Re-Reading Improves Reasoning in Large Language Models

Xiaohan Xu[1], **Chongyang Tao**[2*], **Tao Shen**[3], **Can Xu**[4],
Hongbo Xu[1], **Guodong Long**[3], **Jian-guang Lou**[4], **Shuai Ma**[2]
[1]Institute of Information Engineering, CAS [2]SKLSDE Lab, Beihang University
[3]University of Technology Sydney [4]Microsoft Corporation
shawnxxh@gmail.com, {chongyang, mashuai}@buaa.edu.cn, hbxu@iie.ac.cn
{caxu,jlou}@microsoft.com, {tao.shen,guodong.long}@uts.edu.au

Abstract

To enhance the reasoning capabilities of off-the-shelf Large Language Models (LLMs), we introduce a simple, yet general and effective prompting method, RE2, i.e., **Re-Re**ading the question as input. Unlike most thought-eliciting prompting methods, such as Chain-of-Thought (CoT), which aim to elicit the reasoning process in the output, RE2 shifts the focus to the input by processing questions twice, thereby enhancing the understanding process. Consequently, RE2 demonstrates strong generality and compatibility with most thought-eliciting prompting methods, including CoT. Crucially, RE2 facilitates a "bidirectional" encoding in unidirectional decoder-only LLMs because the first pass could provide global information for the second pass. We begin with a preliminary empirical study as the foundation of RE2, illustrating its potential to enable "bidirectional" attention mechanisms. We then evaluate RE2 on extensive reasoning benchmarks across 14 datasets, spanning 112 experiments, to validate its effectiveness and generality. Our findings indicate that, with the exception of a few scenarios on vanilla ChatGPT, RE2 consistently enhances the reasoning performance of LLMs through a simple re-reading strategy. Further analyses reveal RE2's adaptability, showing how it can be effectively integrated with different LLMs, thought-eliciting prompting, and ensemble strategies.[1]

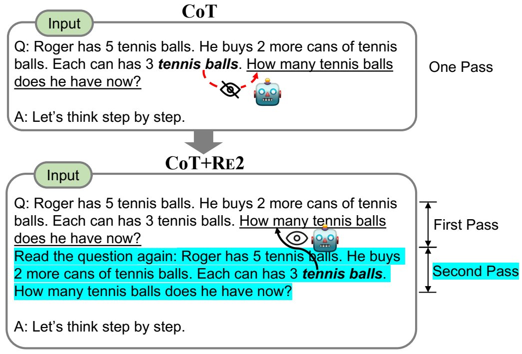

Figure 1: Example inputs of CoT prompting versus CoT prompting with RE2. In original CoT, every token in the question cannot see its later tokens since most LLMs are autoregressive models (**the top figure**). RE2 is a simple prompting method that repeats the question as input. LLMs with RE2 allows each token in the second pass, e.g. *"tennis balls"*, to see its later tokens from the first pass, e.g. *"How many ..."*, achieving an effect of a "bidirectional" understanding (**the bottom figure**).

1 Introduction

In the ever-evolving landscape of artificial intelligence, large language models (LLMs) have emerged as a keystone of natural language understanding and generation (Brown et al., 2020; Touvron et al., 2023a,a; OpenAI, 2023; Xu et al., 2024). As LLMs have become more advanced, a key challenge has emerged: teaching them to reason effectively. The ability to reason well is a key aspect of human intelligence, allowing us to infer, deduce, and solve problems. In LLMs, this skill is crucial for improving their practical use. Despite their impressive abilities, LLMs often have difficulty with reasoning tasks (Blair-Stanek et al., 2023; Arkoudas, 2023), urging researchers to explore more strategies to bolster reasoning ability (Wei et al., 2022b; Gao et al., 2023; Besta et al., 2023).

Existing research on reasoning has predominantly concentrated on designing diverse thought-eliciting prompting strategies to elicit reasoning processes in the output phase, such as Chain-of-Thought (CoT) (Wei et al., 2022b), Program-Aided Language Model (PAL) (Gao et al., 2023), etc. (Yao et al., 2023a; Besta et al., 2023; Wang et al., 2023a). In contrast, scant attention has been paid to the understanding of the input phase. In fact, comprehension is the first step before solving the problem, which is crucially important. However, in the era of generative AI, most LLMs adopt the decoder-only LLMs with unidirectional attention,

[*]Corresponding author
[1]Our code is available at Github.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 15549–15575
November 12-16, 2024

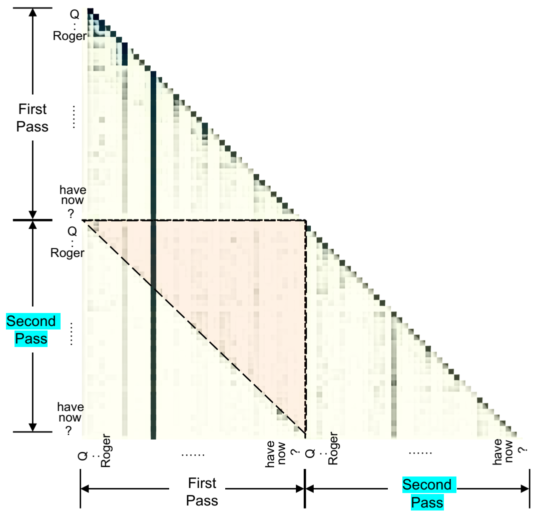

Figure 2: Illustration of the attention distribution in LLaMA-2 by repeating the question as the input (a darker cell indicates higher attention). The region within the red dashed upper triangle demonstrates that every token in the second pass has obvious attention to its later tokens in the first pass. This suggests that re-reading in LLMs is promising for achieving a "bidirectional" understanding of the question.

like GPT-3 (Brown et al., 2020) and LLaMA (Touvron et al., 2023b). This unidirectional attention limits every token's visibility to only previous tokens when encoding a question, potentially impairing the bidirectional understanding of each word in the question (Du et al., 2022). In Figure 1, the last sentence, *"How many ..."*, highlights the question's main focus, which is crucial for the understanding of the preceding words. However, LLMs cannot see the subsequent words when encoding a token due to their unidirectional vision.

Fortunately, many cognitive science studies have revealed that humans tend to re-read questions during learning and problem-solving to enhance the comprehension process (Dowhower, 1987, 1989; Ozek and Civelek, 2006). The first reading provides an overall understanding, which benefits the second reading. Motivated by this, we also conduct a preliminary empirical study for LLaMA-2 (Touvron et al., 2023b) by repeating the question two times as the input using the GSM8K dataset (Cobbe et al., 2021). The attention heatmap in Figure 2 shows that the re-reading strategy allows LLaMA-2 to achieve a "bidirectional" understanding of the question, which is expected to further improve the reasoning performance.

Based on the observation and inspired by the human strategy of re-reading, we present a simple yet effective and general reasoning prompting strategy, RE2, i.e., **Re-Re**ading the question as input (see the illustration in Figure 1). While our RE2 is simple, it offers several advantages for LLMs' reasoning scenarios. (1) This approach mirrors the human strategy of problem-solving. LLMs with RE2 show potential for a "bidirectional" understanding of questions. (2) Repeating questions allows LLMs to allocate more computational resources to input encoding, similar to "horizontally" increasing the depth of neural networks. (3) RE2 emphasizes understanding during the input phase, making it orthogonal to and compatible with most thought-eliciting prompting methods that focus on the output phase, such as CoT and PAL.

To validate the efficacy and generality of RE2, we conducted extensive experiments spanning arithmetic, commonsense, and symbolic reasoning tasks across 14 datasets and 112 experiments. The results show that, with the exception of certain scenarios on vanilla ChatGPT, our RE2 with a simple re-reading strategy consistently enhances the reasoning performance of LLMs. RE2 exhibits versatility across various LLMs, such as Text-Davinci-003, ChatGPT, LLaMA-2-13B, and LLaMA-2-70B, spanning both instruction finetuning (IFT) and non-IFT models. We also explore RE2 in task settings of zero-shot and few-shot, thought-eliciting prompting methods, and the self-consistency setting, highlighting its generality.

2 Methodology

2.1 Vanilla Chain-of-Thought for Reasoning

We begin with a unified formulation to leverage LLMs with CoT prompting to solve reasoning tasks. In formal, given an input x and a target y, a LLM p with CoT prompting can be formulated as

$$y \sim \sum_{z \sim \, p(\mathrm{z}|C_x)} p(\mathrm{y}|C_x, z) \cdot p(z|C_x),$$
$$\text{where } C_x = \mathrm{c}^{(\mathrm{cot})}(x). \tag{1}$$

In this formulation, C_x denotes the prompted input. $\mathrm{c}^{(\mathrm{cot})}(\cdot)$ represents the template with CoT prompting instructions, such as '*let's think step by step*'. z stands for a latent variable of rationale, and z denotes a sampled rationale in natural language. Consequently, the LLMs can break down complex tasks into more manageable reasoning steps, treating each step as a component of the overall solution chain. We employ CoT as a baseline to solve reasoning tasks without compromising its generality. In addition to CoT, our proposed simple RE2 can serve as a "plug & play" module adaptable to most other prompting methods (§2.3).

2.2 Re-Reading (RE2) Improves Reasoning

Drawing inspiration from the human strategy of re-reading, we introduce this strategy for LLM reasoning, dubbed RE2, to enhance understanding in the input phase. With RE2, the prompting process in Eq. 1 can be readily rephrased as:

$$y \sim \sum_{z \sim \, p(z|C_x)} p(\mathrm{y}|C_x, z) \cdot p(z|C_x),$$
$$\text{where } C_x = \mathrm{c}^{(\mathrm{cot})}(\mathrm{re2}(x)). \quad (2)$$

In this formulation, $\mathrm{re2}(\cdot)$ is the re-reading operation of the input. We don't seek complex adjustments for LLMs but aim for a general implementation of $\mathrm{re2}(x)$ that is as simple as follows:

RE2 Prompting

```
Q: {Input Query}
Read the question again: {Input Query}
# Thought-eliciting prompt (e.g.,“Let’s
think step by step") #
```

where '{Input Query}' is a placeholder for the input query, x. The left part of this prompting could incorporate other thought-eliciting prompts. Intuitively, RE2 offers two advantages to enhance the understanding process: (1) it allocates more computational resources to the input, and (2) it facilitates a "bidirectional" understanding of the question, where the first pass provides global information for the second pass.

2.3 Generality of RE2

Due to RE2's simplicity and emphasis on the input phase, it can be seamlessly integrated with a wide range of LLMs and algorithms, including few-shot settings, self-consistency, various thought-eliciting prompting strategies, and more. We offer insights into the integration of RE2 with other thought-eliciting prompting strategies as an illustration.

Compared with those thought-eliciting prompting strategies that focus on the output phase, RE2 shifts the emphasis towards understanding the input. Therefore, RE2 exhibits significant compatibility with them, acting as a “plug & play” module. This synergy has the potential to further enhance the reasoning abilities of LLMs. With a specific thought-eliciting prompting, τ, designed to elicit thoughts from the LLMs, Eq. (3) is rewritten as:

$$y \sim \sum_{z \sim \, p(z|C_x)} p(\mathrm{y}|C_x, z) \cdot p(z|C_x),$$
$$\text{where } C_x = \mathrm{c}^{(\tau)}(\mathrm{re2}(x)). \quad (3)$$

Here, τ denotes various thought-eliciting promptings beyond CoT, such as Plan-and-Solve (Wang et al., 2023a), and Program-Aided Prompt (Gao et al., 2023), etc. We also conducted lots of experiments to validate the generality of RE2 in §3.4.

3 Experiments

3.1 Benchmarks

We assess RE2 prompting across three key categories of reasoning benchmarks. Details of all datasets are shown in Appendix A

Arithmetic Reasoning We consider the following seven arithmetic reasoning benchmarks: the GSM8K benchmark of math word problems (Cobbe et al., 2021), the SVAMP dataset of math word problems with varying structures (Patel et al., 2021), the ASDiv dataset of diverse math word problems (Miao et al., 2020), the AQuA dataset of algebraic word problems (Ling et al., 2017), the AddSub (Hosseini et al., 2014) of math word problems on addition and subtraction for third to fifth grader, MultiArith (Roy and Roth, 2015) dataset of math problems with multiple steps, and the SingelEQ (Roy et al., 2015) dataset of elementary math word problems with single operation.

Commonsense and Symbolic Reasoning For commonsense reasoning, we use CSQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021), and the ARC (Clark et al., 2018). CSQA dataset consists of questions that necessitate various commonsense knowledge. The StrategyQA dataset comprises questions that demand multi-step reasoning. The ARC dataset (denoted as ARC-t) is divided into two sets: a Challenge Set (denoted as ARC-c), containing questions that both retrieval-based and word co-occurrence algorithms answered incorrectly, and an Easy Set (denoted as ARC-e). We evaluate two symbolic reasoning tasks: date understanding (Suzgun et al., 2023a) and Coinflip (Wei et al., 2022b). Date understanding is a subset of BigBench datasets (Suzgun et al., 2023a), which have posed challenges for previous fine-tuning efforts. Coinflip is a dataset of questions on whether

LLMs	Methods	GSM	SVAMP	ASDIV	AQUA	MultiArith	SingleEQ	AddSub
davinci-003	Vanilla	19.48	67.60	69.00	28.74	31.33	86.22	89.87
	Vanilla+RE2	24.79 ↑ 5.31	70.90 ↑ 3.30	71.20 ↑ 2.20	30.31 ↑ 1.57	42.33 ↑ 11.00	87.20 ↑ 0.98	92.15 ↑ 2.28
	CoT	58.98	78.30	77.60	40.55	89.33	92.32	91.39
	CoT+RE2	61.64 ↑ 2.68	81.00 ↑ 2.70	78.60 ↑ 1.00	44.49 ↑ 3.94	93.33 ↑ 4.00	93.31 ↑ 0.99	91.65 ↑ 0.26
ChatGPT	Vanilla	77.79	81.50*	87.00*	63.78*	97.83*	95.28*	92.41*
	Vanilla+RE2	79.45 ↑ 1.66	84.20 ↑ 2.70	88.40 ↑ 1.40	59.45 ↓ 4.33	96.67 ↓ 1.16	94.49 ↓ 0.79	91.65 ↓ 0.76
	CoT	78.77	78.70	85.60	55.51	95.50	93.70	88.61
	CoT+RE2	80.59 ↑ 1.82	80.00 ↑ 1.30	86.00 ↑ 0.40	59.45 ↑ 3.94	96.50 ↑ 1.00	95.28 ↑ 1.58	89.87 ↑ 1.26

Table 1: Results on arithmetic reasoning benchmarks. * denotes that Vanilla is even superior to CoT prompting.

LLMs	Methods	Commonsense					Symbolic	
		CSQA	StrategyQA	ARC-e	ARC-c	ARC-t	Date	Coin
davinci-003	Vanilla	74.20*	59.74	84.81	72.01	80.58	40.92	49.80
	Vanilla+RE2	76.99 ↑ 2.79	59.91 ↑ 0.17	88.22 ↑ 3.41	75.68 ↑ 3.67	84.07 ↑ 3.49	42.01 ↑ 1.09	52.40 ↑ 2.60
	CoT	71.66	67.55	85.69	73.21	81.57	46.07	95.60
	CoT+RE2	73.05 ↑ 1.39	66.24 ↓ 1.31	87.84 ↑ 2.15	76.02 ↑ 2.81	83.94 ↑ 2.37	52.57 ↑ 6.50	99.60 ↑ 4.00
ChatGPT	Vanilla	76.66*	62.36	94.32*	85.41*	91.37*	47.43*	52.00
	Vanilla+RE2	78.38 ↑ 1.72	66.99 ↑ 4.63	93.81 ↓ 0.51	83.19 ↓ 2.22	90.30 ↓ 1.07	47.97 ↑ 0.54	57.20 ↑ 5.20
	CoT	69.94	67.82	93.35	83.53	90.11	43.63	88.80
	CoT+RE2	71.66 ↑ 1.72	69.34 ↑ 1.52	93.14 ↓ 0.21	84.47 ↑ 0.94	90.27 ↑ 0.16	47.15 ↑ 3.52	95.20 ↑ 6.40

Table 2: Results on commonsense and symbolic reasoning benchmarks. * denotes that Vanilla is even superior to CoT prompting.

a coin is still heads up after it is flipped or not based on steps given in the questions.

3.2 Language Models and Implementations

Baseline Prompting. In our implementation, we rigorously evaluate the performance of our RE2 model on two baseline prompting methods: Vanilla and CoT. The Vanilla approach aligns with the standard prompting method outlined in (Wei et al., 2022b; Kojima et al., 2022), wherein no specific prompts are employed to elicit thoughts from LLMs. Conversely, the CoT method guides the model through a step-by-step thought process.

RE2 Prompting. We incorporate RE2 into these baseline methods to assess its impact, denoted as Vanilla+RE2 and CoT+RE2. To avoid the impact of randomness introduced by the demonstrations in a few-shot setting, we mainly assess our method in a zero-shot setting, following (Chen et al., 2023; Wang et al., 2023a; Du et al., 2023). Additionally, for different tasks, we design answer-format instructions in prompts to regulate the format of the final answer, facilitating precise answer extraction. Detailed information regarding the baseline prompting, RE2 prompting, and answer-format instructions can be found in the Appendix B.

Implementations. Our decoding strategy uses greedy decoding with a temperature setting of 0, thus leading to deterministic outputs. For these experiments, we employ two powerful backbones: ChatGPT (gpt-3.5-turbo-0613) (OpenAI, 2022) and davinci-003 (text-davinci-003)[2] , across all prompting methods, including Vanilla, CoT, Vanilla+RE2, and CoT+RE2. We also test RE2 on more advanced GPT-4o-mini in Appendix C.

3.3 Evaluation Results

Table 1 presents the results on arithmetic reasoning datasets, and Table 2 on commonsense reasoning and symbolic reasoning. In almost all scenarios, LLMs with RE2 achieve consistent improvements across both LLMs (davinci-003 and ChatGPT) and prompting methods (Vanilla and CoT). Specifically, davinci-003 with Vanilla+RE2 shows average improvements of 3.81, 2.51, and 1.85 in arithmetic, commonsense, and symbolic tasks, respectively. With CoT, davinci-003 generates intermediate reasoning steps, significantly enhancing the reasoning performance of LLMs. By applying RE2, davinci-003 with CoT+RE2 demonstrates further improvement, with average gains of 2.22, 1.23, and 5.25 in the same categories, respectively. These results indicate that RE2 can benefit LLMs in directly generating answers and improve the performance of CoT leading to correct answers.

[2]https://platform.openai.com/docs/models/gpt-3-5

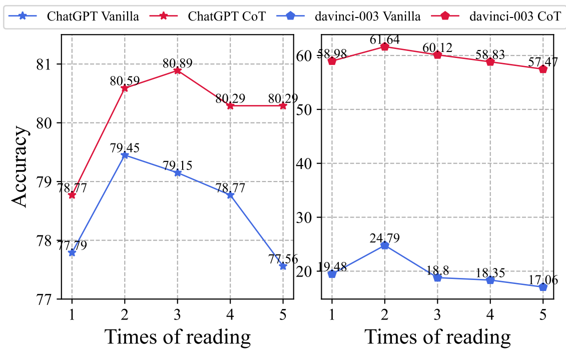

Figure 3: Evaluation results of the times of reading on GSM benchmark.

When applied to ChatGPT, RE2 exhibits consistent improvement on most datasets, except for a slight drop in performance on a few datasets, e.g., AQUA and MultiArith, when using Vanilla+RE2. This exception could be due to ChatGPT's exposure to these datasets with CoT outputs during instruction fine-tuning (IFT) (Chen et al., 2023). On such datasets, ChatGPT with Vanilla still produces CoT-like output (see examples in Appendix G) and even outperforms ChatGPT with CoT (as indicated by the * results in Tables 1 and 2). Chen et al. (2023) obtained similar experimental results and suggested that this occurs because ChatGPT may have been exposed to these task datasets containing CoT explanations without explicit prompting. Therefore, additional explicit instructions, like CoT or RE2, might disrupt this learned pattern in ChatGPT, possibly leading to decreased performance. Nonetheless, on some datasets like SVAMP, ASDIV, CSQA, and Date, RE2 still manages to improve the baseline Vanilla prompting. Moreover, in datasets where CoT prompting normally surpasses Vanilla prompting, such as GSM, StrategyQA, and Coin, RE2 significantly enhances Vanilla prompting (↑ 4.63 on StrategyQA and ↑ 5.20 on the Coin dataset). Overall, our RE2 method still achieves improvements in 71% of the experiments on ChatGPT. More examples from the experiment results can be found in Appendix G.

3.4 Discussions

Times of Question Reading We delve deeper into the impact of the times of question re-reading on reasoning performance. Figure 3 illustrates how the performance of two distinct LLMs evolves concerning various times of question re-reading. An overarching pattern emerges across all models: performance improves until the number of re-reads reaches 2 or 3, after which it begins to decline with further increases in question re-reading times. The potential reasons for inferior performance when reading the question multiple times are two-fold: i) appropriate reading times increase LLMs' ability to generate correct answers. However, excessively repeating questions may serve as demonstrations, causing the LLMs to repeat the questions themselves (see Appendix E for detailed analysis). and ii) repeating the question significantly increase the inconsistency of the LLMs between our inference and pretraining/alignment (intuitively in the learning corpora, we usually repeat a question twice). It's noteworthy that reading the question twice is optimal in most scenarios, which is why we refer to it as "re-reading" in our paper.

LLMs	Methods	GSM
ChatGPT	PS	75.59
	PS+RE2	**76.27**
	PAL	75.59
	PAL + RE2	**79.38**
davinci-003	PS	55.65
	PS+RE2	**58.68**
	PAL	68.61
	PAL + RE2	**70.20**

Table 3: Evaluation results of some thought-eliciting promptings beyond CoT with RE2.

Compatibility with Thought-Eliciting Prompt Strategies Compared to previous methods attempting to elicit thoughts in the output from LLMs, our RE2 emphasizes the understanding of the input. Therefore, we are intrigued to explore whether RE2 is effective with various thought-eliciting prompting strategies other than CoT. To investigate this, we apply RE2 to two other recently introduced prompting methods, namely, Plan-and-Solve (PS) (Wang et al., 2023a) and Program-Aided Language models (PAL) (Gao et al., 2023). The former model devises a plan to divide the entire task into smaller subtasks, and then carries out the subtasks according to the plan, while the latter generates programs as the intermediate reasoning steps. We directly apply our RE2 to these two methods by making a simple alteration to the input by repeating the question. Table 3 presents the evaluation findings on the GSM benchmark. Our observations reveal a consistent trend, akin to what was observed with CoT prompting. These results suggest that the effectiveness of our RE2 generally extends across various prompting methodologies.

LLMs	Methods	GSM	SVAMP	ASDIV	AQUA	MultiArith	SingleEQ	AddSub
davinci-003	Vanilla	16.98	69.10	70.56	28.34	38.67	83.46	88.86
	Vanilla+RE2	19.02 ↑ 2.04	73.60 ↑ 4.50	73.23 ↑ 2.67	27.95 ↓ 0.39	46.00 ↑ 7.33	84.06 ↑ 0.60	89.37 ↑ 0.51
	CoT	56.63	78.90	79.96	46.45	96.16	90.94	88.60
	CoT+RE2	60.12 ↑ 3.49	79.80 ↑ 0.90	81.21 ↑ 1.25	44.89 ↓ 1.56	96.83 ↑ 0.67	91.14 ↑ 0.20	89.37 ↑ 0.77

Table 4: Evaluation results on arithmetic reasoning benchmarks under few-shot setting.

LLMs	Methods	GSM	SVAMP	ASDIV	AQUA	MultiArith	SingleEQ	AddSub
Llama-2-13B	Vanilla	5.76	43.90	52.91	22.44	6.33	68.11	66.58
	Vanilla+RE2	6.82 ↑ 1.06	47.90 ↑ 4.00	53.15 ↑ 0.24	17.32 ↓ 5.12	6.50 ↑ 0.17	69.68 ↑ 1.57	70.12 ↑ 3.54
	CoT	21.99	41.60	45.18	22.83	56.83	58.46	58.99
	CoT+RE2	22.37 ↑ 0.38	46.50 ↑ 4.90	48.81 ↑ 3.63	24.80 ↑ 1.97	55.83 ↓ 0.99	66.34 ↑ 7.88	60.76 ↑ 1.77
Llama-2-70B	Vanilla	11.60	56.60	61.31	20.08	24.67	77.17	80.25
	Vanilla+RE2	13.50 ↑ 1.90	63.60 ↑ 7.00	64.66 ↑ 3.35	22.05 ↑ 1.97	25.00 ↑ 0.33	80.31 ↑ 3.14	84.05 ↑ 3.80
	CoT	49.73	66.90	68.08	37.80	79.83	80.51	74.18
	CoT+RE2	56.71 ↑ 6.98	70.40 ↑ 3.50	70.42 ↑ 2.34	38.58 ↑ 0.78	88.83 ↑ 9.00	81.10 ↑ 0.59	69.37 ↓ 4.81

Table 5: Evaluation results of LLAMA-2 on arithmetic reasoning benchmarks

Compatibility with Few-Shot Prompting It is noteworthy that our proposed re-reading mechanism is compatible with few-shot prompting. To demonstrate this compatibility, we conducted experiments on arithmetic reasoning tasks using the davinci-003 model, employing both Vanilla and CoT prompting methods. The few-shot prompting strategy and exemplars used align with those presented in (Wei et al., 2022b). For both the Vanilla+RE2 and CoT+RE2 methods, we applied the re-reading mechanism to the exemplars as well. The results of these experiments are presented in Table 4. We can observe that the inclusion of the re-reading mechanism consistently enhances the performance of both prompting methods, mirroring our findings in the zero-shot setting.

Effect on Non-IFT Models In our primary experiments, we employed the ChatGPT and davinci-003 models, which had undergone IFT training. These models, being aligned with human-like behavior, are better equipped to follow instructions effectively. Additionally, they may have been exposed to datasets with CoT prompting during their training, making the "re-reading" mechanism potentially more beneficial in recalling explanations. To gauge the broader applicability of our approach and to eliminate any IFT-related impacts, we conducted experiments on non-IFT pretrained models: Llama-2-13B and Llama-2-70B (Touvron et al., 2023b). Llama-2 is an open-source model pretrained on publicly available data without IFT or RLHF fine-tuning. We evaluated Llama-2 on arithmetic reasoning tasks under a zero-shot setting, following (Kojima et al., 2022). The results are presented in Table 5. The results clearly indicate that the re-reading mechanism consistently enhances the performance of both Vanilla and CoT prompting methods across most tasks when applied to Llama-2 models. This observation underscores the generality of our approach and dispels concerns about potential data leakage from IFT during training. This also underscores the versatility of RE2, which can be effectively employed across various model scales and types, regardless of whether they have undergone IFT training or are non-IFT LLM.

LLMs	Methods	GSM	SVAMP
ChatGPT	Vanilla	77.79	81.50
	Vanilla+SC	85.60	87.37
	Vanilla+RE2 +SC	**86.35**†	**87.74**
	CoT	78.77	78.70
	CoT+SC	85.75	84.90
	CoT+RE2 +SC	**86.88**†	**87.70**†

Table 6: Evaluation results of re-reading with self-consistency (t-test, † denote p-value < 0.05).

Compatibility with Self-consistency Existing research indicates that the chain-of-thought prompting approach can be enhanced by adopting the self-consistency method, which involves aggregating the majority final answer from multiple sampled generations. We are also intrigued by the potential for further enhancing the proposed re-reading mechanism using this method. Consequently, we conduct experiments testing the integration of RE2 with the self-consistency approach on the GSM benchmark by using ChatGPT. The temperature is

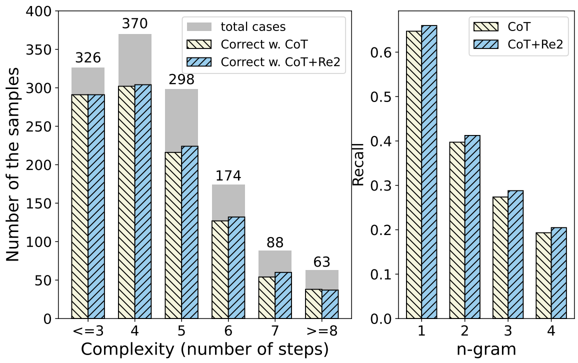

Figure 4: **Left figure**: model performance versus complexity of questions. X-axis means the complexity of questions and Y-axis refers to frequency. The gray hist means the number of total cases for each complexity. **Right figure**: n-gram recall between the generation and the input question. We take the question and generation as the reference and hypothesis respectively.

set to 0.7. We report the results averaged over 10 runs, where we sampled 10 outputs independently from the LLMs in each run. Table 6 demonstrate that self-consistency significantly enhances the performance of both prompting methods. Despite self-consistency's aggregation of multiple answers, our re-reading mechanism still contributes to improvement on most scenarios, indicating its compatibility with the self-consistency approach.

Performance across Different Question Complexity. We further investigate the impact of input question complexity on the reasoning performance of both CoT and CoT+RE2 promptings using ChatGPT on GSM8K dataset, as shown in the left part of Figure 4. In accordance with Fu et al. (2022), we measure question complexity by counting the reasoning steps present in the ground-truth explanations. Our findings reveal that the performance of all promptings generally diminishes as question complexity increases, suggesting that the current LLMs still struggle with handling intricate queries. Notably, the introduction of re-reading enhances performance on various complexities, including those slightly complex questions. This observation underscores the benefits of RE2 for improving reasoning capabilities over complex questions. To further validate the improved understanding ability, we calculate the coverage degree (n-gram recall) between the generations and the input questions, as illustrated in the right part of Figure 4. The results indicate that RE2 increases the n-gram (n=1,2,3,4) recall in the output explanations, underscoring how our method enhances the model's focus on the question during the reasoning process.

Prompt		Vanilla	CoT

P0	`Q: {question}`
	`#Answer format instruction#`
	`A: Let's think step by step.`
		77.79	78.77

P1	`Q: {question}`
	`Read the question again: {question}`
	`#Answer format instruction#`
	`A: Let's think step by step.`
		79.45	**80.59**

P2	`Q: {question}`
	`Q: {question}`
	`#Answer format instruction#`
	`A: Let's think step by step.`
		78.09	79.38

P3	`Q: {question}`
	`A: Let's think step by step.`
	`Read the question again: {question}`
	`#Answer format instruction#`
	`A: Let's think step by step.`
		79.08	80.36

P4	`Q: {question}`
	`A: Let's think step by step.`
	`Q: {question}`
	`#Answer format instruction#`
	`A: Let's think step by step.`
		78.09	79.38

Table 7: Results of different re-reading instructions.

The Impact of Different Re-Reading Instructions We further conduct experiments to examine the influence of RE2 within the context of CoT prompting. Specifically, we design various instructions for question re-reading using ChatGPT on GSM8K dataset. As depicted in P1 and P2 in Table 7, instruction P1, which includes the phrase "`Read the question again:`", exhibits superior performance compared to directly repeating the question twice. These results suggest that providing more detailed re-reading instructions to the LLMs is advantageous. Subsequently, we explore the possibility of introducing re-reading for CoT instructions (i.e. repeating "`Let's think step by step`"), as exemplified in P3 and P4. However, we observe that repeating the thinking process two times does not yield any discernible benefits. It's noteworthy that, in general, question re-reading consistently improves reasoning performance compared to the standard CoT prompting without question re-reading (P0).

Impact on Inference Efficiency and Memory Usage. Re2 doubles the question length in both zero-and few-shot settings, which may affect inference efficiency and memory usage. This section quantitatively explores that impact. We utilize Llama-2 7B with float16 precision and randomly sample 100 instances from the GSM8K dataset. We measure the average inference time and memory usage across four scenarios: Zero-shot, Zero-shot + CoT, Few-shot, and Few-shot + CoT. When applying Re2, the questions in the demonstrations

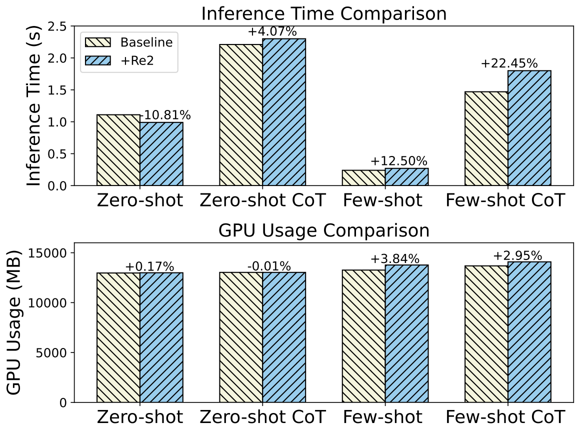

Figure 5: RE2's impact on inference efficiency and GPU memory usage.

are also repeated. All experiments are performed on 8×NVIDIA GeForce RTX 4090 GPUs, with results shown in Figure 5. The findings reveal that RE2 only marginally increases inference time and memory usage in both zero-shot and few-shot settings, even with longer inputs. This minimal impact is attributed to various optimization and inference acceleration techniques in current LLMs, such as grouped-query attention (Touvron et al., 2023b), CUDA, and GPU-based computations. For instance, grouped-query attention is particularly advantageous for long inputs, significantly accelerating decoder inference. Likewise, CUDA and GPU-based computations are highly optimized for parallel processing, especially for matrix multiplications in LLMs (NVIDIA, 2024).

4 Related Work

Reasoning with Large Language Models. LLMs represent a significant milestone in the journey towards artificial general intelligence (AGI) (OpenAI, 2023; Touvron et al., 2023b) . Reasoning ability is particularly crucial on the way towards AGI, where artificial intelligence needs to act or think like human beings (Qiao et al., 2023; Huang and Chang, 2023). In the literature on LLMs, performing reasoning tasks via interaction in natural language plays a significant role in evaluating an LLM, into which academia and industry have been dedicating many endeavors (Wei et al., 2022a; Suzgun et al., 2023b; Turpin et al., 2023). In principle, most works for reasoning with large language models could fall into the paradigm of "Chain-of-Thought" (Wei et al., 2022b; Kojima et al., 2022), which assists LLMs in fulfilling complex reasoning tasks by generating intermediate steps explicitly. Therefore, most of the endeavors are dedicated to improving the basic principle by the following aspects: i) the structure of "chain", e.g., tree (Yao et al., 2023a), graph (Yao et al., 2023b); ii) the modality of the chain, e.g., program (Gao et al., 2023); iii) the reliability of the chain, e.g., self-consistency (Wang et al., 2023c), faithful (Lyu et al., 2023), retrieval-based verifying (He et al., 2023); and iv) decomposition of the chain, e.g., least-to-most (Zhou et al., 2023), decomposed (Radhakrishnan et al., 2023), plan-to-solve (Wang et al., 2023a). In contrast, our simple re-reading strategy for LLMs is orthogonal to these improvements via a trade-off between the intermediate steps and the query itself. Besides, our re-reading strategy is complementary to many previous works by preventing the answer from being derived overwhelmingly from the CoT but overlooking the original query.

Re-reading Strategy in NLP. In deep learning, the success of performing text-understanding tasks (Song et al., 2018; Luo et al., 2019a; Yang et al., 2019; Lei et al., 2019) depends on the heuristics of human reading strategy, e.g., pre-reading, re-reading and post-reading (Saricoban, 2002; Toprak and ALMACIOĞLU, 2009; Pressley and Afflerbach, 2012; Ozek and Civelek, 2006; Dowhower, 1989). Specifically, many effective algorithms have been crafted around the idea of re-reading. Although deep architectures, from multi-layer Bi-LSTM (Huang et al., 2015) to Transformer-encoder (Vaswani et al., 2017), have their mechanisms that provide a form of "re-reading", the notion that simply processing an input once might not be sufficient for understanding or generating a complex output has been long-standing. Initially, Sha et al. (2016) and Sha et al. (2017) found that repeated reading mechanisms do improve performance on some tasks, e.g., sentiment analysis, semantic relation classification, and event extraction. Then, Liu and Li (2016) propose to mimic the repeated reading strategy and present neural networks with multi-level attention, which is proven effective in recognizing implicit discourse relations. Sequentially, Zhu et al. (2018) propose a multi-glance mechanism, modeling the habit of reading behavior, which can benefit a wide range of tasks. Luo et al. (2019b) adopt a network to encode the gist of paragraphs for rough reading and a decision-making policy for careful reading, which can improve extractive summarization. More recently, Springer et al. (2024) have shown the effectiveness of repeating input to get bidirectional embeddings on text embedding tasks. Therefore, it is natural to introduce a re-reading strategy to LLMs' reasoning, since the Transformer-decoder architecture of LLMs with unidirectional attention mechanisms hinders the implicit bidirectional capability.

Knowledge Recall. From the perspective of information seeking, prompting LLMs can be seen as a sort of "knowledge recall" via a parametric fashion, where the prompt can be seen as a retrieval query. In contrast to conventional non-parametric retrieval – vector database (Karpukhin et al., 2020; Izacard et al., 2022) for example, the LLM as a neural knowledge model (Bosselut et al., 2019; AlKhamissi et al., 2022) can easily generalize for huge knowledge coverage, contributing to its efficacy in broad applications. In the context of CoT reasoning, (Chen et al., 2023) conjuncture that LLM can be exposed to certain CoTs during training and easily complete reasoning by knowledge recall. As such, it is natural to adapt the basic but prevalent query augmentation technique in the term-based retrieval domain (Dai and Callan, 2019), which repeats the original query multiple times over the augmented part (Wang et al., 2023b; Shen et al., 2023), into prompting LLMs.

5 Conclusion and Future Works

This paper introduces RE2, a simple and effective prompting method for LLM reasoning that improves performance by "re-reading" the question. By shifting focus to the input phase, RE2 operates independently from other thought-eliciting promptings. Moreover, it shows promise in fostering bidirectional comprehension of questions in decoder-only LLMs. Our comprehensive experiments cover a wide range of reasoning benchmarks, diverse LLM types, various task settings, and compatibility assessments with other prompting methods, validating the efficacy and versatility of RE2. Our findings encourage the research community to prioritize a deeper understanding of input questions, thereby complementing existing thought-eliciting prompting strategies. Future endeavors will aim to explore its versatility in additional contexts beyond reasoning, including multi-turn dialogue and multi-modal reasoning applications.

6 Limitations

In this paper, we introduce a simple yet effective prompting method for enhancing reasoning in LLMs and conduct extensive experiments to validate its effectiveness. Despite our best efforts, there may be still some limitations that remain in our study. Our investigation primarily revolves around empirical studies with extensive experiments to validate RE2, similar to most works in prompting research (Zheng et al., 2023; Yin et al., 2023; Gao et al., 2023). Future efforts will include more theoretical analyses to provide a solid foundation. Additionally, RE2 marginally increases the input length, leading to a slight reduction in efficiency for longer questions during inference. Future work will explore more scenarios except reasoning, such as multi-turn dialogue and multi-modal reasoning.

7 Ethics

We conducted experiments on seven mathematical reasoning benchmarks, comprising GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), ASDiv (Miao et al., 2020), AQuA (Ling et al., 2017), AddSub (Hosseini et al., 2014), MultiArith (Roy and Roth, 2015), SingelEQ (Roy et al., 2015), three commonsense reasoning benchmarks (CSQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021), and ARC (Clark et al., 2018)), and two symbolic benchmarks (Date Understanding (Suzgun et al., 2023a) and Coinflip (Wei et al., 2022b)). Among these, GSM8K and SVAMP datasets utilize code under the MIT License, while AQuA, StrategyQA, Date Understanding, Coinflip utilize code under the Apache-2.0 license, and ARC utilizes code under CC-BY-SA-4.0. The licenses for the remaining datasets are unspecified.

The proposed prompts do not involve the collection or utilization of personal information pertaining to other individuals. Details regarding the prompts used in our experiments are provided in Appendix §B. Furthermore, it is ensured that the prompts utilized in this research do not pose any threat to the safety or well-being of others.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was supported by the National Natural Science Foundation of China (No.61925203 and No.U22B2021).

References

Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona T. Diab, and Marjan Ghazvininejad. 2022. A review on language models as knowledge bases. *CoRR*, abs/2204.06031.

Konstantine Arkoudas. 2023. Gpt-4 can't reason. *arXiv preprint arXiv:2308.03762*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

Andrew Blair-Stanek, Nils Holzenberger, and Benjamin Van Durme. 2023. Can gpt-3 perform statutory reasoning? *arXiv preprint arXiv:2302.06100*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4762–4779. Association for Computational Linguistics.

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 *Advances in Neural Information Processing Systems*.

Jiuhai Chen, Lichang Chen, Heng Huang, and Tianyi Zhou. 2023. When do you need chain-of-thought prompting for chatgpt? *CoRR*, abs/2304.03262.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457.

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.

Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *CoRR*, abs/1910.10687.

Sarah L Dowhower. 1989. Repeated reading: Research into practice. *The Reading Teacher*, 42(7):502–507.

Sarah Lynn Dowhower. 1987. Effects of repeated reading on second-grade transitional readers' fluency and comprehension. *Reading Research Quarterly*, pages 389–406.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: general language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 320–335. Association for Computational Linguistics.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: program-aided language models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.

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*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2023. Rethinking with retrieval: Faithful large language model inference. *CoRR*, abs/2301.00303.

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)*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1049–1065. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*.

Zeyang Lei, Yujiu Yang, Min Yang, Wei Zhao, Jun Guo, and Yi Liu. 2019. A human-like semantic cognition network for aspect-level sentiment classification. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6650–6657. AAAI Press.

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)*.

Yang Liu and Sujian Li. 2016. Recognizing implicit discourse relations via repeated reading: Neural networks with multi-level attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1224–1233. The Association for Computational Linguistics.

Ling Luo, Xiang Ao, Yan Song, Jinyao Li, Xiaopeng Yang, Qing He, and Dong Yu. 2019a. Unsupervised neural aspect extraction with sememes. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5123–5129. ijcai.org.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019b. Reading like HER: human reading inspired extractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3031–3041. Association for Computational Linguistics.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *CoRR*, abs/2301.13379.

Shen Yun Miao, Chao Chun Liang, and Keh Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

NVIDIA. 2024. Cuda c++ programming guide. `https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html`. Accessed: 2024-09-29.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue.

R OpenAI. 2023. Gpt-4 technical report. *arXiv*, pages 2303–08774.

Yesim Ozek and Muharrem Civelek. 2006. A study on the use of cognitive reading strategies by elt students. *The Asian EFL Journal*, 14(1):1–26.

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.

Michael Pressley and Peter Afflerbach. 2012. *Verbal protocols of reading: The nature of constructively responsive reading*. Routledge.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2023. Reasoning with language model prompting: A survey. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5368–5393. Association for Computational Linguistics.

Ansh Radhakrishnan, Karina Nguyen, Anna Chen, Carol Chen, Carson Denison, Danny Hernandez, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamile Lukosiute, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Sam McCandlish, Sheer El Showk, Tamera Lanham, Tim Maxwell, Venkatesa Chandrasekaran, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. Question decomposition improves the faithfulness of model-generated reasoning. *CoRR*, abs/2307.11768.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about Quantities in Natural Language. *Transactions of the Association for Computational Linguistics*.

Arif Saricoban. 2002. Reading strategies of successful readers through the three phase approach. *The Reading Matrix*, 2(3).

Lei Sha, Baobao Chang, Zhifang Sui, and Sujian Li. 2016. Reading and thinking: Re-read LSTM unit for textual entailment recognition. In *COLING 2016,* *26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2870–2879. ACL.

Lei Sha, Feng Qian, and Zhifang Sui. 2017. Will repeated reading benefit natural language understanding? In *Natural Language Processing and Chinese Computing - 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8-12, 2017, Proceedings*, volume 10619 of *Lecture Notes in Computer Science*, pages 366–379. Springer.

Tao Shen, Guodong Long, Xiubo Geng, Chongyang Tao, Tianyi Zhou, and Daxin Jiang. 2023. Large language models are strong zero-shot retriever. *CoRR*, abs/2304.14233.

Yan Song, Shuming Shi, and Jing Li. 2018. Joint learning embeddings for chinese words and their components via ladder structured networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4375–4381. ijcai.org.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed Chi, Denny Zhou, and Jason Wei. 2023a. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13003–13051. Association for Computational Linguistics.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed Chi, Denny Zhou, and Jason Wei. 2023b. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13003–13051. Association for Computational Linguistics.

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)*.

Elif Toprak and Gamze ALMACIOĞLU. 2009. Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners. *Journal of language and Linguistic Studies*, 5(1).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *CoRR*, abs/2305.04388.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

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. *arXiv preprint arXiv:2305.04091*.

Liang Wang, Nan Yang, and Furu Wei. 2023b. Query2doc: Query expansion with large language models. *CoRR*, abs/2303.07678.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023c. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models.

Min Yang, Qiang Qu, Wenting Tu, Ying Shen, Zhou Zhao, and Xiaojun Chen. 2019. Exploring human-like reading strategy for abstractive text summarization. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7362–7369. AAAI Press.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *CoRR*, abs/2305.10601.

Yao Yao, Zuchao Li, and Hai Zhao. 2023b. Beyond chain-of-thought, effective graph-of-thought reasoning in large language models. *CoRR*, abs/2305.16582.

Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. 2023. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15135–15153, Singapore. Association for Computational Linguistics.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. 2023. Progressive-hint prompting improves reasoning in large language models. *CoRR*, abs/2304.09797.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Pengcheng Zhu, Yujiu Yang, Wenqiang Gao, and Yi Liu. 2018. Multi-glance reading model for text understanding. In *Proceedings of the Eight Workshop on Cognitive Aspects of Computational Language Learning and Processing*, pages 27–35, Melbourne. Association for Computational Linguistics.

A Datasets

Table 10 presents statistics and examples for the reasoning benchmarks we considered.

B Specific Prompting Methods

Detailed information regarding various promptings is shown in Table 11 and Table 12. The instructions of answer-format can be found in Table 13.

C GPT-4o-mini Experiments

LLMs are rapidly evolving, with more powerful models emerging frequently. To assess the effectiveness of RE2 on newer, more advanced models, we tested it on GPT-4o-mini, specifically the gpt-4o-mini-2024-07-18 version [3]. The results, presented in Figure 8 and Figure 9, demonstrate that RE2 continues to perform effectively on these more advanced LLMs.

D Attention Analysis

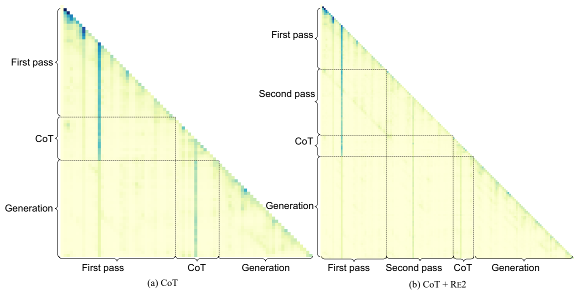

Figure 6: Attention visualization with and without RE2. (a) CoT prompting: there is only one pass for the question. (b) CoT+RE2 re-reads the question, including first pass and second pass. The row of matrix represents the query tokens and the column represents the key tokens.

To gain deeper insights into how RE2 reshapes attention during inference, we visualize the attention distribution by computing the average attention weights across all heads and layers in Llama-2. The results are presented in Figure 6, revealing two key findings: (1) In the block of "Second pass" attending to the "First pass" as shown in (b) for CoT+RE2, we observe explicit attentions in the upper triangle. This observation indicates that tokens in the second question can focus on the tokens behind the corresponding positions in the first question. In this way, RE2 enables a "bidirectional" understanding of the question. Notably, with the inclusion of RE2, the generation process maintains a higher attention weight on the question tokens. By calculating the proportion of attention weights assigned to the question tokens during generation, we observe an increase from 0.32 to 0.40 with the utilization of RE2. This finding suggests that the re-reading mechanism enhances the model's focus on the question during the reasoning process.

[3] https://platform.openai.com/docs/models/gpt-4o-mini

E Perplexity Analysis

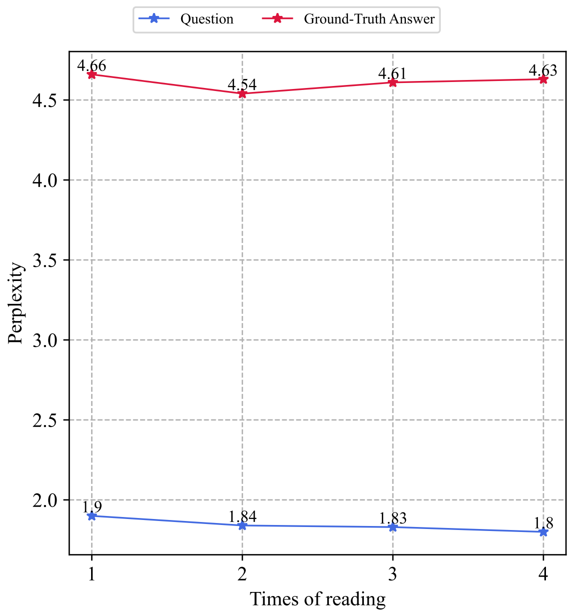

Figure 7: The perplexity of generating the question or the ground-truth answer with increasing reading times.

For the explanation about "overly repeating questions encourages LLMs to repeat the question rather than generate the answer" in Section 3.4, we conducted an experiment. This experiment aims to investigate the likelihood of generating questions versus generating ground-truth responses as reading times of the question increased. We pose two research questions: (1) Does the probability of generating the question as the output increase with more reading times? (2) Does the probability of generating the ground-truth response decrease with more reading times?

Specifically, for each question in the GSM8k dataset, we provide the LLM with the question with varying repetition times as input, and set the LLM's output as the question itself or its ground-truth response. We then calculate the perplexity of generating both the question and the ground-truth answer. Perplexity serves as an indicator reflecting the likelihood of generating a sequence, with lower

LLMs	Methods	GSM	SVAMP	ASDIV	AQUA	MultiArith	SingleEQ	AddSub
GPT-4o-mini	Vanilla	93.40	93.70	95.13	83.46	97.83	97.05	95.70
	Vanilla+RE2	94.09 ↑ 0.69	94.50 ↑ 0.80	95.28 ↑ 0.15	82.68 ↓ 0.78	98.00 ↑ 0.17	96.46 ↓ 0.59	97.22 ↑ 1.52
	CoT	92.87	93.80	94.80	80.31	97.50	96.85	95.95
	CoT+RE2	93.80 ↑ 0.93	94.20 ↑ 0.40	95.09 ↑ 0.29	82.28 ↑ 1.97	98.83 ↑ 1.33	97.05 ↑ 0.20	97.72 ↑ 1.77

Table 8: Results on arithmetic reasoning benchmarks for GPT-4o-mini.

LLMs	Methods	Commonsense				Symbolic	
		CSQA	StrategyQA	ARC-e	ARC-c	Date	Coin
GPT-4o-mini	Vanilla	82.56	75.02	95.12	92.15	66.40	99.80
	Vanilla+RE2	83.95 ↑ 1.39	76.99 ↑ 1.97	95.16 ↑ 0.04	93.17 ↑ 1.02	75.07 ↑ 8.67	100.00 ↑ 0.20
	CoT	82.64	79.13	95.58	93.60	70.73	100.00
	CoT+RE2	83.78 ↑ 1.14	79.52 ↑ 0.39	95.45 ↓ 0.13	93.34 ↓ 0.26	79.40 ↑ 8.67	99.80 ↓ 0.20

Table 9: Results on commonsense and symbolic reasoning benchmarks for GPT-4o-mini.

perplexity indicating a higher likelihood. These experiments are conducted using the Llama 2. The perplexity results are summarized in Figure 7.

The results reveal two key findings: (1) The perplexity of generating questions decreases with increasing reading times, suggesting that the LLM finds it easier to generate the question. (2) With the exception of when reading times = 2, the perplexity of generating the ground-truth response increases overall. This finding aligns with optimal performance observed when the question is read only twice. Moreover, as reading times increase, the LLM appears to be less inclined to generate the answer.

F Case Study

We also conduct a case study to show the effectiveness of our proposed re-reading prompting over the chain-of-thought. We choose two examples from GSM, and the results generated by ChatGPT are listed in Table 14-15. It is evident that our method can better align the evidence in the question with the corresponding explanation hints. We can observe that CoT+RE2 tends to highlight the important evidences in the question before generating the explanation, for example, "*In the morning, she gives 15 cups of feed, and in the afternoon, she gives another 25. So ...*" in Table 14 and "*The bonus is worth half a month's salary, which is ...*" in Table 15. This observation is also consistent with the right figure in Figure 4.

G More Cases

Tables 16-20 provide more examples generated by ChatGPT with CoT and CoT+RE2. We also provide several examples generated by davinci-003 and ChatGPT in the Vanilla prompting (e.g. no instruction) in Tables 21-24. They show that ChatGPT with Vanilla directly generates answer in Coin Filp and Date Understanding dataset (Tables 21-22), but still generates CoT output in some other datasets (Tables 23-24).

Dataset	Type	N	Answer	Example
GSM8K	Math	1319	Number	Josh decides to try flipping a house. He buys a house for $80,000 and then puts in $50,000 in repairs. This increased the value of the house by 150%. How much profit did he make?
SVAMP	Math	1000	Number	After resting they decided to go for a swim. The depth of the water is 15 times Dean's height. Dean is 4 feet taller than Ron. If Ron stands at 13 feet. How deep was the water?
ASDIV	Math	2096	Number	There are 3300 bananas in Janice's banana collection. Janice also has 5 crayons. If the bananas are organized into 75 groups, how big is each group?
AQUA	Math	254	Option	The original price of an item is discounted 22%. A customer buys the item at this discounted price using a $20-off coupon. There is no tax on the item, and this was the only item the customer bought. If the customer paid $1.90 more than half the original price of the item, what was the original price of the item? Answer Choices: A)$61, B)$65, C)$67.40, D)$70, E)$78.20.
MultiArith	Math	600	Number	For the school bake sale Robin made 42 cupcakes. If she sold 22 of them and then made 39 more, how many cupcakes would she have?
SingleEq	Math	508	Number	Alyssa spent half of her allowance going to the movies. She washed the family car and earned 8 dollars. What is her weekly allowance if she ended with 12 dollars ?
AddSub	Math	395	Number	Mike had 34 peaches at his roadside fruit dish . He went to the orchard and picked peaches to stock up . There are now 86 peaches . how many did he pick ?
CSQA	CS	1221	Option	Where would you find magazines along side many other printed works? Answer Choices: A)doctor, B)bookstore, C)market, D)train station, E)mortuary.
StrategyQA	CS	2290	Yes / No	Do the anchors on Rede Globo speak Chinese?
ARC-e	CS	2376	Option	The shape of the moon appears to change regularly during each month. Which of the following best explains why the shape of the moon appears to change? Answer Choices: A)The Earth turns on its axis, B)The Moon turns on its axis, C)The Moon orbits around the Earth, D)Clouds cover the Moon.
ARC-c	CS	1172	Option	What is a similarity between sound waves and light waves? Answer Choices: A)Both carry energy, B)Both travel in vacuums, C)Both are caused by vibrations, D)Both are traveling at the same speed.
Date Understanding	Sym.	369	Date	Yesterday was April 30, 2021. What is the date today in MM/DD/YYYY?
Coin Flip	Sym.	500	Yes / No	A coin is heads up. Breanna flips the coin. Trey does not flip the coin. Omar flips the coin. Patrice does not flip the coin. Is the coin still heads up? Note that "flip" here means "reverse".

Table 10: Details of reasoning benchmarks. Math: arithmetic reasoning. CS: commonsense reasoning. Sym.: symbolic reasoning.

<table>
<tr><th>Methods</th><th>Prompt Content</th></tr>
<tr><td>Vanilla</td><td>Q: {question}
#Answer format instruction#
A:</td></tr>
<tr><td>Vanilla+RE2</td><td>Q: {question}
Read the question again: {question}
#Answer format instruction#
A:</td></tr>
<tr><td>CoT</td><td>Q: {question}
#Answer format instruction#
A: Let’s think step by step.</td></tr>
<tr><td>CoT+RE2</td><td>Q: {question}
Read the question again: {question}
#Answer format instruction#
A: Let’s think step by step.</td></tr>
<tr><td>PS</td><td>Q: {question}
#Answer format instruction#
A: Let’s first understand the problem and devise a plan to solve the problem.
Then, let’s carry out the plan, solve the problem step by step,
and give the ultimate answer. Please explicitly generate the mentioned process:
[Problem Understanding], [Plan], [Solving/Calculations], [Answer]. in your response.</td></tr>
<tr><td>PS+RE2</td><td>Q: {question}
Read the question again: {question}
#Answer format instruction#
A: Let’s first understand the problem and devise a plan to solve the problem.
Then, let’s carry out the plan, solve the problem step by step,
and give the ultimate answer. Please explicitly generate the mentioned process:
[Problem Understanding], [Plan], [Solving/Calculations], [Answer]. in your response.</td></tr>
</table>

Table 11: Specific prompts of Vanilla, Vanilla+RE2, CoT, CoT+RE2, PS, and PS+RE2.

Methods	Prompt Content
PAL	#!/bin/python3 import math import numpy as np import statistics import sympy as sp ########### Task Instruction ############## # You will write python program to solve math problems. # You will only write code blocks. # Please generate your code block in ‘def solution()‘ function, and thus it can be executed by python interpreter. You don’t need to call ‘solution()‘ function because it will be called by the system. # The concrete format of ‘solution()‘ is as follows: # def solution(): # """<question>""" # <your code> # result = <your result> # return result ### # Q: {question} # Your defined ”solution()” function with comments here.
PAL+RE2	#!/bin/python3 import math import numpy as np import statistics import sympy as sp ########### Task Instruction ############## # You will write python program to solve math problems. # You will only write code blocks. # Please generate your code block in ‘def solution()‘ function, and thus it can be executed by python interpreter. You don’t need to call ‘solution()‘ function because it will be called by the system. # The concrete format of ‘solution()‘ is as follows: # def solution(): # """<question>""" # <your code> # result = <your result> # return result ### # Q: {question} # Read the question again: {question} # Your defined ”solution()” function with comments here.

Table 12: Specific prompts of PAL and PAL+RE2

Tasks	Answer-format Instructions
GSM	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
SVAMP	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
ASDIV	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
AQUA	Your answer should be in the form $\boxed{choice}$. There is only one correct choice.
MultiArith	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
SingleEQ	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
AddSub	Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
CommonsenseQA	Your answer should be in the form $\boxed{choice}$. There is only one correct choice.
StrategyQA	Your answer should be either $\boxed{yes}$ or $\boxed{no}$, in the form $\boxed{answer}$.
ARC	Your answer should be in the form $\boxed{choice}$. There is only one correct choice.
Date Understanding	Your answer should be a date, in the format of $\boxed{MM/DD/YYYY}$, e.g. $\boxed{05/01/2022}$.
Coin Flip	Your answer should be either $\boxed{yes}$ or $\boxed{no}$, in the form $\boxed{answer}$.

Table 13: Specific answer-format instruction in each task.

Q: Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens? (Answer: 20)

CoT

Input:
Q: Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate how much feed Wendi gives her chickens in the morning. She has 20 chickens, and she gives each chicken 3 cups of feed, so in total, she gives them 20 x 3 = 60 cups of feed in the morning.
Next, let's calculate how much feed Wendi gives her chickens in the afternoon. Again, she has 20 chickens, and she gives each chicken 3 cups of feed, so in total, she gives them 20 x 3 = 60 cups of feed in the afternoon.
To find out how much feed she needs to give her chickens in the final meal of the day, we need to subtract the total feed given in the morning and afternoon from the total feed needed for the day. The total feed needed for the day for 20 chickens is 20 x 3 x 3 = 180 cups of feed.
So, the feed needed in the final meal of the day is 180 - 60 - 60 = 60 cups of feed.
Therefore, Wendi needs to give her chickens 60 cups of feed in the final meal of the day.
Answer: $\boxed{60}$

CoT+RE2

Input:
Q: Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?
Read the question again: Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, we need to determine the total amount of feed Wendi gives her chickens in the morning and afternoon. In the morning, she gives 15 cups of feed, and in the afternoon, she gives another 25 cups. So, the total amount of feed given in the morning and afternoon is 15 + 25 = 40 cups.
Next, we need to determine the total amount of feed needed for Wendi's flock of 20 chickens. Since each chicken requires 3 cups of feed, we multiply 3 by 20 to get 60 cups.
Finally, we subtract the total amount of feed given in the morning and afternoon from the total amount needed. 60 - 40 = 20
Therefore, Wendi needs to give her chickens 20 cups of feed in the final meal of the day.
Answer: $\boxed{20}$

Table 14: GSM Example 4 (Complexity=3)

Q: Tim gets a promotion that offers him a 5% raise on his $20000 a month salary. It also gives him a bonus worth half a month's salary. How much money will he make in a year? (Answer: 262500)

CoT

Input:
Q: Tim gets a promotion that offers him a 5% raise on his $20000 a month salary. It also gives him a bonus worth half a month's salary. How much money will he make in a year?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate the 5% raise on Tim's $20000 monthly salary. To do this, we multiply $20000 by 5% (or 0.05):
$20000 * 0.05 = $1000
So, Tim's raise is $1000 per month.
Next, let's calculate the bonus worth half a month's salary. Half of $20000 is $10000, so Tim's bonus is $10000.
Now, let's calculate Tim's total monthly salary after the raise and bonus:
$20000 (original salary) + $1000 (raise) + $10000 (bonus) = $31000
Finally, let's calculate Tim's annual salary by multiplying his monthly salary by 12:
$31000 * 12 = $372000
Therefore, Tim will make $372000 in a year. So, the answer is $\boxed{372000}$.

CoT+RE2

Input:
Q: Tim gets a promotion that offers him a 5% raise on his $20000 a month salary. It also gives him a bonus worth half a month's salary. How much money will he make in a year?
Read the question again: Tim gets a promotion that offers him a 5% raise on his $20000 a month salary. It also gives him a bonus worth half a month's salary. How much money will he make in a year?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate the 5% raise on Tim's $20000 monthly salary.
The raise amount would be 5% of $20000, which is equal to 0.05 * $20000 = $1000.
So, Tim's new monthly salary would be $20000 + $1000 = $21000.
Next, let's calculate the bonus amount. The bonus is worth half a month's salary, which is equal to 0.5 * $21000 = $10500.
Now, to find out how much money Tim will make in a year, we need to calculate the total salary for 12 months and add the bonus.
Total salary for 12 months = $21000 * 12 = $252000.
So, Tim will make a total of $252000 + $10500 = $262500 in a year.
Therefore, Tim will make $\boxed{262500}$ in a year.

Table 15: GSM Example 641 (Complexity=6)

Q: Colby loves going to the movies and every month his parents give him $150 to spend at the movies. Tickets for Fridays and Saturdays cost $10. Tickets for any other day cost $7. Popcorn costs $8 and boxes of candy cost $2. It is the last day of the month and it's a Friday. He wants to make sure he gets a popcorn and box of candy that night. How many movies can he see if he already saw 5 movies on a Friday or Saturday, 8 movies on other days, had 2 tubs of popcorn, and four boxes of candy that month? (Answer: 1)

CoT

Input:
Q: Colby loves going to the movies and every month his parents give him $150 to spend at the movies. Tickets for Fridays and Saturdays cost $10. Tickets for any other day cost $7. Popcorn costs $8 and boxes of candy cost $2. It is the last day of the month and it's a Friday. He wants to make sure he gets a popcorn and box of candy that night. How many movies can he see if he already saw 5 movies on a Friday or Saturday, 8 movies on other days, had 2 tubs of popcorn, and four boxes of candy that month?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate the cost of the movies Colby has already seen. He has seen 5 movies on a Friday or Saturday, which cost $10 each, so that is a total of 5 * $10 = $50. He has also seen 8 movies on other days, which cost $7 each, so that is a total of 8 * $7 = $56. Therefore, the total cost of the movies he has already seen is $50 + $56 = $106.
Next, let's calculate the cost of the popcorn and candy he wants to buy on the last day of the month. The popcorn costs $8 and the candy costs $2, so the total cost of the snacks is $8 + $2 = $10.
Now, let's subtract the cost of the movies Colby has already seen and the cost of the snacks from the $150 his parents gave him. $150 - $106 - $10 = $34.
Since each movie on a Friday or Saturday costs $10 and Colby has $34 left, he can see a maximum of $34 / $10 = 3 movies on a Friday or Saturday.
In conclusion, Colby can see a total of 3 + 5 = 8 movies in the month.
$\boxed{8}$

CoT+RE2

Input:
Q: Colby loves going to the movies and every month his parents give him $150 to spend at the movies. Tickets for Fridays and Saturdays cost $10. Tickets for any other day cost $7. Popcorn costs $8 and boxes of candy cost $2. It is the last day of the month and it's a Friday. He wants to make sure he gets a popcorn and box of candy that night. How many movies can he see if he already saw 5 movies on a Friday or Saturday, 8 movies on other days, had 2 tubs of popcorn, and four boxes of candy that month?
Read the question again: Colby loves going to the movies and every month his parents give him $150 to spend at the movies. Tickets for Fridays and Saturdays cost $10. Tickets for any other day cost $7. Popcorn costs $8 and boxes of candy cost $2. It is the last day of the month and it's a Friday. He wants to make sure he gets a popcorn and box of candy that night. How many movies can he see if he already saw 5 movies on a Friday or Saturday, 8 movies on other days, had 2 tubs of popcorn, and four boxes of candy that month?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate the total amount Colby has spent so far:
For the 5 movies on Fridays and Saturdays, he spent 5 x $10 = $50. For the 8 movies on other days, he spent 8 x $7 = $56. For the 2 tubs of popcorn, he spent 2 x $8 = $16. For the 4 boxes of candy, he spent 4 x $2 = $8.
Adding all these expenses together, Colby has spent $50 + $56 + $16 + $8 = $130.
Now, let's calculate how much money he has left from the $150 his parents gave him:
Colby started with $150 - $130 = $20.
Since it's the last day of the month and it's a Friday, he wants to buy a popcorn ($8) and a box of candy ($2). This means he will spend $8 + $2 = $10.
Therefore, Colby will have $20 - $10 = $10 left.
Now, let's calculate how many movies he can see with the remaining $10:
Since tickets on Fridays and Saturdays cost $10, and he has $10 left, he can see 1 movie on a Friday or Saturday.
Therefore, Colby can see 1 more movie on the last day of the month.
So, the final answer is $\boxed{1}$.

Table 16: GSM Example 926 (Complexity>=9)

Q: Mike was a pen pal with 5 people. He stopped being penpals with 2 of them. They each send 2 letters a week that are 5 pages long. He responds in kind. He can write a page every 6 minutes. How many hours does he spend writing a week? (Answer: 3)

CoT

Input:
Q: Mike was a pen pal with 5 people. He stopped being penpals with 2 of them. They each send 2 letters a week that are 5 pages long. He responds in kind. He can write a page every 6 minutes. How many hours does he spend writing a week?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate how many letters Mike receives in a week from the 5 pen pals. Each pen pal sends 2 letters per week, so 5 pen pals would send him 5 * 2 = 10 letters in a week.
Next, let's calculate how many pages Mike needs to write in response to each letter. Each letter is 5 pages long, and Mike responds in kind, so he would write 5 pages for each letter he receives.
Therefore, Mike needs to write a total of 10 * 5 = 50 pages in a week.
Now, let's calculate how many minutes it takes for Mike to write one page. We are given that he can write a page in 6 minutes. Since Mike needs to write 50 pages, it will take him 50 * 6 = 300 minutes to complete all the writing.
Lastly, let's convert the minutes to hours. There are 60 minutes in an hour, so 300 minutes is equal to 300 / 60 = 5 hours.
Therefore, Mike spends 5 hours writing in a week.
Answer: $\boxed{5}$.

CoT+RE2

Input:
Q: Mike was a pen pal with 5 people. He stopped being penpals with 2 of them. They each send 2 letters a week that are 5 pages long. He responds in kind. He can write a page every 6 minutes. How many hours does he spend writing a week?
Read the question again: Mike was a pen pal with 5 people. He stopped being penpals with 2 of them. They each send 2 letters a week that are 5 pages long. He responds in kind. He can write a page every 6 minutes. How many hours does he spend writing a week?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.
Output:
First, let's calculate the total number of letters that Mike receives in a week. Since he was pen pals with 5 people, and he stopped being pen pals with 2 of them, he is still receiving letters from 5 - 2 = 3 people.
Each of these 3 people sends 2 letters a week, so the total number of letters that Mike receives in a week is 3 * 2 = 6 letters.
Next, let's calculate the total number of pages that Mike needs to write in response. Each letter is 5 pages long, so the total number of pages that Mike needs to write is 6 * 5 = 30 pages.
Since Mike can write a page every 6 minutes, the total time he spends writing in a week is 30 * 6 = 180 minutes.
To convert this to hours, we divide by 60 since there are 60 minutes in an hour. So, Mike spends 180 / 60 = 3 hours writing in a week.
Therefore, Mike spends $\boxed{3}$ hours writing a week.

Table 17: GSM Example 242 (Complexity=6)

Q: Terri is knitting a sweater with two sleeves, a collar, and a decorative rosette. The body of the sweater takes 900 stitches to complete, the collar takes a tenth of that number of stitches, and the rosette takes twice as many as the collar. The whole sweater is an 1800-stitch project. How many stitches does each sleeve take? (Answer: 315)

CoT

Input:

Q: Terri is knitting a sweater with two sleeves, a collar, and a decorative rosette. The body of the sweater takes 900 stitches to complete, the collar takes a tenth of that number of stitches, and the rosette takes twice as many as the collar. The whole sweater is an 1800-stitch project. How many stitches does each sleeve take?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.

Output:

First, we know that the body of the sweater takes 900 stitches to complete.
Next, we know that the collar takes a tenth of the number of stitches as the body. So, the collar takes 900 stitches / 10 = 90 stitches.
Finally, we know that the rosette takes twice as many stitches as the collar. So, the rosette takes 90 stitches * 2 = 180 stitches.
To find out how many stitches each sleeve takes, we need to subtract the stitches of the body, collar, and rosette from the total number of stitches in the sweater.
So, each sleeve takes 1800 stitches - 900 stitches (body) - 90 stitches (collar) - 180 stitches (rosette) = 630 stitches.
Therefore, each sleeve takes $\boxed{630}$ stitches.

CoT+RE2

Input:

Q: Terri is knitting a sweater with two sleeves, a collar, and a decorative rosette. The body of the sweater takes 900 stitches to complete, the collar takes a tenth of that number of stitches, and the rosette takes twice as many as the collar. The whole sweater is an 1800-stitch project. How many stitches does each sleeve take?
Read the question again: Terri is knitting a sweater with two sleeves, a collar, and a decorative rosette. The body of the sweater takes 900 stitches to complete, the collar takes a tenth of that number of stitches, and the rosette takes twice as many as the collar. The whole sweater is an 1800-stitch project. How many stitches does each sleeve take?
Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response.
A: Let's think step by step.

Output:

First, we know that the body of the sweater takes 900 stitches to complete.
Next, we are told that the collar takes a tenth of the number of stitches in the body of the sweater. Since the body has 900 stitches, the collar will have 900 / 10 = 90 stitches.
Finally, we are told that the rosette takes twice as many stitches as the collar. Since the collar has 90 stitches, the rosette will have 90 * 2 = 180 stitches.
Now, let's add up all the stitches to find the total number of stitches in the sweater: 900 (body) + 90 (collar) + 180 (rosette) = 1170 stitches.
We are given that the whole sweater is an 1800-stitch project. So, the remaining stitches must be for the sleeves.
1800 (total stitches) - 1170 (stitches accounted for) = 630 stitches for the sleeves.
Since there are two sleeves, each sleeve will have 630 / 2 = 315 stitches.
Therefore, each sleeve takes 315 stitches.

Table 18: GSM Example 273 (Complexity=6)

Q: Solo has to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook and 8 pages from his Geography textbook. Solo read 15 pages on Monday. If he has 4 more days to complete all his reading, how many pages does he need to read, on average, in one day? (Answer: 6)
CoT
Input: Q: Solo has to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook and 8 pages from his Geography textbook. Solo read 15 pages on Monday. If he has 4 more days to complete all his reading, how many pages does he need to read, on average, in one day? Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response. A: Let's think step by step. **Output:** First, let's calculate how many pages Solo has left to read. He needs to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook, and 8 pages from his Geography textbook. So, the total number of pages he needs to read is 4 + 20 + 7 + 8 = 39 pages. Since Solo has 4 more days to complete all his reading, we can divide the total number of pages by the number of days. Therefore, Solo needs to read an average of 39 pages / 4 days = 9.75 pages per day. Therefore, Solo needs to read an average of $\boxed{9.75}$ pages per day.
CoT+RE2
Input: Q: Solo has to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook and 8 pages from his Geography textbook. Solo read 15 pages on Monday. If he has 4 more days to complete all his reading, how many pages does he need to read, on average, in one day? Read the question again: Solo has to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook and 8 pages from his Geography textbook. Solo read 15 pages on Monday. If he has 4 more days to complete all his reading, how many pages does he need to read, on average, in one day? Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response. A: Let's think step by step. **Output:** First, let's calculate the total number of pages Solo needs to read. He needs to read 4 pages from his Science textbook, 20 pages from his Social Studies textbook, 7 pages from his History textbook, and 8 pages from his Geography textbook. So, the total number of pages he needs to read is 4 + 20 + 7 + 8 = 39 pages. Next, let's calculate the number of pages he has left to read after Monday. He read 15 pages on Monday, so he has 39 - 15 = 24 pages left to read. Finally, let's calculate the average number of pages he needs to read per day. He has 4 more days to complete his reading, so he needs to read 24 / 4 = 6 pages per day. Therefore, Solo needs to read an average of $\boxed{6}$ pages per day.

Table 19: GSM Example 101 (Complexity=6)

Q: Would a nickel fit inside a koala pouch? (Answer: yes)
CoT
Input: Q: Would a nickel fit inside a koala pouch? Your answer should be either $\boxed{yes}$ or $\boxed{no}$, in the form $\boxed{answer}$. A: Let's think step by step. **Output:** First, let's consider the size of a nickel. A nickel has a diameter of approximately 21.21 mm and a thickness of approximately 1.95 mm. Next, let's consider the size of a koala pouch. A koala pouch is designed to hold a baby koala, also known as a joey. The size of a koala pouch can vary, but it is generally not large enough to fit objects the size of a nickel. Therefore, based on the size of a nickel and the size of a koala pouch, it is unlikely that a nickel would fit inside a koala pouch. $\boxed{No}$
CoT+RE2
Input: Q: Would a nickel fit inside a koala pouch? Read the question again: Would a nickel fit inside a koala pouch? Your answer should be either $\boxed{yes}$ or $\boxed{no}$, in the form $\boxed{answer}$. A: Let's think step by step. **Output:** First, let's consider the size of a nickel. A nickel has a diameter of about 0.835 inches (21.21 mm) and a thickness of about 0.077 inches (1.95 mm). Next, let's consider the size of a koala pouch. A koala pouch is designed to hold a baby koala, also known as a joey. The pouch is stretchy and expands as the joey grows. However, the size of the pouch may vary depending on the age and size of the joey. Based on the size of a nickel and the stretchiness of a koala pouch, it is possible that a nickel could fit inside a koala pouch. However, it would depend on the size of the pouch and the position of the joey inside. So, the answer is $\boxed{yes}$, a nickel could potentially fit inside a koala pouch.

Table 20: StrategyQA Example 262

Input: A coin is heads up. Hank does not flip the coin. Janine does not flip the coin. Frankie flips the coin. Isa flips the coin. Is the coin still heads up? Note that "flip" here means "reverse". Your answer should be either $\boxed{yes}$ or $\boxed{no}$, in the form $\boxed{answer}$. A:
davinci-003 with Vanilla
Output: No.
ChatGPT with Vanilla
Output: No.

Table 21: Coin Flip Example 16. Without explicit instruction, ChatGPT and davinci-003 directly generate the final answer.

Input: Yesterday was April 30, 2021. What is the date tomorrow in MM/DD/YYYY? Your answer should be a date, in the format of $\boxed{MM/DD/YYYY}$, e.g. $\boxed{05/01/2022}$. A:
davinci-003 with Vanilla
Output: 05/01/2021
ChatGPT with Vanilla
Output: 11/13/2019.

Table 22: Date Understanding Example 1. Without explicit instruction, ChatGPT and davinci-003 directly generate the final answer.

Input: Ned had to wash 9 short sleeve shirts and 21 long sleeve shirts before school. If he had only washed 29 of them by the time school started, how many did he not wash? Your final answer should be a single numerical number, in the form $\boxed{answer}$, at the end of your response. A:
davinci-003 with Vanilla
Output: He did not wash 20 shirts. $\boxed{20}$
ChatGPT with Vanilla
Output: Ned had to wash a total of 9 + 21 = 30 shirts.He washed 29 shirts, so he did not wash 30 - 29 = «30-29=1»1 shirt. Therefore, he did not wash $\boxed{1}$ shirt.

Table 23: MultiArith Example 9. Without explicit instruction, ChatGPT still can generate the CoT output, but davinci-003 does not.

Input: A trader sold an article at a profit of 20% for Rs.360. What is the cost price of the article? Answer Choices: A) 270, B) 300, C) 280, D) 320, E) 315 Your answer should be in the form $\boxed{choice}$. There is only one correct choice. A:
davinci-003 with Vanilla
Output: $\boxed{B}$
ChatGPT with Vanilla
Output: Let's assume the cost price of the article is x. The selling price of the article is given as Rs. 360, which is 120% of the cost price (100% + 20% profit). So, we can write the equation as: x + 20% of x = 360 Simplifying the equation, we get: 1.2x = 360 Dividing both sides by 1.2, we find: x = 300 Therefore, the cost price of the article is Rs. 300. The correct answer is $\boxed{(B)300}$.

Table 24: AQUA Example 4. Without explicit instruction, ChatGPT still can generate the CoT output, but davinci-003 does not.