Towards A Unified View of Answer Calibration for Multi-Step Reasoning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) employing Chain-of-Thought (CoT) prompting have broadened the scope for improving multi-step reasoning capabilities. We generally divide multi-step reasoning into two phases: *path generation* to generate the reasoning path(s); and *answer calibration* post-processing the reasoning path(s) to obtain a final answer. However, the existing literature lacks systematic analysis on different answer calibration approaches. In this paper, we summarize the taxonomy of recent answer calibration techniques and break them down into step-level and path-level strategies. We then conduct a thorough evaluation on these strategies from a unified view, systematically scrutinizing step-level and path-level answer calibration across multiple paths. Experimental results reveal that integrating the dominance of both strategies tends to derive optimal outcomes. Our study holds the potential to illuminate key insights for optimizing multi-step reasoning with answer calibration.

1 Introduction

Chain-of-Thought (CoT) prompting (Wei et al., 2022) has significantly improved multi-step reasoning capabilities of Large Language Models (LLMs) (Zhao et al., 2023b; Qiao et al., 2023). As seen from Figure 1, the process of multi-step reasoning generally contains two primary modules: *reasoning path generation* which generates one or multiple reasoning paths (Fu et al., 2023; Yao et al., 2023b); and *answer calibration* which post-processes the reasoning path(s) to calibrate the initial output (Wang et al., 2023i; Zhao et al., 2023a).

In practice, answer calibration is pluggable and can be integrated into path generation models. The answer calibration framework can be divided into step and path levels, applicable to single or multiple paths, as illustrated in Figure 1. For *step-level* answer calibration on a single path, the model

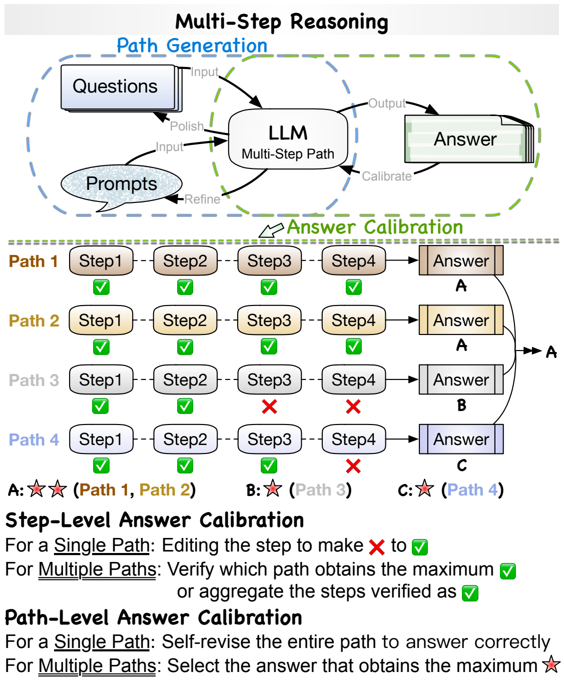


Figure 1: Illustration of answer calibration for multi-step reasoning with LLM.

rectifies errors in intermediate-step answers of a generated path (Zhao et al., 2023a). For *step-level* answer calibration on multiple paths, the model verifies each intermediate-step answer (Weng et al., 2023) or aggregates the correct step answers (Cao, 2023) from multiple paths. For *path-level* answer calibration on a single path, the model revises the entire rationale to obtain the correct answer (Baek et al., 2023). For *path-level* answer calibration on multiple paths, the model produces a result indicating the consensus of all candidate paths (Wang et al., 2023i; Yoran et al., 2023). As answer calibration can identify and rectify errors in the reasoning path, or even holistically utilize multiple candidate paths, it plays a vital role in multi-step reasoning to ensure a precise, consistent and reliable reasoning process (Pan et al., 2023).

However, we argue that the crucial factors driving the success of answer calibration strategies remain obscure, with a comprehensive systematic analysis still underexplored. To bridge the gap, our study investigates: (1) The specific conditions where answer calibration notably boosts multi-step reasoning performance; (2) The strengths and weaknesses of step-level versus path-level answer calibration, and the pathway to attaining optimal performance; (3) The robustness and generalizability of answer calibration strategies.

To address these questions, we dissect cutting-edge answer calibration techniques for multi-step reasoning with LLMs, and introduce a unified framework that elucidates step-level and path-level strategies. We define two thresholds to respectively signify the step-level and path-level dominance in the unified framework. We then undertake a comprehensive evaluation of answer calibration strategies, *w.r.t.* accuracy, faithfulness, informativeness, consistency, and perplexity over steps or paths. Through rigorous experiments on five representative multi-step reasoning tasks involving arithmetic and commonsense, we find that: **(1)** employing answer calibration can enhance accuracy, with the improvement being more noticeable in zero-shot scenarios (§4.2) and less significant on stronger backbone models (§4.4); **(2)** The optimal performance of the unified answer calibration strategy typically achieved by synthesizing step-level and path level dominance (§4.3); **(3)** path-level answer calibration is more beneficial in improving accuracy, and step-level answer calibration is more effective for mitigating low-quality prompting (§4.5); **(4)** answer calibration can improve consistency on arithmetic tasks but weakens faithfulness, informativeness and perplexity on both arithmetic and commonsense tasks (§4.6).

2 Related Work

Reasoning Path Generation. Previous methods for reasoning path generation mostly focus on two aspects to improve reasoning process, including refining input query or prompts (*input refinement*) and polishing the reasoning path (*rationale polish*).

As for *input refinement*, Zero-shot CoT (Kojima et al., 2022) and Few-shot CoT (Wei et al., 2022) are classic methods to elicit multi-step reasoning ability of LLMs, with "Let's think step by step" prompts. To decouple planning and execution, Wang et al. (2023g); Sun et al. (2023) devise a plan by prompting and divide and conquer multi-step tasks. To enrich prompts, Wang et al. (2023b) leverage structure triples as evidence, Kong et al. (2023) design role-play prompting, and Xu et al. (2023) employ re-reading instructions. Besides, LLM performance can also be affected by prompt complexity (Fu et al., 2023) and formats, such as program (Gao et al., 2023; Chen et al., 2023; Sel et al., 2023; Jie et al., 2023; Lei and Deng, 2023; Wang et al., 2023d; Bi et al., 2024) and table (Jin and Lu, 2023). Further, Wang et al. (2023c); Shi et al. (2023); Liang et al. (2023) propose to adaptively utilize prompts. Apart from refining prompts, Xi et al. (2023b) progressively refine the given questions, Wang et al. (2023j) convert semantically-wrapped questions to meta-questions, and Jie and Lu (2023) augment training data with program annotations.

In terms of *rationale polish*, recent work mainly focus on step-aware training (Wang et al., 2023k) and path-level optimization. For step-aware training, Zhang et al. (2023) introduce step-by-step planning and Lee and Kim (2023) recursively tackle intermediate steps; Jiang et al. (2023a) reconstruct the reasoning rationale within prompts by residual connections; Paul et al. (2023) iteratively provide feedback on step answers; Lanchantin et al. (2023) leverage self-notes as intermediate steps and working memory; Li et al. (2023b); Ling et al. (2023); Lightman et al. (2023) propose to verify on intermediate step answers; Li et al. (2023a); Wang et al. (2023e) process step-aware verification by knowledge base retrieval. For path-level optimization, Li and Qiu (2023) enable LLMs to self-improve via pre-thinking and recalling relevant reasoning paths as memory; Wang et al. (2023d); Yue et al. (2023) leverage hybrid rationales in formats of natural language and program. Some work also generate deliberate rationales beyond CoT, such as Tree-of-Thought (Yao et al., 2023b; Long, 2023), Graph-of-Thought (Yao et al., 2023e; Besta et al., 2023) and Hypergraph-of-Thought (HoT) (Yao et al., 2023a).

Answer Calibration. Given generated reasoning path(s), answer calibration methods *post-process* the path(s) to calibrate the answer, involving step- or path-level calibration on one or multiple path(s).

Step-level answer calibration. Xue et al. (2023); Cao (2023) propose to rectify factual inconsistency and reasoning logic between intermediate steps. Miao et al. (2023); Wu et al. (2024) check the correctness of each intermediate step. Zhao et al. (2023a) post-edit multi-step reasoning paths with external knowledge. Yao et al. (2023c); Hao et al. (2023); Shinn et al. (2023); Yao et al. (2023d) draw up a plan and act step by step with LLMs as agents

(Wang et al., 2023f; Xi et al., 2023a), encouraging interaction with the environment to provide feedback. Weng et al. (2023); Jiang et al. (2023b) unleash the self-verification ability of LLMs, by forward reasoning and backward verification on intermediate step answers. Zhou et al. (2023) propose code-based self-verification on reasoning steps.

Path-level answer calibration. Zelikman et al. (2022) present a self-taught reasoner to iteratively generate rationales. Zheng et al. (2023) progressively use the generated answers as hints to make double-check. Mountantonakis and Tzitzikas (2023) enrich generated reasoning paths with hundreds of RDF KGs for fact checking. Baek et al. (2023) iteratively rectify errors in knowledge retrieval and answer generation for knowledge-augmented LMs. To cultivate the reasoning ability of smaller LMs, Ho et al. (2023); Wang et al. (2023h,l) propose to fine-tune CoT for knowledge distillation. Huang et al. (2022) demonstrate that LLMs can self-improve with high-confidence rationale-augmented answers. Yoran et al. (2023) prompt LLMs to meta-reason over multiple paths. Liu et al. (2023); Madaan et al. (2023) leverage feedback to improve model initial outputs. Wan et al. (2023) adaptively select in-context demonstrations from previous outputs to re-generate answers. Wang et al. (2023i) leverage self-consistency decoding strategy to majority vote on multiple path answers. Aggarwal and Yang (2023) propose adaptive-consistency to reduce sample budget.

3 Comprehensive Analysis of Answer Calibration

3.1 Formulation of Answer Calibration

Given a question denoted as $\mathcal{Q}$ and its associated prompt P, we leverage the LLM to generate the result $\mathcal{R}$. $\mathcal{R}$ can either encompass a single reasoning path $\mathcal{P}$ with an initial answer $\mathcal{A}$ or multiple reasoning paths $\mathbb{P} = \{\mathcal{P}_i\}_{i\in[1,N]}$ with a corresponding answer set $\mathbb{A} = \{\mathcal{A}_i\}_{i\in[1,N]}$. The total number of paths in $\mathbb{P}$ is N. In this paper, we analyze under the assumption that each reasoning path comprises a maximum of M steps. Paths exceeding M steps are truncated, and those with fewer steps are padded. The intermediate step answers for each reasoning path $\mathcal{P}_{(i)}$ are represented as $\{a_j\}^{(i)}_{j\in[1,M]}$.

Step-Level Answer Calibration. Given a single reasoning path $\mathcal{P}$ with an initial final path answer $\mathcal{A}$ and intermediate step answers $\{a_j\}_{j\in[1,M]}$, the objective of step-level answer calibration is to rectify any erroneous a_j, so that deriving the correct $\mathcal{A}$. For multiple reasoning paths $\mathbb{P}$, step-level answer calibration seeks to either select the reasoning path with the maximum correct intermediate step answers or aggregate the verified correct steps to form the most accurate reasoning path, leading to a correct final path answer. *Self-verification* (Weng et al., 2023) is an effective approach for step-level answer calibration on multiple reasoning paths.

Path-Level Answer Calibration. Given a single reasoning path $\mathcal{P}$ with an initial final path answer $\mathcal{A}$, the goal of path-level answer calibration is to revise the wrong $\mathcal{A}$. For multiple reasoning paths $\mathbb{P} = \{\mathcal{P}_i\}_{i\in[1,N]}$ with corresponding answers $\mathbb{A} = \{\mathcal{A}_i\}_{i\in[1,N]}$, path-level answer calibration is designed to select the reasoning path from $\mathbb{P}$ with the most consistent answer in $\mathbb{A}$. *Self-consistency* (Wang et al., 2023i) is a widely-used efficacious technique for path-level answer calibration on multiple reasoning paths.

3.2 Unified View of Answer Calibration

Considering the advantages of both step-level and path-level answer calibration, we propose to integrate the two strategies on multiple paths. Given the multiple generated reasoning paths $\mathbb{P} = \{\mathcal{P}_i\}_{i\in[1,N]}$, we define a unified score $\mathcal{D}_i$ for each $\mathcal{P}_i$ (with the final path answer: $\mathcal{A}_i$ and intermediate step answers: $\{a_j\}^{(i)}_{j\in[1,M]}$):

$$\mathcal{D}_i = \underbrace{\alpha\frac{n_i}{N}}_{path-level} + \underbrace{(1-\alpha)\frac{m_i}{M}}_{step-level} \tag{1}$$

where $n_i \in [1, N]$ is the frequency of $\mathcal{A}_i$ existing in $\mathbb{A}$, $m_i \in [0, M]$ is the number of correct intermediate steps in $\mathcal{P}_i$, and α is a hyper-parameter. *The final answer is* $\mathcal{A}_{i^*}$ *satisfying* $i^* = \arg\max_{i\in[1,N]}(\mathcal{D}_i)$.

To better analyze the effects of varying α in the unified framework, we then define particular choices for α which we call *step and path level dominant answer calibration*.

Definition 1. ***Step-Level Dominant Answer Calibration****: This choice refers to the level of α at which the step-level score is used as the dominant criterion, with the path-level score given much smaller weight and only serving to break ties when necessary. Specifically, larger m_i always results in larger $\mathcal{D}_i$, no matter how small n_i is. We denote this as: $\forall n_j, n_k \in [1, N]$ and $m_j, m_k \in$*

$[0, M]$, where $n_j < n_k$ and $m_j > m_k$, the scores D_j and D_k should satisfy

$$\alpha\frac{n_j}{N} + (1-\alpha)\frac{m_j}{M} > \alpha\frac{n_k}{N} + (1-\alpha)\frac{m_k}{M}$$

Thus we can obtain

$$\alpha < \frac{1}{\frac{M(n_k - n_j)}{N(m_j - m_k)} + 1} \tag{2}$$

If Eq (2) is constant, we can infer that

$$\alpha < \min\left(\frac{1}{\frac{M(n_k - n_j)}{N(m_j - m_k)} + 1}\right) = \frac{1}{\frac{M\max(n_k - n_j)}{N\min(m_j - m_k)} + 1} \tag{3}$$

As $1 \leq n_j < n_k, n_j + n_k \leq N$, and $0 \leq m_k < m_j$, we can deduce that $\min(m_j - m_k) = 1$, $\max(n_k - n_j) = N - 2$. From the above, we deduce:

$$\alpha < \frac{1}{\frac{M(N-2)}{N} + 1} \tag{4}$$

Definition 2. ***Path-Level Dominant Answer Calibration**: For this choice, $\mathcal{D}_i$ gives priority to the path-level score, with the step-level score given much smaller weight and only serving to break ties when necessary. Concretely, larger n_i always conduces larger $\mathcal{D}_i$, no matter how small m_i is. We denote this as: $\forall n_j, n_k \in [1, N]$ and $m_j, m_k \in [0, M]$, where $n_j > n_k$ and $m_j < m_k$, the scores D_j and D_k should satisfy*

$$\alpha\frac{n_j}{N} + (1-\alpha)\frac{m_j}{M} > \alpha\frac{n_k}{N} + (1-\alpha)\frac{m_k}{M}$$

Analogously, we can obtain

$$\alpha > \frac{1}{\frac{M(n_j - n_k)}{N(m_k - m_j)} + 1} \tag{5}$$

If Eq (5) is constant, we can infer that

$$\alpha > \max\left(\frac{1}{\frac{M(n_j - n_k)}{N(m_k - m_j)} + 1}\right) = \frac{1}{\frac{M\min(n_j - n_k)}{N\max(m_k - m_j)} + 1} \tag{6}$$

As $1 \leq n_k < n_j$, and $0 \leq m_j < m_k \leq M$, we deduce that $\min(n_j - n_k) = 1$, $\max(m_k - m_j) = M - 0 = M$. From the above, we deduce:

$$\alpha > \frac{1}{\frac{1}{N} + 1} \tag{7}$$

In general, considering *step-level and path-level answer calibration dominance*, we can obtain two thresholds: $\frac{1}{\frac{M(N-2)}{N}+1}$ and $\frac{1}{\frac{1}{N}+1}$. Note that **$\boldsymbol{\alpha = 0}$ and $\boldsymbol{\alpha = 1}$ are respectively equivalent to the self-verification and self-consistency strategies**.

3.3 Evaluation of Answer Calibration

Calculation of `ROSCOE` Scores. In addition to the classical evaluation metric: Accuracy, Golovneva et al. (2023) have proposed **`ROSCOE`**, a suite of metrics for multi-step reasoning, under four perspectives: semantic alignment (`ROSCOE-SA`), semantic similarity (`ROSCOE-SS`), logical inference, and (`ROSCOE-LI`) and language coherence (`ROSCOE-LC`). Due to space limits, we select some representative scores from `ROSCOE` as evaluation metrics in the experiments.

Given source ground truth rationale ($\boldsymbol{s}$) and generated rationale ($\boldsymbol{h}$) with multiple steps (h_i), we calculate five scores (*All scores satisfy the principle that larger is better*):

(1) Faithfulness$_{step}$ ($\boldsymbol{h} \rightarrow \boldsymbol{s}$): To assess whether the model misconstrues the problem statement, or if the reasoning path is too nebulous, irrelevant, or improperly employs input information.

$$\textstyle\sum_{i=1}^{N} r\text{-align}(h_i \rightarrow \boldsymbol{s})/N \tag{8}$$

where N is the number of steps and r-align is used to measure how well $h_i \in \boldsymbol{h}$ can be aligned with any one of the steps in the ground truth path $\boldsymbol{s}$.

(2) Informativeness$_{path}$ ($\boldsymbol{h} \rightarrow \boldsymbol{s}$): To measure the level of concordance between the generated path and the source, and if the generated reasoning path is well-grounded with respect to the source.

$$[1 + \cos(\boldsymbol{h}, \boldsymbol{s})]/2 \tag{9}$$

where $\cos(\cdot, \cdot)$ is a function for cosine similarity.

(3) Consistency$_{steps}$ ($h_i \leftrightarrow h_j$): To measure logical entailment errors *within* the reasoning steps.

$$1 - \max_{i=2..N} \max_{j<i} p_{\text{contr}}(h_i, h_j) \tag{10}$$

where p_{contr} is used to assess the likelihood of step pairs contradicting each other. $h_i \in \boldsymbol{h}$ and $h_j \in \boldsymbol{h}$.

(4) Consistency$_{path}$ ($\boldsymbol{h} \leftrightarrow \boldsymbol{s}$): To evaluate mistakes in logical entailment between the generated reasoning path $\boldsymbol{h}$ and source context $\boldsymbol{s}$:

$$1 - \max_{i=1..N} \max_{j=1..T}\ p_{\text{contr}}(h_i, s_j) \tag{11}$$

where p_{contr} is the likelihood of source and generated steps contradicting each other. $s_j \in \boldsymbol{s}$; $h_i \in \boldsymbol{h}$.

(5) Perplexity$_{path}$ ($\boldsymbol{h}$): As an indicator of language coherence, it calculates average perplexity of all tokens in the generated reasoning path steps.

$$1/\text{PPL}(\boldsymbol{h}) \tag{12}$$

where PPL denotes the perplexity.

Task	Method	Accuracy ↑	Faithfulness ↑ (Over Steps)	Informativeness ↑ (Over Path)	Consistency ↑ (Within Steps)	Consistency ↑ (Within I/O)	Perplexity ↑ (Over Path)
GSM8K	CoT	80.21	88.73	96.38	**97.94**	96.94	9.14
	CoT + SV	82.34 (+2.13)	86.22 (−2.51)	95.19 (−1.19)	96.78 (−1.16)	93.46 (−3.48)	**14.90** (+5.76)
	CoT + SC	**87.11** (+6.90)	**88.83** (+0.10)	**96.40** (+0.02)∼	97.90 (−0.04)∼	**97.44** (+0.50)	8.90 (−0.24)
	ZS CoT	62.85	86.58	95.61	<u>97.30</u>	93.07	<u>15.67</u>
	ZS CoT + SV	67.70 (+4.85)	86.24 (−0.34)	95.19 (−0.42)	96.78 (−0.52)	93.44 (+0.37)	14.90 (−0.77)
	ZS CoT + SC	<u>71.42</u> (+11.14)	<u>86.70</u> (+0.12)	<u>95.67</u> (+0.06)∼	97.19 (−0.11)	<u>94.57</u> (+1.50)	14.95 (−0.72)
SVAMP	CoT	78.20	**87.73**	**95.74**	30.57	9.82	**6.65**
	CoT + SV	**85.80** (+7.60)	87.26 (−0.47)	95.00 (−0.74)	33.39 (+2.82)	**10.41** (+0.59)	6.23 (−0.42)
	CoT + SC	84.40 (+6.20)	87.60 (−0.13)	95.71 (−0.03)	**33.51** (+2.94)	9.92 (+0.10)	6.22 (−0.43)
	ZS CoT	72.80	<u>87.46</u>	95.77	31.71	18.39	<u>11.93</u>
	ZS CoT + SV	81.20 (+8.40)	86.92 (−0.54)	95.05 (−0.72)	<u>35.27</u> (+3.56)	<u>20.24</u> (+1.85)	11.44 (−0.49)
	ZS CoT + SC	<u>82.00</u> (+9.20)	87.40 (−0.06)	<u>95.81</u> (+0.04)∼	34.73 (+3.02)	19.67 (+1.28)	11.68 (−0.25)
MultiArith	CoT	97.67	**88.53**	**94.91**	7.77	7.47	5.51
	CoT + SV	**98.33** (+0.66)	88.36 (−0.17)	94.38 (−0.53)	**46.59** (+38.82)	**24.56** (+17.09)	**10.54** (+5.03)
	CoT + SC	98.17 (+0.50)	88.42 (−0.11)	94.82 (−0.09)	10.22 (+2.45)	9.29 (+1.82)	5.33 (−0.18)
	ZS CoT	87.00	<u>89.32</u>	95.30	<u>47.54</u>	24.39	<u>10.75</u>
	ZS CoT + SV	<u>97.00</u> (+10.00)	88.35 (−0.97)	94.38 (−0.92)	46.26 (−1.28)	<u>24.58</u> (+0.19)	10.54 (−0.21)
	ZS CoT + SC	<u>97.00</u> (+10.00)	89.18 (−0.14)	<u>95.32</u> (+0.02)∼	47.42 (−0.12)	23.83 (−0.56)	10.63 (−0.12)
MathQA	CoT	52.83	**85.99**	**95.31**	49.57	23.78	**7.64**
	CoT + SV	**54.74** (+1.91)	85.93 (−0.06)	95.24 (−0.07)	51.39 (+1.82)	24.61 (+0.83)	7.18 (−0.46)
	CoT + SC	54.47 (+1.64)	85.93 (−0.06)	95.20 (−0.11)	**51.73** (+2.16)	**25.03** (+1.25)	7.15 (−0.49)
	ZS CoT	49.45	85.20	<u>96.08</u>	23.50	13.76	13.44
	ZS CoT + SV	<u>52.86</u> (+3.41)	<u>85.93</u> (+0.73)	95.24 (−0.84)	<u>51.40</u> (+27.90)	<u>24.63</u> (+10.87)	7.19 (−6.25)
	ZS CoT + SC	49.51 (+0.06)	85.22 (+0.02)∼	96.08 (−0.00)∼	23.66 (+0.16)	13.79 (+0.03)	<u>13.48</u> (+0.04)∼
CSQA	CoT	74.77	81.40	92.57	**95.57**	**57.54**	2.46
	CoT + SV	74.04 (−0.73)	80.89 (−0.51)	92.10 (−0.47)	92.77 (−2.80)	56.05 (−1.49)	**2.47** (+0.01)∼
	CoT + SC	**75.27** (+0.50)	**81.50** (+0.10)	**92.71** (+0.14)	95.04 (−0.53)	56.97 (−0.57)	2.43 (−0.03)
	ZS CoT	67.57	<u>79.77</u>	<u>95.26</u>	<u>25.81</u>	29.17	<u>9.90</u>
	ZS CoT + SV	66.42 (−1.15)	79.06 (−0.71)	94.65 (−0.61)	25.36 (−0.45)	28.56 (−0.61)	9.06 (−0.84)
	ZS CoT + SC	<u>71.58</u> (+4.01)	79.51 (−0.26)	95.21 (−0.05)∼	25.08 (−0.73)	<u>29.69</u> (+0.52)	8.96 (−0.94)

Table 1: Comprehensive performance (%) with different strategies on GPT-3.5 (`gpt-3.5-turbo`). **CoT**: Few-shot CoT (Wei et al., 2022) with complex-prompting (Fu et al., 2023); **ZS-CoT**: Zero-Shot CoT (Kojima et al., 2022); **SV**: Self-Verification (Weng et al., 2023); **SC**: Self-Consistency (Wang et al., 2023i). **Best few-shot results** are marked in **bold**; <u>best *zero-shot* results</u> are <u>underlined</u>. I/O: input/output. ↑: larger is better. ∼, ∼: comparable.

4 Experiments

4.1 Setup

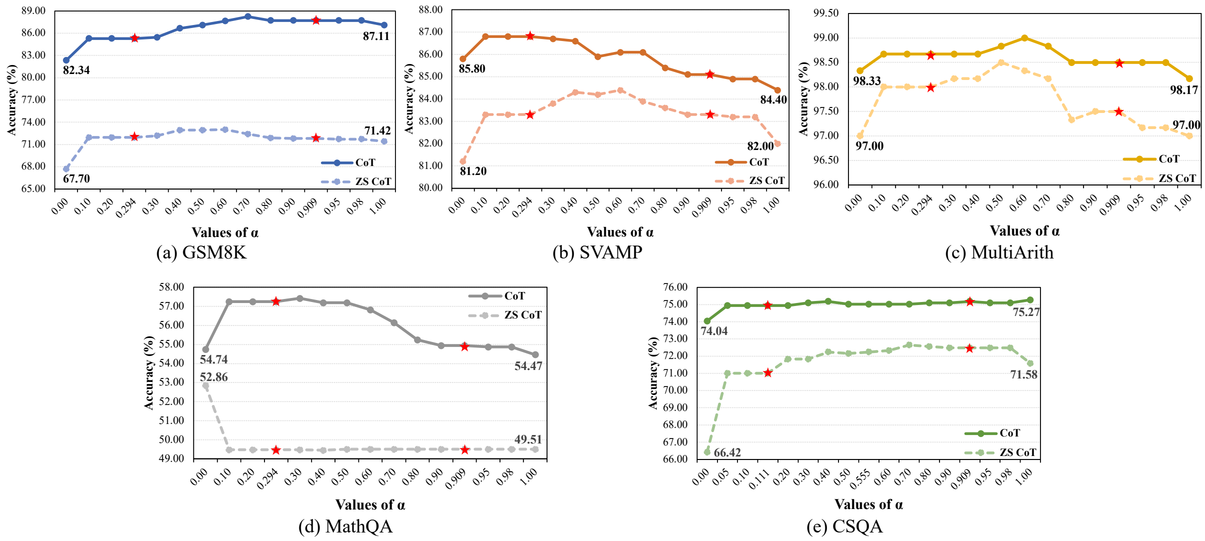
Evaluation Metrics. In this paper, we aim to conduct comprehensive evaluation on multi-step reasoning, thus we select some scores from `ROSCOE` (Golovneva et al., 2023) as introduced in §3.3, which contains a suite of metrics allowing us to evaluate the quality of reasoning rationales, not limited to the correctness of final answers.

Datasets. We evaluate on five benchmark datasets involving arithmetic and commonsense multi-step reasoning: **GSM8K** (Cobbe et al., 2021), **SVAMP** (Patel et al., 2021), **MultiArith** (Roy and Roth, 2015), **MathQA** (Amini et al., 2019) and **CSQA** (Talmor et al., 2019).

Models. For reasoning *path generation*, we leverage **Zero-shot CoT (ZS CoT)** (Kojima et al., 2022) and **Few-shot CoT (CoT)** (Wei et al., 2022) with complexity-based prompting (Fu et al., 2023). For *answer calibration*, we employ **Self-Verification (SV)** (Weng et al., 2023) and **Self-Consistency (SC)** (Wang et al., 2023i) on multiple paths. SV is a step-level strategy, which verifies intermediate-step answers and returns the path containing the maximum number of correct step answers. SC is a path-level strategy, which conducts majority voting on final answers of all generated paths and selects the most consistent result.

Implementation. We release the codes and generated results anonymously[1]. In this paper, the number of reasoning paths N defined in Eq (1) is 10, and number of intermediate steps M is 3 on all datasets except for CSQA where M is 10. We utilize GPT-3.5 (200B) with `gpt-3.5-turbo` engine as the backbone LLM to generate reasoning paths, and the temperature is set to 0.7. We also leverage GPT-4 (OpenAI, 2023) with `gpt-4` engine to generate ground-truth rationales given the ground-truth answers for all datasets excluding GSM8K (which already contains them). For evaluation referring to `ROSCOE` (Golovneva et al., 2023), we respectively lever-

[1] https://anonymous.4open.science/r/Eval_Multi-Step_Reasoning-4E60.



(a) GSM8K (b) SVAMP (c) MultiArith (d) MathQA (e) CSQA

Figure 2: Accuracy under different integrated *step-level* and *path-level* answer calibration strategies, varying with the values of α defined in Eq (1). Performance with two thresholds of $\frac{1}{\frac{M(N-2)}{N}+1}$ and $\frac{1}{\frac{1}{N}+1}$ are marked as ★.

age `all-MiniLM-L6-v2`/*SentenceTransformer*, and pretrained `gpt2-large` (Radford et al., 2019) to obtain token/sentence embedding and calculate perplexity defined in Eq (12). All the reasoning paths for CoT and ZS CoT were generated during 8th to 23rd June 2023, and answer calibration on the generated reasoning paths was conducted during 12th October to 8th November 2023.

4.2 Analysis on Step-Level and Path-Level Answer Calibration Strategies

We respectively incorporate the effective step-level and path-level answer calibration strategies, Self-Verification (SV) and Self-Consistency (SC), into CoT-based models operating on multiple paths. We evaluate their performance using six evaluation metrics, with the results presented in Table 1.

Generally, **in terms of *accuracy*, employing answer calibration is effective.** Seen from Table 1, we find that models equipped with SV and SC obviously outperform vanilla methods, as both few-shot and zero-shot CoT employing SV/SC achieve significant accuracy improvements on almost all tasks. Notably, zero-shot CoT with SV and SC achieves much more significant outperformance of accuracy than few-shot settings on almost all tasks, demonstrating that **answer calibration is more effective in zero-shot settings**. As zero-shot CoT is relatively challenging due to the absence of task-specific in-context learning, answer calibration strategies essentially creating a feedback loop where the model assesses its own performance and adjusts accordingly, could help to mitigate biases and overfitting to specific patterns during inference, allowing the model to better generalize to new types of problems and datasets.

Furthermore, **in terms of *other metrics*, answer calibration can improve *consistency* on arithmetic tasks but weakens *faithfulness*, *informativeness* and *perplexity* on both arithmetic and commonsense tasks.** Observed from Table 1, we find that SV and SC weaken the *perplexity* score, suggesting that the rationale generated from multiple paths is more complex than that from a single path with CoT models. However, these two strategies improve *consistency* scores on arithmetic tasks, intuitively benefiting from multiple paths. As SV verifies answers for intermediate steps and SC considers answers for all paths, they naturally enhance consistency within steps and between input/output (I/O). Additionally, SV and SC worsen *faithfulness* and *informativeness* on almost all tasks. The possible reason is that answer calibration on multiple paths focuses more on answer accuracy, while its increased complexity of its rationales tends to result in lower alignment and concordance between the source content and the output path. Generally, despite the benefits of employing SV and SC to CoT-based methods, the improvements are task-dependent and vary across different metrics.

4.3 Analysis on Integrated Answer Calibration Strategies

We then integrate step-level and path-level answer calibration strategies, varying α as defined in Eq (1). We present the accuracy of the integrated strategies in Figure 2. As observed, accuracy peaks at a specific value of α between

Engine	Strategy	GSM8K	SVAMP	MultiArith	CSQA
GPT-3 (175B) `code-davinci-001`	CoT	13.84	38.42	45.85	46.75
	CoT + SV	13.92↑	38.96↑	46.19↑	47.68↑
	CoT + SC	23.40⇑	54.58⇑	79.82⇑	54.92⇑
	CoT + SC + SV	23.59⇑	54.68⇑	80.01⇑	55.09⇑
Instruct-GPT (175B) `code-davinci-002`	CoT	60.81	75.87	96.13	77.42
	CoT + SV	65.14⇑	76.99↑	99.15⇑	77.83↑
	CoT + SC	78.00⇑	86.77⇑	100.00⇑	81.43⇑
	CoT + SC + SV	78.32⇑	86.94⇑	100.00⇑	81.53⇑
GPT-3.5 (200B) `gpt-3.5-turbo`	CoT	80.21	78.20	97.67	74.77
	CoT + SV	82.34↑	85.80⇑	98.33↑	74.04↓
	CoT + SC	87.11⇑	84.40⇑	98.17↑	75.27↑
	CoT + SC + SV	88.25⇑	86.80⇑	99.00↑	75.18↑

Table 2: Accuracy (%) with different backbone engines. ↑/⇑: slightly/significantly better; ↓: slightly worse than the baseline few-shot CoT. We refer to Weng et al. (2023) for results with GPT-3 and Instruct-GPT engines. As Weng et al. (2023) didn't test on MathQA dataset, we also exclude the results of MathQA here for fair comparisons.

the two thresholds defined in Eq (4) and (7) in almost all scenarios across all tasks, demonstrating that **optimal model performance should balance both step-level and path-level answer calibration dominance**. Besides, we notice that for "CoT on SVAMP task" in Figure 2(b) and "zero-shot CoT on MathQA task" Figure 2(d), employing integrated answer calibration strategies reaches a peak with α not between the two thresholds, and the overall performance remains stably lower than the initial best accuracy with $\alpha = 0$ (*i.e.*, SV). The possible reason may related to *employing SV* (i.e., $\alpha = 0$) *presenting more significant advantages than SC* (i.e., $\alpha = 1$) *in the two scenarios*. Specifically, CoT on SVAMP respectively achieves accuracy of 85.80% and 84.40% when α values 0 (SV) and 1 (SC), with the difference larger than 1%; Zero-shot CoT on MathQA employing SV and SC achieves accuracy of 52.86% v.s. 49.51%, where the difference is larger than 3%. Except for these two distinctive scenarios, others in Figure 2 obtain the optimal results by synthesizing step-level and path level answer calibration dominance.

In conclusion, the value of α plays a significant role in the performance of both few-shot and zero-shot CoT. Optimal ranges of α for each task are mostly between the two thresholds of step-level and path-level answer calibration dominance. The marked two thresholds represent boundaries for optimizing performance, which could guide further fine-tuning. Besides, the performance variance across datasets implies that the characteristics of each task, such as complexity, size, or the nature of the tasks. Models equipped with answer calibration strategies may require task-specific tuning to achieve the best performance.

4.4 Effects of Backbone Models

We compare accuracy on CoT-based answer calibration strategies with different LLM backbone engines, and present results in Table 2.

As observed from the results, for GPT-3 and Instruct-GPT, both self-verification (SV) and self-consistency (SC) provide consistent improvements; while on the larger GPT-3.5 model, their improvements are significantly weaker, particularly for SV, with which accuracy even slightly drops on the CSQA task. The possible reason is that GPT-3.5 is more prone to making mistakes when verifying on intermediate-step answers for multiple paths. Further, for integrated answer calibration strategies (SV+SC), the model's performance is close to the better one between SV and SC. Generally, path-level answer calibration is more advantageous than step-level one, with relatively higher accuracy and lower computation cost. We can infer that **answer calibration strategies, especially path-level self-consistency, provide benefits in many cases, particularly on less powerful LLMs**.

We further speculate, if the path generation for CoT with strong backbone LLM is sophisticated enough, the answer calibration may be simplified. We can directly conduct *path-level* answer calibration for multiple paths. But these findings cannot indicate that step-level answer calibration is meaningless for stronger backbone LLMs. As seen from Table 1, LLM equipped with step-level answer calibration is relatively beneficial to improve consistency scores. Besides, as mentioned in (Weng et al., 2023), step-level answer calibration can provide explainable answers by verifying on intermediate-step answers, making results more reliable.

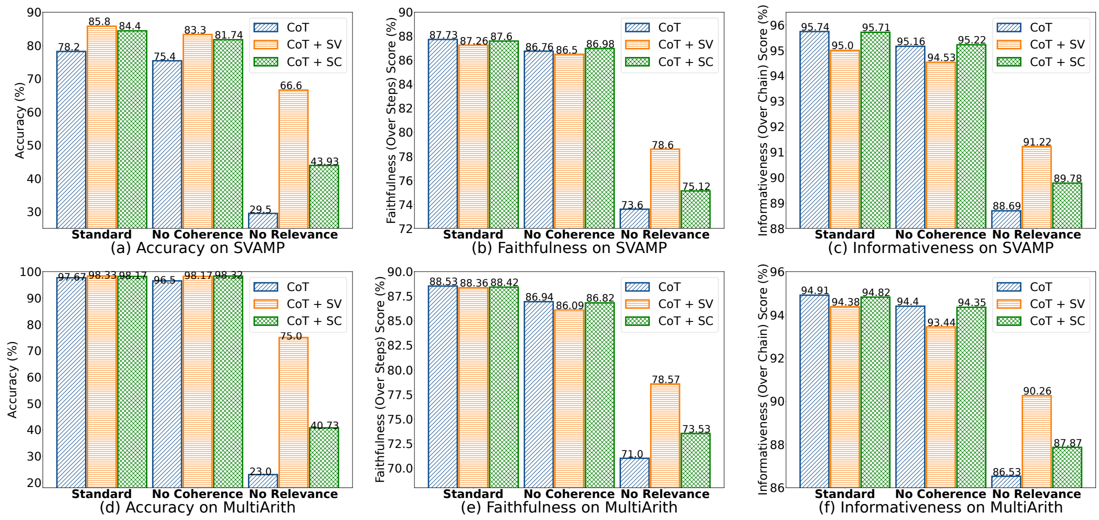


Figure 3: Performance (%) of "*Accuracy*, *Faithfulness (Over Steps)* and *Informativeness (Over Path)*" on SVAMP and MultiArith with different prompting on CoT models. We didn't show full results of other tasks for space limits.

4.5 Effects of Prompting

We further demonstrate the effects of prompting with few-shot demonstrations on answer calibration, evaluated on CoT models.

We respectively input prompts of *no coherence* and *no relevance* for few-shot CoT referring to Wang et al. (2023a) (examples are listed in Appendix A), and present performance on SVAMP and MultiArith in Figure 3. As seen, the deficiency of coherence and relevance in the prompting significantly weaken the performance of all models, with no relevance having a more profound impact than no coherence. In addition, CoT+SV achieves comparable performance with CoT+SC when prompting is standard or not coherent. Further, CoT+SV tends to perform significantly better than CoT+SC, when prompting with no relevance, indicating that step-level answer calibration strategy SV, is beneficial to maintain performance under adverse conditions. This observation suggests **the robustness of step-level answer calibration**. It also highlights **the potential benefits of step-level answer calibration strategies to mitigate performance degeneration caused by poor prompting**. The possible reason is that step-level answer calibration strategies break down the task into subtasks, and these subtasks are simple enough so that less likely to be influenced by the low-quality prompts.

4.6 Analysis on Tasks

As seen from Table 1,2, and Figure 2, generally, **SV and SC present more significant outperformance on arithmetic tasks than on the commonsense task (CSQA)**. Further, for CSQA, employing answer calibration tends to worsen the consistency scores, which is contrary to the trend observed in arithmetic tasks. The possible explanation lies in the characteristics of each task, such as complexity, size, or the nature of the tasks. In the CSQA task, correct intermediate steps may not always contribute to a coherent reasoning path due to potential irrelevance and redundancy. Specifically, even if we calibrate both intermediate step and path answers, there can be some correct commonsense statements while irrelevant to the question, resulting in worse consistency and perplexity. Conversely, in arithmetic tasks, correct intermediate answers almost guarantee a consistent reasoning path, as all intermediate answers are necessary and will contribute to a correct final answer.

5 Conclusion and Future Work

In this paper, we dissect multi-step reasoning into path generation and answer calibration, and provide a unified view of answer calibration strategies through a comprehensive evaluation. We find that path-level answer calibration is particularly potent in improving accuracy, while step-level answer calibration is more suitable for addressing issues related to low-quality prompting. The improvement is more pronounced in zero-shot scenarios and less significant on stronger backbone models. We also define step-level and path-level answer calibration dominance with two thresholds, and propose to integrate of the two types of strategies, which is promising to achieve optimal performance. Our findings suggest that answer calibration is a versatile strategy that can be integrated into various models to bolster multi-step reasoning capabilities of LLMs. In the future, we aim to develop more sophisticated multi-step reasoning models, drawing on the insights and conclusions from this study.

Limitations

The main limitation for this paper is that we didn't analyze more answer calibration strategies, such as step-/path-level methods on the single path, and varying the numbers of steps and paths in the integrated answer calibration strategies. Besides, we can also employ answer calibration strategies to other path generation models, not limited to CoT-based methods. Further, we should also evaluate answer calibration strategies on more tasks to make the results more sufficient.

References

Aman Madaan Pranjal Aggarwal and Mausam Yiming Yang. 2023. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with llms. In *EMNLP*, pages 12375–12396. Association for Computational Linguistics.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *NAACL-HLT (1)*, pages 2357–2367. Association for Computational Linguistics.

Jinheon Baek, Soyeong Jeong, Minki Kang, Jong Park, and Sung Ju Hwang. 2023. Knowledge-augmented language model verification. In *EMNLP*, pages 1720–1736. Association for Computational Linguistics.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2023. Graph of thoughts: Solving elaborate problems with large language models. *CoRR*, abs/2308.09687.

Zhen Bi, Ningyu Zhang, Yinuo Jiang, Shumin Deng, Guozhou Zheng, and Huajun Chen. 2024. When do program-of-thoughts work for reasoning? In *AAAI*. AAAI Press.

Lang Cao. 2023. Enhancing reasoning capabilities of large language models: A graph-based verification approach. *CoRR*, abs/2308.09267.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *ICLR*. OpenReview.net.

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

Olga Golovneva, Moya Peng Chen, S pencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. ROSCOE: A suite of metrics for scoring step-by-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *EMNLP*, pages 8154–8173. Association for Computational Linguistics.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *ACL (1)*, pages 14852–14882. Association for Computational Linguistics.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *CoRR*, abs/2210.11610.

Song Jiang, Zahra Shakeri, Aaron Chan, Maziar Sanjabi, Hamed Firooz, Yinglong Xia, Bugra Akyildiz, Yizhou Sun, Jinchao Li, Qifan Wang, and Asli Celikyilmaz. 2023a. Resprompt: Residual connection prompting advances multi-step reasoning in large language models. *CoRR*, abs/2310.04743.

Weisen Jiang, Han Shi, Longhui Yu, Zhengying Liu, Yu Zhang, Zhenguo Li, and James T. Kwok. 2023b. Forward-backward reasoning in large language models for verification. *CoRR*, abs/2308.07758.

Zhanming Jie and Wei Lu. 2023. Leveraging training data in few-shot prompting for numerical reasoning. In *ACL (Findings)*, pages 10518–10526. Association for Computational Linguistics.

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

Ziqi Jin and Wei Lu. 2023. Tab-cot: Zero-shot tabular chain of thought. In *ACL (Findings)*, pages 10259–10277. 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*.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xin Zhou. 2023. Better zero-shot reasoning with role-play prompting. *CoRR*, abs/2308.07702.

Jack Lanchantin, Shubham Toshniwal, Jason Weston, Arthur Szlam, and Sainbayar Sukhbaatar. 2023. Learning to reason and memorize with self-notes. *CoRR*, abs/2305.00833.

Soochan Lee and Gunhee Kim. 2023. Recursion of thought: A divide-and-conquer approach to multi-context reasoning with language models. In *ACL (Findings)*, pages 623–658. Association for Computational Linguistics.

IokTong Lei and Zhidong Deng. 2023. Selfzcot: a self-prompt zero-shot cot from semantic-level to code-level for a better utilization of llms. *CoRR*, abs/2305.11461.

Xiaonan Li and Xipeng Qiu. 2023. Mot: Memory-of-thought enables chatgpt to self-improve. In *EMNLP*, pages 6354–6374. Association for Computational Linguistics.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Lidong Bing, Shafiq R. Joty, and Soujanya Poria. 2023a. Chain of knowledge: A framework for grounding large language models with structured knowledge bases. *CoRR*, abs/2305.13269.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023b. Making language models better reasoners with step-aware verifier. In *ACL (1)*, pages 5315–5333. Association for Computational Linguistics.

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

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *CoRR*, abs/2305.20050.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. In *NeurIPS*.

Hao Liu, Carmelo Sferrazza, and Pieter Abbeel. 2023. Chain of hindsight aligns language models with feedback. *CoRR*, abs/2302.02676.

Jieyi Long. 2023. Large language model guided tree-of-thought. *CoRR*, abs/2305.08291.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *CoRR*, abs/2303.17651.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2023. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *CoRR*, abs/2308.00436.

Michalis Mountantonakis and Yannis Tzitzikas. 2023. Using multiple RDF knowledge graphs for enriching chatgpt responses. In *ECML/PKDD (7)*, volume 14175 of *Lecture Notes in Computer Science*, pages 324–329. Springer.

OpenAI. 2023. GPT-4 technical report. *OpenAI*.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *CoRR*, abs/2308.03188.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *NAACL-HLT*, pages 2080–2094. Association for Computational Linguistics.

Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. 2023. REFINER: reasoning feedback on intermediate representations. *CoRR*, abs/2304.01904.

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 *ACL (1)*, pages 5368–5393. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *EMNLP*, pages 1743–1752. The Association for Computational Linguistics.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Lu Wang, Ruoxi Jia, and Ming Jin. 2023. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *CoRR*, abs/2308.10379.

Fobo Shi, Peijun Qing, Dong Yang, Nan Wang, Youbo Lei, Haonan Lu, and Xiaodong Lin. 2023. Prompt space optimizing few-shot reasoning success with large language models. *CoRR*, abs/2306.03799.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*.

Simeng Sun, Yang Liu, Shuohang Wang, Chenguang Zhu, and Mohit Iyyer. 2023. PEARL: prompting large language models to plan and execute actions over long documents. *CoRR*, abs/2305.14564.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL-HLT (1)*, pages 4149–4158. Association for Computational Linguistics.

Xingchen Wan, Ruoxi Sun, Hanjun Dai, Sercan Ö. Arik, and Tomas Pfister. 2023. Better zero-shot reasoning with self-adaptive prompting. In *ACL (Findings)*, pages 3493–3514. Association for Computational Linguistics.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023a. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *ACL (1)*, pages 2717–2739. Association for Computational Linguistics.

Jianing Wang, Qiushi Sun, Nuo Chen, Xiang Li, and Ming Gao. 2023b. Boosting language models reasoning with chain-of-knowledge prompting. *CoRR*, abs/2306.06427.

Jinyuan Wang, Junlong Li, and Hai Zhao. 2023c. Self-prompted chain-of-thought on large language models for open-domain multi-hop reasoning. In *EMNLP (Findings)*, pages 2717–2731. Association for Computational Linguistics.

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

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023e. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *CoRR*, abs/2308.13259.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. 2023f. A survey on large language model based autonomous agents. *CoRR*, abs/2308.11432.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023g. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *ACL (1)*, pages 2609–2634. Association for Computational Linguistics.

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. 2023h. SCOTT: self-consistent chain-of-thought distillation. In *ACL (1)*, pages 5546–5558. Association for Computational Linguistics.

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

Yiming Wang, Zhuosheng Zhang, and Rui Wang. 2023j. Meta-reasoning: Semantics-symbol deconstruction for large language models. *CoRR*, abs/2306.17820.

Zekun Wang, Ge Zhang, Kexin Yang, Ning Shi, Wangchunshu Zhou, Shaochun Hao, Guangzheng Xiong, Yizhi Li, Mong Yuan Sim, Xiuying Chen, Qingqing Zhu, Zhenzhu Yang, Adam Nik, Qi Liu, Chenghua Lin, Shi Wang, Ruibo Liu, Wenhu Chen, Ke Xu, Dayiheng Liu, Yike Guo, and Jie Fu. 2023k. Interactive natural language processing. *CoRR*, abs/2305.13246.

Zhaoyang Wang, Shaohan Huang, Yuxuan Liu, Jiahai Wang, Minghui Song, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. 2023l. Democratizing reasoning ability: Tailored learning from large language model. In *EMNLP*, pages 1948–1966. Association for Computational Linguistics.

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

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *EMNLP (Findings)*, pages 2550–2575. Association for Computational Linguistics.

Zhenyu Wu, Meng Jiang, and Chao Shen. 2024. Get an a in math: Progressive rectification prompting. In *AAAI*. AAAI Press.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huan, and Tao Gui. 2023a. The rise and potential of large language model based agents: A survey. *CoRR*, abs/2309.07864.

Zhiheng Xi, Senjie Jin, Yuhao Zhou, Rui Zheng, Songyang Gao, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023b. Self-polish: Enhance reasoning in large language models via problem refinement. *CoRR*, abs/2305.14497.

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

Tianci Xue, Ziqi Wang, Zhenhailong Wang, Chi Han, Pengfei Yu, and Heng Ji. 2023. RCOT: detecting and rectifying factual inconsistency in reasoning by reversing chain-of-thought. *CoRR*, abs/2305.11499.

Fanglong Yao, Changyuan Tian, Jintao Liu, Zequn Zhang, Qing Liu, Li Jin, Shuchao Li, Xiaoyu Li, and Xian Sun. 2023a. Thinking like an expert: Multimodal hypergraph-of-thought (hot) reasoning to boost foundation modals. *CoRR*, abs/2308.06207.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023b. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023c. React: Synergizing reasoning and acting in language models. In *ICLR*. OpenReview.net.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. 2023d. Retroformer: Retrospective large language agents with policy gradient optimization. *CoRR*, abs/2308.02151.

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

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. In *EMNLP*, pages 5942–5966. Association for Computational Linguistics.

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

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STar: Bootstrapping reasoning with reasoning. In *NeurIPS*.

Mengxue Zhang, Zichao Wang, Zhichao Yang, Weiqi Feng, and Andrew S. Lan. 2023. Interpretable math word problem solution generation via step-by-step planning. In *ACL (1)*, pages 6858–6877. Association for Computational Linguistics.

Ruochen Zhao, Xingxuan Li, Shafiq R. Joty, Chengwei Qin, and Lidong Bing. 2023a. Verify-and-edit: A knowledge-enhanced chain-of-thought framework. In *ACL (1)*, pages 5823–5840. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023b. A survey of large language models. *CoRR*, abs/2303.18223.

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.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2023. Solving challenging math word problems using GPT-4 code interpreter with code-based self-verification. *CoRR*, abs/2308.07921.

Appendices

A Cases of Low-Quality Prompts

We list some examples of prompts in Table 3.

Prompt Setting	***Example Query (Arithmetic Reasoning)*** *Leah had 32 chocolates and her sister had 42.* *If they ate 35,* *how many pieces do they have left in total?*
Standard CoT	Originally, Leah had 32 chocolates and her sister had 42. So in total they had 32 + 42 = 74. After eating 35, they had 74 - 35 = 39 pieces left in total. The answer is 39.
No Coherence	After eating 32 + 42 = 74, they had 32 pieces left in total. Originally, Leah had 74 - 35 = 39 chocolates and her sister had 35. So in total they had 42. The answer is 39.
No Relevance	Patricia needs to donate 19 inches, and wants her hair to be 31 inches long after the donation. Her hair is 29 inches long currently. Her hair needs to be 19 + 31 = 50 inc long when she cuts it. So she needs to grow 50 - 29 = 21 more inches. The answer is 21.

Table 3: Examples of prompts (standard, no coherence and no relevance) in our experiments.