm Intelligence Algorithms

As depicted in Figure 1(a), zero-shot CoT prompting (Kojima et al., 2022) appends the same sentence *"Let's think step by step"* or the recently proposed Plan-and-Solve prompting (Wang et al., 2023a) connects the same sentence *"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"* to each instance, encouraging LLMs to generate multi-step reasoning. Given the continuous evolution of sentence

[1]https://github.com/meta-llama/llama3

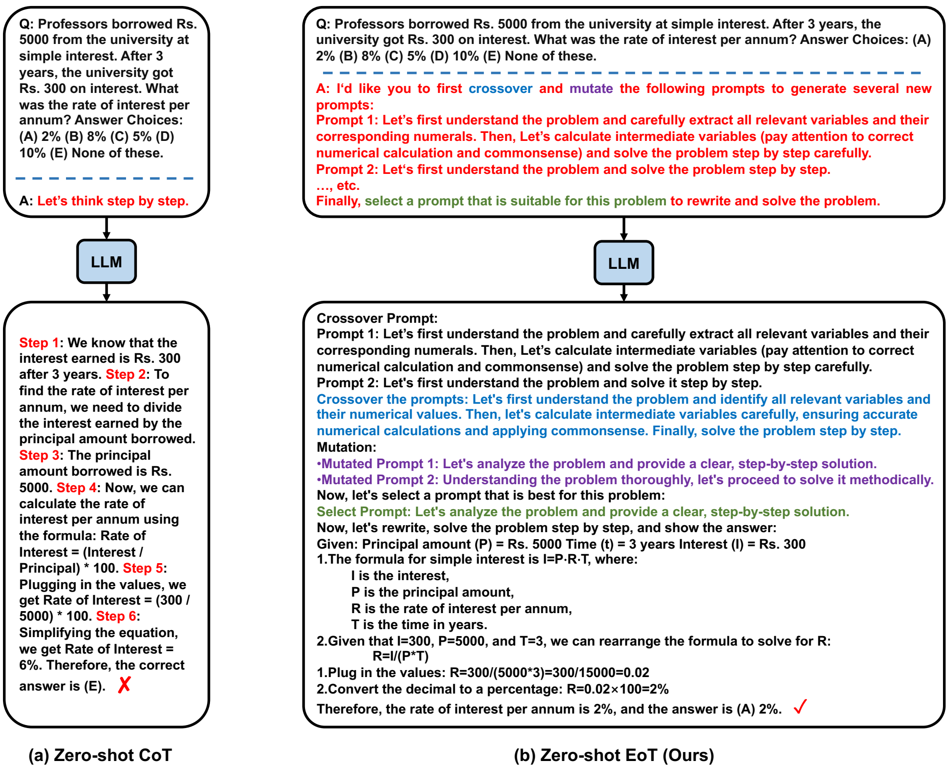

Figure 1: Example inputs and outputs of GPT-3.5-Turbo with (a) Zero-shot CoT prompting and (b) Zero-shot EoT prompting. Zero-shot CoT prompting attaches the sentence *"Let's think step by step"* for each instance to encourage LLMs to generate multi-step reasoning. Our proposed method, EoT prompting, uses the LLMs as an evolutionary optimizer and generates suitable CoT prompting for each instance.

prefixes during the pre-training phase of large language models, using identical CoT prompting for all instances may disrupt predictions and lead to a decline in performance.

To address these concerns, we aim to identify suitable CoT prompting for each instance of the current reasoning task within a discrete space before proceeding with the reasoning process. However, determining the most suitable CoT prompting for each instance in a discrete space poses a challenge. Fortunately, swarm intelligence algorithms provide a solution. We employ the large language model as an optimizer, executing crossover and mutation on the initialized CoT prompting, denoted as LLM-Crossover and LLM-Mutation. As illustrated in Figure 1(b), for a given problem $\mathcal{Q}$, we first initialize several CoT promptings $\mathcal{T}_1$ and $\mathcal{T}_2$. Subsequently, we first use the large language model as the evolutionary optimizer, applying the LLM-Crossover operation on $\mathcal{T}_1$ and $\mathcal{T}_2$, which is defined as:

$$\mathcal{T}_c = \text{LLM-Crossover}(\mathcal{T}_1, \mathcal{T}_2) \tag{2}$$

Then, we enable LLM-Mutation on the crossovered CoT prompting $\mathcal{T}_c$, which is defined as:

$$\mathcal{T}_m = \text{LLM-Mutation}(\mathcal{T}_c) \tag{3}$$

This leverages the powerful generative capability of the large language model to generate additional CoT promptings.

We aim to generate more high-quality promptings by evolving from the initial ones. To obtain good initial promptings, we use either auto-generated promptings (Zhou et al., 2023) or manual-designed promptings as the initial promptings. Additionally, pursuing a more diverse set of selectable CoT promptings, it is customary to subject the model to crossover and mutation operations iteratively. However, the temporal demand tends to escalate proportionally with the quantity of generated CoT promptings. Consequently, we opt for a default strategy of conducting a singular round of crossover and mutation operations to mitigate reasoning time. As illustrated in Figure 2, our analysis delves into the correlation between the number of CoT promptings (i.e., the population size N) generated through multiple rounds of crossover and mutation operations and the performance of LLMs.

3.2 Problem Rewriting with Generated Prompt and Answer Extraction

Based on the generated and initialized pool of CoT promptings, we enable the LLMs to select the most optimal or contextually suitable CoT prompting for the current problem $\mathcal{Q}$. Subsequently, to enhance the retention of the LLMs regarding the problem, we employ the selected CoT prompting to rewrite the question $\mathcal{Q}$ and instruct the LLMs to conduct reasoning. The formalization of this process is exemplified as follows:

$$P(\mathcal{A}|\mathcal{T}_o, \mathcal{Q}) = P(\mathcal{A}|\mathcal{T}_o, R(\mathcal{Q}), \mathcal{C})P(\mathcal{C}|\mathcal{T}_o, R(\mathcal{Q})) \quad (4)$$

Here, $\mathcal{T}_o$ denotes the selected CoT prompting by LLMs, $\mathcal{C}$ denotes a sampled rationale in natural language, $\mathcal{A}$ is the generated answer, and $R(\cdot)$ means rewriting the question $\mathcal{Q}$ with $\mathcal{T}_o$. For instance, in Figure 1b, for a given question $\mathcal{Q}$: *Professors borrowed Rs. 5000 from the university at simple interest. After 3 years, the university got Rs. 300 on interest. What was the rate of interest per annum? Answer Choices: (A) 2% (B) 8% (C) 5% (D) 10% (E) None of these.* We employ the chosen CoT prompting (*Let's analyze the problem and provide a clear, step-by-step solution.*) to rewrite the question $R(\mathcal{Q})$. Then, the LLMs generate an intermediate reasoning process and predict the final answer for the question $\mathcal{Q}$. Moreover, our method defaults to employing the greedy decoding strategy for the generation of output.

Similar to the zero-shot CoT prompting, our EoT prompting incorporates specific trigger sentences, such as *"Therefore, the answer (arabic numerals) is"*, into the sentences generated by LLMs through EoT prompting. Following this augmentation, the composite text is reintroduced to LLMs, producing the desired answer format. In Appendix A.2, we present the trigger sentences utilized for different reasoning tasks.

4 Experiments

4.1 Experimental Setup

Datasets We systematically evaluate the efficacy of our proposed method across ten datasets encompassing three main categories: arithmetic, commonsense, and symbolic tasks. For arithmetic reasoning tasks, we consider the following six arithmetic reasoning problem benchmarks: (1) MultiArith (Roy and Roth, 2015), (2) GSM8K (Cobbe et al., 2021), (3) AddSub (Hosseini et al., 2014), (4) AQuA (Ling et al., 2017), (5) SingleEq (Koncel-Kedziorski et al., 2015), and (6) SVAMP (Patel et al., 2021). SingleEq and AddSub comprise more straightforward problems that do not require multi-step task resolution calculations. Conversely, MultiArith, AQUA, GSM8K, and SVAMP present more intricate challenges, demanding multi-step reasoning for effective problem-solving. In the realm of commonsense reasoning, we include (7) CommonsenseQA (Talmor et al., 2019) and (8) StrategyQA (Geva et al., 2021). CommonsenseQA requires the application of diverse forms of commonsense knowledge for accurate answers. Meanwhile, StrategyQA tasks models with deducing implicit multi-hop reasoning to respond to posed questions. For symbolic tasks, we select Last Letter Concatenation and Coin Flip (Wei et al., 2022). Last Letter Concatenation challenges the model to concatenate the last letters of individual words. At the same time, the Coin Flip task requires the model to determine whether a coin remains in a heads-up position after being flipped or left undisturbed. Details on dataset statistics are provided in Appendix A.1.

Baselines We conduct a comparative analysis between our proposed zero-shot EoT prompting method and several task-specific zero-shot CoT prompting methods: (1) Zero-shot CoT prompting (Kojima et al., 2022), which appends a sentence *"Let's think step by step"* before the reasoning process; (2) Zero-shot PS and PS+ prompting (Wang et al., 2023a), employing a "plan-and-solve" strategy to guide the model throughout the inference process; (3) Zero-shot RE2 prompting (Xu et al., 2023), a plug & play approach that entails re-reading the question before engaging in the reasoning process; (4) APE prompting (Zhou et al., 2023), utilizing LLMs to generate instructions automatically and requiring additional training. We also compare our method with two few-shot CoT prompting methods: Few-shot Manual-CoT prompting (Wei et al., 2022), utilizing eight manually crafted examples as demonstrations, and Few-shot AuTo-CoT prompting (Zhang et al., 2023), which automatically selects examples through clustering for diversity.

Implementation Details We mainly use ChatGPT (GPT-3.5-Turbo-0613) (OpenAI, 2022) and Llama-3-8B-Instruct as the backbone language models. Regarding decoding strategy, we employ greedy decoding with a temperature setting of 0 and implement self-consistency prompting with a temper-

Table 1: Accuracy of six math reasoning datasets on GPT-3.5-Turbo with different zero-shot and few-shot CoT prompting methods. The boldfaced and underlined fonts indicate the best and the second results in the zero-shot settings, respectively.

Method	MultiArith	GSM8K	AddSub	AQuA	SingleEq	SVAMP	Average	Δ
Zero-Shot								
CoT	95.3	75.3	86.6	55.1	92.9	79.0	80.7	-
PS	92.3	76.3	85.8	56.7	90.2	75.8	79.5	**-1.2**
PS+	93.8	76.1	86.6	58.7	92.5	79.4	81.2	**+0.5**
RE2	**96.8**	76.9	88.6	<u>59.8</u>	91.7	79.7	82.3	**+1.6**
APE	93.3	**80.2**	<u>88.9</u>	59.4	**94.1**	<u>81.5</u>	<u>82.9</u>	**+2.2**
EoT (Ours)	<u>96.0</u>	<u>78.5</u>	**91.1**	**62.2**	<u>93.7</u>	**82.0**	**83.8**	**+3.1**
Few-Shot								
Manual-CoT	95.5	75.9	89.9	58.7	92.3	81.1	82.2	**+1.5**
AuTo-CoT	96.2	77.3	90.6	61.8	92.7	81.8	83.4	**+2.7**

ature setting of 0.7. We set the initial number of promptings to two to reduce inference time and perform one iteration of crossover and mutation. Furthermore, to fortify the robustness and generalizability of our proposed method, we conduct complementary evaluations utilizing GPT-4 (OpenAI, 2023). For the few-shot baselines, Manual-CoT and Auto-CoT, we adhere to the configurations outlined in the Wei et al. (2022) and Zhang et al. (2023). We adopt accuracy as our evaluation metric for all datasets.

4.2 Main Results

Results on Arithmetic Reasoning. Table 1 and Table 2 present a thorough performance comparison between our zero-shot EoT prompting and existing zero-shot and few-shot baselines on the arithmetic reasoning datasets with GPT-3.5-Turbo and Llama-3-8B-Instruct. In contrast to prevalent zero-shot CoT, PS, and PS+ prompting methods, our EoT prompting exhibits notable improvements in performance across six arithmetic reasoning datasets, showcasing particularly significant improvements on the AddSub, SVAMP, AQuA, GSM8K and SingleEq datasets. Furthermore, on average, our EoT prompting achieves a 3.1% and 2.6% score improvement over zero-shot CoT prompting and PS+ prompting methods on GPT-3.5-Turbo. EoT prompting also achieves a 1.6% and 2.1% score improvement over zero-shot CoT prompting and PS+ prompting methods on Llama-3-8B-Instruct. Concerning the zero-shot RE2 prompting, our EoT prompting outperforms it across five datasets on both GPT-3.5-Turbo and Llama-3-8B-Instruct. The observed similarity between the zero-shot RE2 prompting, characterized by repetitive questions, and our approach of rewriting questions using CoT prompting generated via swarm intelligence algorithms suggests the advantageous impact of enhancing the model's capacity to retain questions on the reasoning process. Compared to the automatic prompting generation method, APE, our method improves average performance by 0.9 and 2.3% on GPT-3.5-Turbo and Llama-3-8B-Instruct, respectively. Concurrently, we compare our proposed EoT prompting with a few-shot methods: Manual-CoT and Auto-CoT. The results indicate that our proposed EoT prompting surpasses Manual-CoT and Auto-CoT on six and five arithmetic reasoning datasets on GPT-3.5-Turbo, respectively, suggesting the effectiveness of our zero-shot EoT prompting in achieving comparable results to few-shot methods in arithmetic reasoning datasets without the need for example selection.

Results on Commonsense Reasoning and Symbolic Reasoning. Table 3 shows the result on two commonsense reasoning datasets. Our EoT prompting exhibits superior performance in the zero-shot setting relative to zero-shot CoT prompting, PS prompting, PS+ prompting, RE2 prompting, and APE methods on two commonsense reasoning datasets. Conversely, compared to two few-

Table 2: Accuracy of six math reasoning datasets on Llama-3-8B-Instruct with different zero-shot and few-shot CoT prompting methods. The boldfaced and underlined fonts indicate the best and the second results in the zero-shot settings, respectively.

Method	MultiArith	GSM8K	AddSub	AQuA	SingleEq	SVAMP	Average	Δ
Zero-Shot								
CoT	<u>95.2</u>	<u>80.4</u>	85.3	50.0	<u>90.4</u>	83.6	<u>80.8</u>	-
PS	92.2	78.6	<u>87.6</u>	47.6	89.6	83.3	79.8	**-1.0**
PS+	94.7	79.1	86.3	48.4	89.4	<u>83.7</u>	80.3	**-0.5**
RE2	94.5	80.1	86.8	48.0	90.0	**84.0**	80.6	**-0.2**
APE	92.3	78.5	86.1	<u>50.4</u>	89.8	83.2	80.1	**-0.7**
EoT (Ours)	**95.3**	**81.7**	**89.4**	**53.1**	**91.4**	83.5	**82.4**	+1.6
Few-Shot								
Manual-CoT	95.8	81.7	86.8	52.0	91.5	83.9	82.0	+1.2
AuTo-CoT	96.8	82.0	87.6	52.4	92.1	84.1	82.5	+1.3

shot methods, Manual-CoT and Auto-CoT, our zero-shot EoT prompting demonstrates comparatively lower performance on these two commonsense reasoning datasets. This observation implies that commonsense reasoning problems may necessitate a certain degree of demonstration to guide the model reasoning process.

We also show the result on two symbolic reasoning datasets: Last Letters and Coin Flip. Our EoT prompting performs better than zero-shot CoT prompting, PS prompting, PS+ prompting, and the RE2 prompting methods on these two symbolic reasoning datasets, especially in the Last Letter dataset. In contrast to few-shot methods, Manual-CoT, and Auto-CoT, our EoT prompting excels relative to these methods in the Last Letter dataset while demonstrating comparable performance in the Coin Flip dataset. This observation suggests the effectiveness of our zero-shot EoT prompting in achieving comparable results to few-shot methods in symbolic reasoning datasets without the need for example selection.

5 Additional Experiments and Analysis

5.1 Results of EoT Prompting in GPT-4

To evaluate the performance of our proposed zero-shot EoT prompting with more powerful models, as shown in Table 4, we conduct experiments on GPT-4 using three arithmetic reasoning datasets: AQuA, AddSub, and SVAMP. We compare our zero-shot EoT prompting against four alternative methods: zero-shot CoT prompting, PS+ prompting, RE2 prompting, and APE prompting. The results presented in Table 4 reveal that our zero-shot EoT prompting yields superior performance compared to these methods, suggesting that our proposed method maintains robust performance advantages when applied to more powerful language models.

5.2 Ablation Study of EoT

We perform the ablation study of our EOT prompting measured on four math reasoning datasets under the zero-shot setting to understand the importance of different factors. As delineated in Table 5, the notations ’R’, ’C’, and ’M’ denote the operations of rewrite, crossover, and mutate, respectively. Our observations indicate that refraining from employing EoT prompting for problem rewriting results in a discernible decline in model performance across all tasks. This underscores the importance of augmenting the model’s comprehension of problems through a more profound engagement, thereby fostering more effective inference. Furthermore, while generating our EoT promptings, the omission of crossover or mutation processes results in a significant performance decrease across all tasks except the SVAMP dataset. Notably, the AQuA dataset exhibits a pronounced performance degradation, emphasizing the indispensability of the crossover and mutation processes in the effec-

Table 3: Accuracy of commonsense reasoning and symbolic reasoning datasets on GPT-3.5-turbo and Llama-3-8B-Instruct with different zero-shot and few-shot CoT prompting methods. CSQA denotes CommonsenseQA

Method	CSQA	StrategyQA	Last Letters	Coin Flip	CSQA	StrategyQA	Last Letters	Coin Flip
	GPT-3.5-Turbo				*Llama-3-8B-Instruct*			
Few-Shot								
Manual-CoT	75.3	70.1	75.7	99.2	74.5	72.6	73.4	99.0
AuTo-CoT	77.1	71.3	76.3	99.6	75.6	72.9	74.8	99.6
Zero-Shot								
CoT	64.9	65.7	72.6	98.6	67.1	68.4	71.5	97.2
PS	68.6	66.4	71.3	97.0	68.4	67.9	70.8	96.8
PS+	70.9	67.8	70.4	97.6	69.1	68.7	71.9	98.0
RE2	71.5	68.1	74.3	97.6	68.9	70.2	71.5	98.2
APE	69.1	70.6	-	-	71.6	69.8	-	-
EoT (ours)	**73.1**	**69.9**	**77.0**	**99.0**	**72.1**	**71.8**	**76.4**	**99.4**

Table 4: Results of different methods measured on three math reasoning datasets with GPT-4.

Method	AQuA	AddSub	SVAMP
Zero-shot CoT	72.8	94.9	89.7
Zero-shot PS+	73.2	96.5	89.2
Zero-shot RE2	74.0	96.2	90.1
Zero-shot APE	73.6	93.7	90.1
Zero-shot EoT (ours)	**76.4**	**97.5**	**92.9**

tive generation of our EoT prompting.

Table 5: Ablation study of EoT measured on four math reasoning datasets with GPT-3.5-Turbo. 'R', 'C', and 'M' denote rewrite, crossover, and mutate, respectively.

Method	AQuA	AddSub	SVAMP	GSM8K
EoT	62.2	91.1	82.0	78.5
-w/o R	61.4	90.1	80.7	76.4
-w/o C	58.7	89.1	82.3	76.2
-w/o M	57.5	88.1	81.1	76.9

5.3 Results of Prompting with Self-Consistency

Existing research suggests that the CoT prompting method can be enhanced through the incorporation of self-consistency (Wang et al., 2023b). This is achieved by generating N reasoning results, with the final answer determined by a majority voting process. Our interest is additionally piqued by the prospect of further augmenting the proposed EoT prompting through self-consistency. Consequently, experimental validations are conducted across four arithmetic reasoning datasets: AddSub, AQuA, SingleEq, and SVAMP. As depicted in Table 6, the comparative assessment involves an analysis of the performance of zero-shot CoT prompting, PS+ prompting, and RE2 prompting after applying the self-consistency method. Our EoT prompting exhibits superior performance across diverse arithmetic reasoning datasets when compared to these baselines.

Table 6: Results of different methods in a zero-shot setting with self-consistency measured on four math reasoning datasets with GPT-3.5-Turbo.

Method	AddSub	AQuA	SingleEq	SVAMP
CoT +SC	87.1	62.6	94.5	80.6
PS+ +SC	88.6	63.0	94.1	81.1
RE2 +SC	89.6	63.4	94.9	80.8
EoT +SC (ours)	**92.9**	**65.4**	**95.5**	**83.9**

5.4 Effect of Population Size

In our prior experiments, we strategically employ the EoT prompting method to facilitate a singular round of crossover and mutation operations, aiming to optimize inference speed. In this context, our objective is to systematically verify the relationship between the number of our EoT promptings (i.e., represented as the population size N) generated during multiple rounds of crossover and mutation operations and the ensuring model performance. As depicted in Figure 2, we conduct the experiments across four arithmetic reasoning datasets, including SingleEq, AddSub, SVAMP, and CSQA. The results manifest a discernible positive correlation, wherein an increased quantity of CoT promptings (i.e., a larger population size N) corresponds to a consistent enhancement in the model's performance. Thus, in scenarios where inference speed is either of lesser concern or can be overlooked, our EoT prompting affords substantial performance gains. This empirical evidence substantiates the efficacy of our proposed approach.

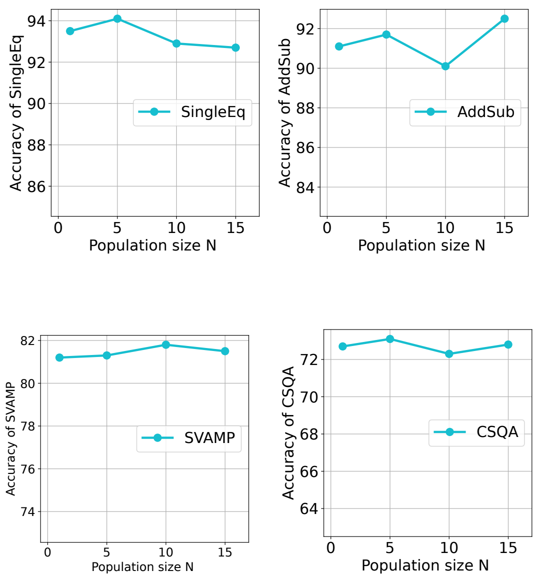

Figure 2: Results of different population size N measured on four math reasoning datasets with GPT-3.5-Turbo.

5.5 Whether the Selections Made by LLMs Are Random?

Our method utilizes LLMs to select promptings suitable for the current problem. However, are the selections made by the large language models random? We first iteratively generate 10 CoT promptings using the LLM to investigate this. Then, we allow the LLM to select the most appropriate CoT promptings and perform reasoning. Additionally, we randomly sampled CoT promptings from the ten generated promptings and conducted reasoning. We conduct experiments on the AQuA and SVAMP datasets using GPT-3.5-Turbo, and the results are depicted in Figure 3. It is evident that the performance of LLM-based selection significantly outperforms random selection, suggesting

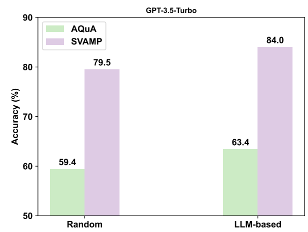

Figure 3: Performance of random selection and LLM-based selection on GPT-3.5-Turbo

that the selections made by LLMs are not random but guided by the robust prior knowledge of LLMs.

6 Related Work

Chain-of-Thought Prompting Built upon in-context learning (Brown et al., 2020), the recently introduced CoT prompting (Kojima et al., 2022; Wei et al., 2022; Wang et al., 2023b) significantly enhances the reasoning capabilities of LLMs. CoT prompting not only deepens the model's understanding of subtle questions and their underlying logic but also generates a series of explicit reasoning steps. Subsequent works (Wang et al., 2023a; Schaeffer et al., 2023; Zhang et al., 2023; Xu et al., 2023) have proposed different approaches to address complex problems. Our EoT prompting, by treating LLMs as evolutionary optimizers and generating distinct discrete CoT promptings for each instance, demonstrates superior performance across various reasoning problems.

Additional related works can be found in Appendix C

7 Conclusion

This paper introduces *EoT*, a novel zero-shot CoT prompting method. EoT prompting generates diverse CoT promptings tailored to specific instances within a task through swarm intelligence algorithms. The proposed method surpasses existing zero-shot CoT, PS+, RE2, and APE prompting methods across various reasoning datasets, demonstrating notable performance, especially in arithmetic and symbolic reasoning. Extensive experiments and analyses validate the effectiveness of zero-shot EoT prompting, showcasing its potential to enhance LLMs' reasoning capabilities. We believe there is considerable potential for refining the application of swarm intelligence algorithms based on LLMs to enhance model reasoning capabilities.

Limitations

In our proposed method, we have integrated core elements of swarm intelligence algorithms to leverage the capabilities of large language models for chain-of-thought reasoning. Notably, specific swarm intelligence algorithms, such as differential evolution, still need to be explored in our current experimentation and could be deferred for investigation in future endeavors. Our preliminary experiments are exclusively conducted using GPT-3.5-Turbo, Llama-3-8B-Instruct, and GPT-4. Considering the substantial costs associated with API usage, we intend to broaden the validation of our proposed method across a more extensive range of large language models in subsequent stages, aiming to enhance the generalizability and robustness of our method, ensuring its applicability across various language models and further validating its efficacy. Moreover, we do not evaluate our proposed EoT prompting under the few-shot setting because of the substantial costs associated with API usage. We leave this for further exploration in the future.

References

Anonymous. 2024a. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*.

Anonymous. 2024b. Large language models to enhance bayesian optimization. In *The Twelfth International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Angelica Chen, David Dohan, and David R. So. 2023. Evoprompting: Language models for code-level neural architecture search. *ArXiv*, abs/2302.14838.

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

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, Yujiu Yang, Tsinghua University, and Microsoft Research. 2023. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *ArXiv*, abs/2309.08532.

Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation*, 11:1–18.

Noah Hollmann, Samuel Müller, and Frank Hutter. 2023. Large language models for automated data science: Introducing CAAFE for context-aware automated feature engineering. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2023. Instance-aware prompt learning for language understanding and generation. *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, 22(7):199:1–199:18.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *EMNLP*.

Junzhi Li and Ying Tan. 2018. Loser-out tournament-based fireworks algorithm for multimodal function optimization. *IEEE Transactions on Evolutionary Computation*, 22(5):679–691.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Shengcai Liu, Caishun Chen, Xinghua Qu, Ke Tang, and Yew Soon Ong. 2023. Large language models as evolutionary optimizers. *ArXiv*, abs/2310.19046.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *ArXiv*, abs/2103.10385.

Elliot Meyerson, M. Nelson, Herbie Bradley, Arash Moradi, Amy K. Hoover, and Joel Lehman. 2023. Language model crossover: Variation through few-shot prompting. *ArXiv*, abs/2302.12170.

Melanie Mitchell. 1998. *An introduction to genetic algorithms*. MIT Press.

Jean-Baptiste Mouret. 2024. Large language models help computer programs to evolve. *Nature*, 625 7995:452–453.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue. In *OpenAI Blog*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco J R Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, Alhussein Fawzi, Josh Grochow, Andrea Lodi, Jean-Baptiste Mouret, Talia Ringer, and Tao Yu. 2023. Mathematical discoveries from program search with large language models. *Nature*, 625:468 – 475.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? In *Thirty-seventh Conference on Neural Information Processing Systems*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, V. G. Vinod Vydiswaran, and Hao Ma. 2022. IDPG: an instance-dependent prompt generation method. *CoRR*, abs/2204.04497.

Xiaohan Xu, Chongyang Tao, Tao Shen, Can Xu, Hongbo Xu, Guodong Long, and Jian-Guang Lou. 2023. Re-reading improves reasoning in language models. *ArXiv*, abs/2309.06275.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

A Details of Experimental Setup

A.1 Datasets

Table 7 shows the statistics of datasets used in our experiment.

Table 7: Details of datasets evaluated in our experiment.

No.	Dataset	Samples	Avg Words	Answer Format	Domain
1	SingleEq	508	27.4	Number	Math
2	AddSub	395	31.5	Number	Math
3	GSM8K	1319	46.9	Number	Math
4	MultiArith	600	31.8	Number	Math
5	SVAMP	1000	31.8	Number	Math
6	AQuA	254	51.9	Option	Math
7	CommonsenseQA	1221	27.8	Option	Commonsense
8	StrategyQA	2290	9.6	Yes/No	Commonsense
9	Coin Flip	500	37.0	Yes/No	Symbolic
10	Last Letters	1000	15.0	String	Symbolic

A.2 Answer Extraction Prompts

Table 8 shows a list of answer extraction prompts used in our experiments.

Table 8: Answer extraction prompts used in our experiments across all baselines.

No	Dataset	Answer Extraction Prompts
1	SingleEq	Therefore, the answer (arabic numerals) is
2	AddSub	Therefore, the answer (arabic numerals) is
3	GSM8K	Therefore, the answer (arabic numerals) is
4	MultiArith	Therefore, the answer (arabic numerals) is
5	SVAMP	Therefore, the answer (arabic numerals) is
6	AQuA	Therefore, among A through E, the answer is
7	CommonsenseQA	Therefore, among A through E, the answer is
8	StrategyQA	Therefore, the answer (Yes or No) is
9	Coin Flip	Therefore, the answer (Yes or No) is
10	Last Letters	Therefore, the answer is

B Additional Analysis

B.1 Effect of Number of Initializing CoT Promptings

Our method can initialize multiple CoT promptings rather than limit them to just two. By initializing multiple CoT promptings and performing a single round of crossover and mutation, it is possible to generate more promptings better suited to the current reasoning problem. Figure 4 shows the experimental results of our method after initializing multiple CoT promptings and applying crossover and mutation with GPT-3.5-Turbo. As the number of initial promptings increases, the performance of the LLM in AQuA and AddSub improves. Considering the escalating temporal demand and token count limitations with increasing CoT promptings, we set the number of initializing CoT promptings to two.

B.2 Effect of Initialization Prompts

To assess the impact of varied initializations of CoT prompting on the ensuring quality of generated EoT prompting, we conduct a series of experiments to investigate the influence of EoT prompting

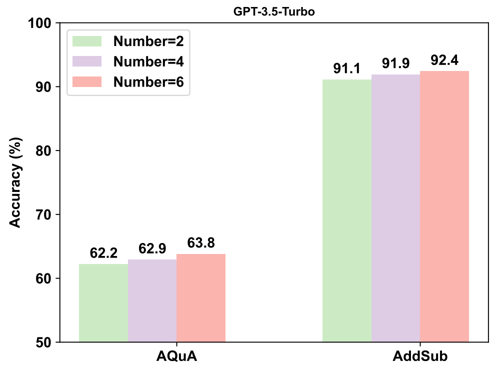

Figure 4: Performance of the number of initializing CoT Promptings on GPT-3.5-Turbo

instructions systematically. As illustrated in Table 9, the experiments encompass four arithmetic reasoning datasets: AddSub, SVAMP, AQuA, and GSM8K. P1 designates the prompt employed by the zero-shot CoT prompting method, while P2, P3, and P4 signify the prompts integral to our proposed method. Notably, it is observed that the EoT prompting instruction utilized in P4 exhibits superior performance, surpassing the previously employed P3 in antecedent experiments. This observation underscores the potential for leveraging swarm intelligence algorithms to generate CoT promptings for each instance, thereby warranting further exploration. We show the example outputs of different reasoning tasks in Appendix D.

Table 9: Performance comparison of trigger sentences measured on four math reasoning datasets with GPT-3.5-Turbo.

No.	Trigger Sentence	AddSub	SVAMP	AQuA	GSM8K
P1	Let's think step by step.	86.6	79.0	55.1	75.3
P2	Below are the two Prompts: Prompt 1: Let's think step by step. Prompt 2: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and commonsense) and solve the problem step by step carefully. I'd like you to follow the instruction step-by-step to generate a new prompt: 1. Crossover the two prompts and generate a new prompt. 2. Mutate the prompt generated in Step 1 and generate a new prompt. 3. Select a prompt that is suitable for this problem and solve the problem.	90.4	80.5	59.8	76.9
P3	I'd like you to first crossover and mutate the following two prompts to generate a new prompt: Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and commonsense) and solve the problem step by step carefully. Prompt 2: Let's first understand the problem and solve the problem step by step. Finally, select a prompt that is suitable for this problem to rewrite and solve the problem.	91.1	82.0	62.2	78.5
P4	I'd like you to follow the instructions step-by-step to solve the problem step by step, and show the answer. 1. Crossover the following prompts and generate a new prompt: Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully. Prompt 2: Let's first understand the problem and solve the problem step by step. 2. Mutate the crossover prompt in Step 1 to generate the final prompt. 3. Apply a prompt that is suitable for this problem to rewrite and solve the problem.	**91.4**	**82.5**	**63.4**	**79.1**

C Additional Related Work

LLMs and Prompting With the increasing model complexity and the scale of parameters, LLMs have unlocked emerging capabilities, notably in-context learning (ICL) (Brown et al., 2020). The ICL strategy directly incorporates demonstrations into manually crafted prompts, enabling LLMs to perform exceptionally well without requiring task-specific fine-tuning. Recently, researchers have proposed continuous prompt tuning (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021) to overcome challenges in discrete prompt searching. For instance, Wu et al. (2022) and Jin et al. (2023) seek suitable prompts for each instance by learning continuous prompt information relevant to the instance. However, these methods require fine-tuning the parameters of the entire model, which are not friendly for LLMs. In contrast, our EoT prompting seeks suitable prompt information for each instance in a discrete space, avoiding fine-tuning the parameters of the entire model while maintaining good interpretability and robustness.

LLMs and Optimization Algorithms Recent research has seen a flourishing exploration of treating LLMs as optimizers (Anonymous, 2024a; Liu et al., 2023; Meyerson et al., 2023). Some recent endeavors, leveraging the formidable capabilities of LLMs, have demonstrated impressive performance in tasks such as neural network search (Chen et al., 2023), mathematical problem-solving (Romera-Paredes et al., 2023), and various other domains by integrating LLMs with swarm intelligence algorithms (Guo et al., 2023; Mouret, 2024; Anonymous, 2024b; Hollmann et al., 2023). In our work, we pioneer the application of considering LLMs as part of swarm intelligence algorithms, specifically applying our EoT prompting to CoT reasoning, yielding favorable results across diverse tasks.

D Example Outputs of Zero-shot EoT

Tables 10 to 13 list example outputs generated by our proposed method Zero-shot EoT prompting for several datasets, including SVAMP, AQuA, AddSub, and MultiArith.

Table 10: Example output of Zero-shot-EoT for SVAMP

Q: Jack received 10 emails in the morning, 5 emails in the afternoon and 4 emails in the evening. How many more emails did Jack receive in the afternoon than in the evening?
A: I'd like you to follow the instructions step-by-step to solve the problem step by step, and show the answer.
1. Crossover the following prompts and generate a new prompt:
Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully.
Prompt 2: Let's first understand the problem and solve the problem step by step.
2. Mutate the crossover prompt in Step 1 to generate the final prompt.
3. Apply a prompt that is suitable for this problem to rewrite and solve the problem.
Output:
Step 1: Crossover the prompts to generate a new prompt. Crossover Prompt: "Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, let's solve the problem step by step."
Step 2: Mutate the crossover prompt. Mutated Prompt: "First, let's carefully examine the problem and extract all relevant variables along with their corresponding numbers. Then, we'll calculate intermediate values (ensuring accuracy and using common sense) and solve the problem methodically."
Step 3: Apply the mutated prompt to rewrite and solve the problem.
Given: Morning emails = 10, Afternoon emails = 5, Evening emails = 4. To find: How many more emails did Jack receive in the afternoon than in the evening?
Solution: Extract the relevant variables and their corresponding numbers:
Morning emails = 10, Afternoon emails = 5, Evening emails = 4.
Calculate the difference between afternoon and evening emails:
Difference = Afternoon emails - Evening emails
Difference = 5 - 4
Difference = 1
So, Jack received 1 more email in the afternoon than in the evening.

Table 11: Example output of Zero-shot-EoT for AQuA

Q: A rope 20 meters long is cut into two pieces. If the length of one piece of rope is 3 meters shorter than the length of the other, what is the length, in meters, of the longer piece of rope? Answer Choices: (A) 7.5 (B) 8.9 (C) 9.9 (D) 11.5 (E) 11.7.
A: I'd like you to follow the instructions step-by-step to solve the problem step by step, and show the answer.
1. Crossover the following prompts and generate a new prompt:
Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully.
Prompt 2: Let's first understand the problem and solve the problem step by step.
2. Mutate the crossover prompt in Step 1 to generate the final prompt.
3. Apply a prompt that is suitable for this problem to rewrite and solve the problem.
Output:
Crossover Prompt: "Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, let's solve the problem step by step."
Mutated Prompt: "First, let's thoroughly grasp the problem and identify all pertinent variables along with their respective values. Next, we'll systematically work through the problem, ensuring accuracy at each step, until we find the solution."
Selected Prompt: "First, let's thoroughly grasp the problem and identify all pertinent variables along with their respective values. Next, we'll systematically work through the problem, ensuring accuracy at each step, until we find the solution."
Now, let's solve the problem using this prompt:
Given:
Total length of rope = 20 meters
Length of one piece of rope = ?
Length of the other piece of rope = ?
To find: What is the length, in meters, of the longer piece of rope?
Solution: Extract the relevant variables and their corresponding numbers: Total length of rope = 20 meters
Set up the equation based on the problem statement: Let x be the length of the longer piece of rope. Then, the length of the shorter piece of rope is x-3 meters.
The sum of the lengths of the two pieces of rope is equal to the total length of the rope:x + (x-3)=20
Solve the equation for
2x-3=20
2x=23
x=11.5
So, the length of the longer piece of rope is 11.5 meters.
Answer: (D) 11.5

Table 12: Example output of Zero-shot-EoT for AddSub

Q: There are 54 scissors in the drawer. Keith placed 22 scissors in the drawer. How many scissors are now there in all?
A: I'd like you to follow the instructions step-by-step to solve the problem step by step, and show the answer.
1. Crossover the following prompts and generate a new prompt:
Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully.
Prompt 2: Let's first understand the problem and solve the problem step by step.
2. Mutate the crossover prompt in Step 1 to generate the final prompt.
3. Apply a prompt that is suitable for this problem to rewrite and solve the problem.
Output:
Crossover Prompt: "Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and determine the solution step by step."
Mutated Prompt: "First, let's thoroughly comprehend the problem and identify all relevant variables along with their respective values. Then, we'll proceed to calculate intermediate values (ensuring accuracy and using common sense) and arrive at the solution methodically."
Selected Prompt: "Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and determine the solution step by step."
Now, let's solve the problem using this prompt:
Given:
Total number of scissors in the drawer = 54
Scissors placed by Keith = 22
Solution:
Extract the relevant variables and their corresponding numbers:
Total number of scissors in the drawer = 54
Number of scissors placed by Keith = 22
Calculate the number of scissors now:
Number of scissors now = Total number of scissors + Number of scissors placed by Keith
Number of scissors now = 54 + 22
Number of scissors now = 76
So, there are now 76 scissors in all.
Answer: 76

Table 13: Example output of Zero-shot-EoT for MultiArith

Q: A store had 27 coloring books in stock. They ended up putting them on sale and getting rid of 6 of them. The put the ones they still had onto shelves with 7 on each shelf. How many shelves did they use?
A: I'd like you to follow the instructions step-by-step to solve the problem step by step and show the answer.
1. Crossover the following prompts and generate a new prompt:
Prompt 1: Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, Let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully.
Prompt 2: Let's first understand the problem and solve the problem step by step.
2. Mutate the crossover prompt in Step 1 to generate the final prompt.
3. Apply the final prompt in Step 2 to think, rewrite, solve the problem step by step, and show the answer.
Output:
Crossover Prompt: "Let's understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, let's calculate intermediate variables and solve the problem step by step."
Mutated Prompt: "First, let's thoroughly understand the problem and identify all relevant variables along with their respective values. Then, we'll proceed to calculate intermediate values (ensuring accuracy and using common sense) and unravel the solution methodically."
Selected Prompt: "Let's first understand the problem and carefully extract all relevant variables and their corresponding numerals. Then, let's calculate intermediate variables (pay attention to correct numerical calculation and common sense) and solve the problem step by step carefully."
Now, let's solve the problem using this prompt:
Given:
Total number of coloring books in stock = 27
Number of coloring books sold = 6
Number of coloring books left = Total number - Number sold = 27 - 6 = 21
Number of coloring books placed on each shelf = 7
To find: How many shelves did they use?
Solution:
Extract the relevant variables and their corresponding numbers:
Total number of coloring books left = 21
Number of coloring books placed on each shelf = 7
Calculate the number of shelves used:
Number of shelves = Total number of coloring books left / Number of coloring books placed on each shelf
Number of shelves = 21 / 7
Number of shelves = 3
So, they used 3 shelves.
Answer: 3