IJHSR
ijhighschoolresearch.org
RESEARCH ARTICLE

The Effects of Chain-of-Thought Prompting on the Performance of Moderate-sized LLMs in Competitive Math

Eric L. Min

Westwood High School, 12400 Mellow Meadow Dr, Austin, TX, 78750, USA, eric.lin.min7@gmail.com

ABSTRACT: Modern state-of-the-art large language models (LLMs), such as Deepseek V3 and GPT-4o, demonstrate significantly better performance on a variety of reasoning-related tasks when compared with small-scale (SLMs) and moderate-sized LLMs (MSLMs). *Chain-of-Thought* (CoT) prompting, where models are guided towards generating intermediate reasoning steps, has been shown to improve performance in LLMs. It produces notable gains across arithmetic, commonsense, and symbolic benchmarks, such as GSM8K. However, the effects of CoT prompting in MSLMs, particularly more recent models, remain unclear. This is often due to high degrees of prompt sensitivity, and thus, we explore the impact of CoT prompting in MSLMs, evaluating the degree to which encouraging these relatively smaller models to "think out loud" can enhance a model's rigorous task performance. In this paper, we evaluate the abilities of CoT-prompted MSLMs in the reasoning-reliant field of mathematics, particularly competitive mathematics, contrasting these results with zero-shot baselines and prompt variants to produce conclusions. The empirical gains reveal striking connections between prompt architecture and MSLM capacity. For example, the Llama-4-Maverick model exhibited signs of inherent internal prompting, supporting the result that the model performed the best when prompted with a specific example with a high degree of task alignment.

KEYWORDS: Robotics and Intelligent Machines, Cognitive Systems, Large Language Model, Chain of Thought (CoT), Competitive Mathematics.

Introduction

LLMs have achieved remarkable fluency in natural language tasks.[1,2] However, despite their apparent proficiency, these models are not actually capable of reasoning in the human sense.[3] LLMs fundamentally lack true reasoning because they operate as sophisticated pattern matchers, essentially stochastic parrots, that are trained via next token prediction on large-scale text collections, optimizing for the statistical likelihood of word sequences rather than the derivation of logically coherent and logic-driven conclusions.[4]

LLMs are evolving from static prompt response systems to dynamic cognitive agents, and progress hinges on structured prompting (*Chain-of-Thought, Tree-of-Thought, Forest-of-Thought*),[5,6] and tool usage (*ReAct, PAL*).[7] CoT functions by attempting to guide language models towards generating intermediate reasoning steps before answering; however, these linear reasoning chains are fragile with error propagation. Tree-of-Thought explores a branching space of thought hypotheses, generating and evaluating multiple reasoning chains before backtracking to the most common answer. ReAct (Reasoning + Acting) and PAL (Program-aided language) integrate external results from external code, API calls, or tools mid-reasoning to enhance their performance.

CoT stands out as a strong paradigm in language model reasoning since it strategically structures how the model "thinks,"[2] often significantly improving accuracy, transparency, and task performance.[8] CoT prompting enables LLMs to tackle multi-step reasoning tasks with noticeably improved accuracy.[9] It has been demonstrated that integrating intermediate reasoning steps often dramatically enhances performance on arithmetic, commonsense, and symbolic benchmarks. By eliciting an explicit sequence of reasoning steps, CoT offers users visibility into the model's cognitive process. This transparency is invaluable for debugging and evaluating trustworthiness.

Recent advances in MSLMs and SLMs underscore their growing significance beyond niche tasks, offering a combination of efficiency, control, and domain specialization that makes them particularly suited for enterprise and edge deployment scenarios. These lightweight models typically require far fewer computational resources than large-scale models, enabling deployment on devices like smartphones or edge servers while delivering low-latency and cost-effective inference.[10] By training or fine-tuning MSLMs and SLMs on domain-specific datasets, such as healthcare,[11] legal documents, and finance, organizations can achieve higher contextual accuracy and reduce hallucination risk compared to general-purpose LLMs, which is critical in regulated industries. Moreover, because MSLMs and SLMs can be deployed entirely on-device or within a secure network, they address privacy and compliance concerns inherent to cloud-based models, ensuring full data control as well as facilitating alignment with regulations such as GDPR or HIPAA. Some further argue that SLMs are not only sufficiently powerful but also inherently more suitable and economically advantageous as the core of agentic AI systems, proposing a modular architecture where heterogeneous dialogue is required.[12] Contemporary research further advocates for these smaller-scale language models in building agentic AI systems - models that perform specialized, repetitive tasks efficiently - highlighting their tailored fit for large-scale modular AI frameworks where smaller models complement larger,

DOI: 10.36838/IJHSR87.20

more general ones. However, they are not without certain drawbacks, as these models exhibit limitations in generality and robustness, particularly in complex reasoning or out-of-domain scenarios, and may be more susceptible to security vulnerabilities like jailbreak attacks.[13] Collectively, these findings imply that while MSLMs and SLMs may not necessarily replace LLMs for deep reasoning tasks, they offer a strategically powerful, accessible, and responsible alternative for many real-world applications where scale, cost, and control play a significant role.

Recent work has shown that CoT prompting can substantially improve reasoning in very large models, with reported gains across arithmetic, commonsense, and symbolic reasoning tasks.[2] These results support the idea that certain reasoning abilities are emergent, appearing once models exceed a sufficient scale. However, evidence also indicates that CoT may be ineffective, or even harmful, for smaller models, with accuracy drops of 15-30% observed on reasoning benchmarks.[14] Research on smaller and moderate-sized models has largely prioritized efficiency, domain specialization, and deployment constraints. While valuable, this focus has a significant shortcoming: it often treats reasoning capability as a secondary concern or an input for distillation, rather than systematically evaluating how in-context prompting strategies like CoT interact with these more constrained architectures.

This gap between well-established CoT gains in very large LLMs and the lack of systematic evidence for moderate-sized models frames our central question: can CoT prompting reliably enhance reasoning in MSLMs, and if so, to what extent and in which contexts?

In light of this, our work directly investigates the effectiveness of CoT prompting in moderate-sized and small language models. Specifically, we study CoT-augmented reasoning in Phi-4, Gemma-3-27b-it, and Llama 4 Maverick, evaluating their performance on competitive mathematical reasoning benchmarks relative to zero-shot baselines and internal prompting variations. This research aims to systematically establish the benefits of CoT for these resource-efficient architectures, moving beyond prior work that focuses predominantly on either very large LLMs or distilled small models. Through this analysis in multi-step, higher-level mathematics, we seek to illuminate how prompt architecture and model capacity in determining reasoning capability, thereby identifying when and how CoT prompting yields reliable improvements in models accessible to a wider range of users and applications.

Methods

In this paper, we evaluate the impact of various types of CoT prompts on the performance of MSLMs by drawing comparisons between the baseline performance (no prompting),[15] and the performance of the MSLMs when using 4 different CoT prompts to solve AMC12 competitive math questions.[16] Reasoning algorithms such as CoT are best evaluated with AMC12 math problems due to their multi-step structure, requiring chaining of operations (intermediate value calculations, applying formulas, reinterpretation), as well as their objective verifiability, since answers are binary and easy to check. AMC12 problems are a useful metric also due to their moderate difficulty and high discrimination, as the problems are neither trivial nor hyper-technical, and they challenge LLMs with stepwise logical and multi-concept problem-solving while maintaining strong discrimination across solver levels.[17] These questions offer distinct advantages over other benchmarks, serving as a good balance between the basic arithmetic of GSM8K math sets and the depth and challenge of greater difficulty benchmarks such as AIME or MATH that often make problems unsolvable by LLMs. AMC12 problems are also characterized by cognitive complexity beyond basic arithmetic, requiring parsing, deduction, and inference, skills central to deeper reasoning capabilities. Language models struggle with the precise symbol manipulation and algorithmic thinking needed for such problems, and CoT prompting helps bridge this symbolic-execution gap.

In our experiment, we use the MSLMs Gemma-3-27b-it, Llama-4-Maverick, and Phi-4, with the first two having been released 3 months prior to our experiment, and the latter being released 6 months prior, as shown in Appendix B. We classify these LLMs as MSLMs due to their model parameters, model families, and model pricing. According to IBM (2024), the families of these three models are all classified as small language models for their number of parameters used during computation. The three models also have significantly cheaper pricing than state-of-the-art LLMs, and LLMs such as Gemini 2.5 Pro have more than 8 times the price of Llama 4 Maverick, even with Llama 4 Maverick being the most expensive out of the three MSLMs. Despite being described as "lightweight,"[18] the three models have 27.4 billion, 17 billion, and 14.7 billion parameters, respectively, which all exceed the typically accepted model parameter upper bound of 14 billion when identifying SLMs.

Shown in Appendix A, we utilize 4 different CoT approaches, labeled CoT1, CoT2, CoT3, and CoT4. Both CoT1 and CoT2 are classified as one-shot CoT prompts, using a single handcrafted exemplar embedded with stepwise rationales. CoT1 possesses the longest input length in terms of both characters and words, followed by CoT2, CoT3, and CoT4. CoT1 provides a rigorous, expert-crafted solution that demonstrates the level of reasoning quality and structural clarity we aim to achieve. The remaining prompts were generated by the state-of-the-art LLM Deepseek-V3, allowing us to test whether AI-generated prompts can approximate this level of rigor while also enhancing replicability, reducing dependency on manual craftsmanship, and enabling scalable application cross-domains.

CoT1 utilized an ABA layout (introduction-example-task restated), starting with a general zero-shot prompt, then introducing an example question of great difficulty, solving it with stepwise rationales, before restating the zero-shot prompt and stating the problem to be solved. The question used for CoT1 was an AMC12 problem consisting of a combination of algebra, number theory, and combinatorics, three of the four major subjects in competitive mathematics. Geometry was not included, as multi-domain problems rarely span all four categories simultaneously, and geometric reasoning in particular

relies on diagrammatic and spatial intuition that text-based CoT prompting does not easily capture. The idea behind this prompt is for structured priming of expert reasoning, and this high-quality solution demonstrates domain-specific heuristics, logical sequencing, and mathematical formalisms. In particular, this example acts as a behavioral template, guiding the model toward producing output that mirrors the structure, clarity, and rigor expected in expert-level problem solving. The significance of this prompt is also highlighted by its alignment with the target problem distribution. The AMC12 contest is characterized by problems that require multi-step algebraic, number theoretic, combinatorial, and logical reasoning. Framing the prompt around this distribution helps ensure that the model applies reasoning strategies that are both appropriate in depth and transferable across a wide range of problems.

CoT2 provided a detailed framework for problem solving, consisting of problem decomposition, strategy selection, rigorous execution, and validation, and then applied this framework to an elementary computation math problem before stating certain critical rules and introducing the problem to be solved. This prompt emphasized expert emulation through role conditioning, as well as procedural decomposition of problem-solving. This structured prompt functions as a general-purpose reasoning blueprint for high-level mathematical problem solving. By combining role priming, procedural scaffolding, strategic heuristics, and validation mandates, it trains LLMs not only to solve problems accurately but also to explain and verify their work in a way that mirrors expert human mathematics.

Both CoT3 and CoT4 are classified as zero-shot CoT prompts, as neither introduces an example problem. CoT3 is greatly similar to CoT2 in idea, utilizing a similar 4-stage framework but being more concise and omitting the worked example. This streamlined, scaffold-based CoT prompt strikes a balance between structure and brevity, guiding LLMs through expert-level mathematical problem solving without the need for worked examples. By emphasizing decomposition, strategy selection, stepwise execution, and validation, it reinforces mathematically sound reasoning patterns while remaining general-purpose and token-efficient.

Finally, CoT4 utilized a short, general CoT prompt. This minimalistic prompt serves to guide LLMs to engage in basic Chain-of-Thought reasoning through lightweight instructions. It achieves broad applicability with minimal structure overhead. While it lacks the modular scaffolding of more elaborate prompts, its fluency, domain alignment, and efficiency make it suitable as a baseline prompt in symbolic math pipelines, especially when interpretability and prompt brevity are priorities.

These four prompts were chosen to vary along two key dimensions: exemplar inclusion and degree of structural scaffolding. CoT1, with an expert-crafted solution, represents an upper bound on the domain alignment. CoT2 and CoT3 explore the role of explicit procedural scaffolding with and without examples, respectively, while CoT4 provides a lightweight baseline for measuring the value of added structure. These dimensions are remarkable because they represent two of the most widely recognized and manipulable parameters of prompt design: whether to anchor the model with concrete demonstrations or leave it to infer strategies, and whether to provide explicit stepwise frameworks or rely on the model's internal reasoning habits. Together, this set lets us evaluate how exemplar priming and structural decomposition influence mathematical reasoning in MSLMs.

For our dataset, we use 3 years of AMC12 exam questions (2022,2023,2024), for a total of 150 questions, as two exams are released each year with 25 questions per exam. Since LLMs are probabilistic generators, each call can produce different reasoning or answers.[19,20] Thus, each question is attempted a total of 5 times to capture stochastic behavior, avoid outlier bias, and follow evaluation best practices. This helps to reflect true model variability, averaging out randomness, improving correctness via majority voting, and enhancing comparability and reliability. Therefore, we use three distinct MSLMs, a set of 5 prompts (including the baseline), an evaluation dataset of 150 questions, and 5 attempts per evaluation item, for a total of 11250 LLM calls. All data are available on GitHub upon request.

To analyze the results produced from these calls, we define four evaluation metrics: the Aggregate Attempt Accuracy Metric (AAA), Thresholded Problem-Solving Rate Metric (TSPR-2), Majority-Consensus Problem Accuracy Metric (MCPA), as well as a Challenging-Subset Aggregate Attempt Accuracy Metric (CSAAA). Together, these metrics form a hierarchy of evaluation, capturing raw potential, minimal understanding, robust reliability, and elite reasoning.

The AAA metric evaluates the total number of correct responses across all individual problem-solving attempts, calculated as $\sum_{i=1}^{n}(x_i)$ where x_i represents the number of correct responses for problem i, and n is the total number of problems. For example, a problem with 4 correct solves out of 5 would add 4 to the score. This measures overall performance across attempts, regardless of consistency, as even problems solved with low frequency earn credit.

The TSPR-2 metric applies a thresholding criterion: a problem is deemed "solved" if the model answers it correctly in at least 2 out of 5 attempts. TSPR-2 can be calculated as $\sum_{i=1}^{n} I(x_i \geq 2)$ where I represents the indicator function, evaluating to 1 when the condition is met, and 0 otherwise. For example, a question with 2 correct solves would add 1 to the score, while a question with 1 correct solve would not. This metric captures minimal reliability, identifying whether the model demonstrates some grasp of the problem-solving method, even if inconsistently, akin to recognizing early signs of understanding in student work.

MCPA is also a threshold metric, but with a stricter threshold: a problem is only deemed "solved" if the model answers it correctly in at least 3 out of 5 attempts. This can be expressed as $\sum_{i=1}^{n} I(x_i \geq 3)$. For example, a question with 3 correct solves would add 1 to the score, while a question with 2 correct solves would not. This metric captures robust mastery, requiring the model's reasoning to be dominant and repeatable across attempts, much like human grading standards that demand consistency before awarding full credit.

DOI: 10.36838/IJHSR87.20

CSAAA is a subset of the AAA metric, and it instead considers only the hardest questions in each exam. These are regarded as the "final fives", the last five questions of the AMC12, and these questions are widely recognized for their disproportionate difficulty relative to the earlier portions of the exam. These problems typically demand a significantly higher level of abstract reasoning, problem-solving creativity, and multi-step logic, distinguishing themselves from the more procedural or recall-based problems that appear earlier in the test. Since these problems often resist solution through brute-force computation or superficial pattern recognition, they serve as an effective benchmark for assessing whether a model is capable of reasoning in a structured and context-aware manner.

The following tables summarize results across these four metrics for different prompting strategies. The "Baseline" column reports raw values as fractions, where the numerator indicates the metric-observed score and the denominator indicates the maximum possible score under that metric. The subsequent columns (CoT1-CoT4) report percentage changes relative to the baseline, with positive values denoting improvement and negative values denoting deterioration. This presentation facilitates cross-metric comparison by normalizing results to relative deviations from a common reference point.

Results and Discussion

Below are the results of the Gemma-3-27b-it model when prompted with the four prompts, relative to a zero-shot baseline, as well as the baseline performance. A negative value would indicate a deterioration in performance.

Table 1: Gemma-3-27b-it evaluation dataset performance. CoT prompting leads to consistent improvements over the zero-shot baseline across all four evaluation metrics, with the strongest relative gains appearing in CoT1 and CoT4.

	Baseline	CoT1	CoT2	CoT3	CoT4
AAA	431/750	11.37%	6.26%	2.32%	9.28%
TSPR-2	95/150	14.74%	9.47%	6.32%	12.63%
MCPA	86/150	17.44%	12.79%	5.81%	6.98%
CSAAA	30/150	20.00%	20.00%	6.67%	33.33%

Below are the performances of the Llama-4-Maverick model when prompted with the four prompts, relative to a zero-shot baseline. The baseline performance is included as well.

Table 2: Llama-4-Maverick evaluation dataset performance. CoT prompting generally improves performance over the zero-shot baseline, with the strongest relative gains in CoT1 and moderate gains in CoT2.

	Baseline	CoT1	CoT2	CoT3	CoT4
AAA	426/750	7.75%	0.23%	-2.34%	-2.11%
TSPR-2	97/150	11.34%	3.09%	-4.12%	-3.09%
MCPA	87/150	1.15%	-5.75%	-2.30%	-2.30%
CSAAA	33/150	21.21%	21.21%	24.24%	12.12%

Below are the performances of the Phi-4 model when prompted with the four prompts, relative to a zero-shot baseline.

Table 3: Phi-4 evaluation dataset performance. Across the four evaluation metrics, CoT1-CoT3 generally reduce performance relative to the zero-shot baseline, indicating that these prompt styles do not transfer well to Phi-4. In contrast, CoT4 yields consistently positive gains, with especially large improvements on CSAAA.

	Baseline	CoT1	CoT2	CoT3	CoT4
AAA	340/750	0.29%	-7.94%	-7.65%	8.24%
TSPR-2	82/150	2.44%	-8.54%	-6.10%	8.54%
MCPA	69/150	-4.35%	-14.49%	-11.59%	8.70%
CSAAA	12/150	16.67%	8.33%	-25.00%	58.33%

Below is a chart of the AAA performances of the MSLMs when prompted with the prompt variants.

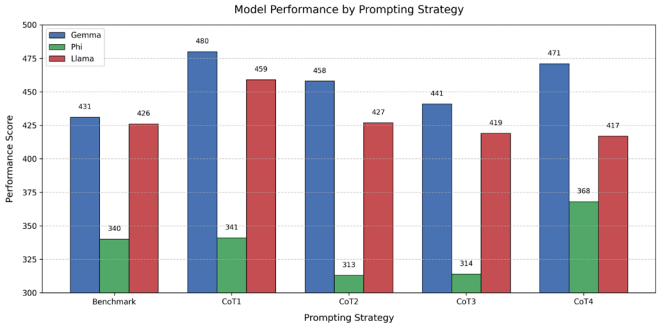

Figure 1: Model performance by prompting strategy. Note the truncated y-axis starting at a score of 300. CoT1 yields performance gains for all three models, while CoT4 produces especially large improvements for Phi-4 and strong gains for Gemma but a slight decrease for Llama relative to its baseline.

Under the zero-shot baseline, the Gemma-3-27b-it model could not produce labeled stepwise rationales. After introducing the four CoT prompts, it began doing so. This shift confirms that CoT prompting can induce reasoning behaviors not naturally present in the model's zero-shot outputs. From the results collected, shown in Table 1, it can be observed that both CoT1 and CoT4 generally led to better increases in performance than CoT2 and CoT3. These findings show that the quality and relevance of instructional structure directly influence the downstream. Despite its brevity, CoT4 resulted in a noticeable improvement, suggesting that even lightweight CoT prompts can elicit structured reasoning behavior within MSLMs. This result underscores the intrinsic latent capability of the Gemma-3-27b-it model to perform multi-step reasoning when nudged appropriately, and confirms findings from prior work on zero-shot prompting.[21]

The modest accuracy gains observed for CoT2 and especially CoT3 are a key finding. These two prompts often reduced accuracy compared with the benchmark (Tables 1-3). This degradation aligns with two well-documented failure modes in language models: hallucination and catastrophic forgetting. These phenomena highlight how mismatches in prompt structure or abstraction level can lead to degradation in reasoning quality. CoT2 presented a problem-solving framework along

with a generic arithmetic example, but this example lacked alignment with the structural and conceptual complexity of the AMC12 tasks. As a result, the model likely relied on superficial pattern matching, producing computationally valid but logically incorrect reasoning chains, a hallmark of hallucination in LLMs.[22]

An important characteristic of the benchmark performance of the Llama-4-Maverick model is the existence of labeled stepwise rationales without the aid of explicit prompting, suggesting the existence of a structured internal reasoning schema encoded during pretraining or fine-tuning, and this characteristic is shown in Appendix C. This behavior may indicate that the model has internalized procedural norms for explanation, possibly through exposure to datasets containing math problem solutions or CoT demonstrations.

This observed emergent behavior may reduce the model's responsiveness to certain types of external prompting, particularly when such prompts introduce incompatible or underspecified reasoning formats. In the case of CoT4, which used a minimal zero-shot prompt, it is likely the prompt failed to align with the model's pre-learned schema for rational generation. CoT4 was designed to elicit labeled stepwise rationales. However, this behavior was already present in the baseline results. This is supported by CoT4's noticeably worse performance when compared to CoT1, CoT2, and CoT3, shown in Table 2. While CoT3 used an AMC12-specific framework, this framework was generalized. As a result, rather than reinforcing the model's internal reasoning patterns, these prompts may have introduced prompt interference, where the model is forced to reconcile a vague or foreign framework with an already established reasoning process.[23] CoT2 likely also suffered from this interference, as the example and worked solution provided was a trivial arithmetic computation, and the framework included was also overgeneralizing. CoT2 was also much longer than CoT3 and CoT4. Although it seems intuitive that more context would improve performance, several recent studies reveal that excessively long prompts can degrade language model reasoning capabilities, even well before reaching their context window limits.[24,25] However, it can be seen that CoT1, CoT2, and CoT3 demonstrated similar degrees of improvement on the CSAAA metric, suggesting that the model potentially experienced reduced interference and improved alignment when attempting harder questions.

Despite the input length of CoT1 being significantly longer than that of the other prompts, the Llama-4-Maverick model demonstrated the best overall performance when prompted with CoT1. This is likely due to the high degree of alignment between the worked example and the evaluation dataset, as well as the domain-appropriate vocabulary.[26]

Figure 1 shows that the Phi-4 model performed notably worse overall. This suggests that minimal zero-shot prompting may align more closely with its internal capabilities than complex or structured alternatives. Consequently, specific or multi-step prompts such as CoT1, CoT2, and CoT3 may introduce format misalignment or overload for the MSLM, reducing coherence and increasing illogical or incomplete answers. This phenomenon mirrors findings in prompting sensitivity, as models with less robust conditioning capacities are more vulnerable to format-induced variability and may underperform when prompted with overly structured templates.[27] These ideas are all supported by the significant and notably better performance of the Phi-4 model when prompted with CoT4, as shown in Table 3.

Future Work

The reasoning capabilities within MSLMs can be advanced through frameworks similar to the following three, ensuring consistency, efficiency, and robustness. The first framework is blueprint guidance via LLM-generated templates, and this can ensure consistency,[28] as these blueprints provide structure for stepwise solutions, helping MSLMs systematically approach complex tasks without increasing model size.

Chain-of-Thought feeds into the blueprint process, and although CoT reveals the full reasoning chain, the blueprint framework distills it into a more abstract, MSLM-compatible scaffold, enabling consistent, structured guidance tailored to each model's capacity. Efficiency can be obtained through targeted scaffolding, providing necessary LLM hints during uncertain reasoning steps, leading to an accuracy boost with minimal overhead.[29] In this framework, the MSLM generates its own reasoning trajectory: $R = (r_1, r_2, r_3 \dots r_m)$ for a given prompt, each step r_i is evaluated. Flagged steps are replaced with alternative reasoning suggestions r_i' generated by an LLM: $r_i' \sim P_{LLM}(r_i' \mid Q, r_{<1})$, ensuring interventions are sparse and targeted, focusing on key decision points rather than full chain assistance. Finally, robust reasoning and self-verification can be promoted through a self-play framework,[30] where independent MSLMs generate and critique reasoning trajectories in tandem. Only reasoning trajectories that pass mutual agreement or validation are accepted, reducing noise and hallucination, and these methods don't rely on training data, making them scalable across tasks.

Causal inference within MSLMs and SLMs remains an emerging frontier, largely unexplored compared to its study in relatively larger LLMs, but it holds significant promise for advancing lightweight, domain-specialist AI systems. Recent surveys on causal inference and LLMs review how causal methodologies, such as causal discovery and treatment effect estimation,[31] can enrich causal inference. Potential future work includes adapting and evaluating MSLMs and SLMs on causal benchmarks such as Cladder and CRAB; developing domain-specific, smaller-scale agents that integrate structured knowledge graphs for causal discovery and treatment effect estimation; designing blueprint prompting strategies to scaffold causal reasoning in smaller models; and analyzing whether causal structure is internally represented within SLM and MSLM activations. Collectively, these directions promise to make causal reasoning accessible in lightweight, efficient models, expanding the utility of SLMs and MSLMs in high-stakes, resource-constrained, or privacy-preserving settings.

Conclusion

We observe that CoT prompting - guiding models to articulate intermediate reasoning steps - can yield notable per-

DOI: 10.36838/IJHSR87.20

formance gains in small and moderate-sized language models. Understanding a model's inherent capabilities, particularly its strengths, weaknesses, and latent biases, is crucial for crafting prompts that align well with its internal reasoning processes. Tailoring prompts effectively requires more than just clarity or examples; it relies on how compatible the prompt is with the model's built-in affordances. Capacity-aware prompt design involves diagnosing the model's strengths (e.g., numeral reasoning, understanding of text) and weaknesses (e.g., multi-step inference, nuanced instruction following) and then adjusting prompt structure accordingly. For instance, models at a smaller-scale or with lower computational affinity for complex inference may perform best with minimal zero-shot prompts (as demonstrated by the Phi-4 model, as well as, to a significantly less drastic degree, the Gemma-3-27b-it model) with short, clear instructions that avoid unnecessary examples or reasoning chains, while more capable models benefit from richer scaffolding (as demonstrated by both the Llama-4-Maverick model and the Gemma-3-27b-it model). Established benchmarks can be utilized to systematically identify domains where the model excels or struggles, and taken together, these findings advocate for prompting strategies that prioritize clarity, repeatability, and alignment with model capabilities, thereby fostering outputs that are both accurate and consistent.

Appendix

Appendix A: List of four prompts.

CoT1	"""You will be provided a math question. Solve the problem carefully and rigorously. Proceed step by step. For each step: explicitly state what you are doing. Make sure to state formulas and casework clearly. Here is an example:\Example problem: How many positive integers n satisfy \[\dfrac{n+1000}{\{70\}} = floor(\sqrt{n})? (Recall that floor(x) is the greatest integer not exceeding x.)\Example solution outline:\Step 1: We are given that \[\dfrac{n+1000}{\{70\}} = floor(\sqrt{n}). floor(\sqrt{n}) must be an integer, which means that $n+1000$ is divisible by 70. As $1000\equiv 20\pmod{70}$, this means that $n\equiv 50\pmod{70}$, so we can write $n=70k+50$ for some integer value k.\Step 2: Therefore, substituting $n=70k+50$ into the original equation, we get \[\dfrac{n+1000}{\{70\}}=\dfrac{70k+1050}{\{70\}}=k+15=floor(\sqrt{n}).\Step 3: Notice that the right hand side is a floor function, which implies that the value obtained after applying the floor function is at most 1 less than the value obtained before applying the floor function.\Step 4: Then, $k+15$ must be within 1 of $\sqrt{70k+50}$. This gives us the inequalities $\sqrt{70k+50}-1 < k+15$ and $k+15\leq\sqrt{70k+50}$.\Step 5: Squaring the second inequality, $k+15\leq\sqrt{70k+50}$, we get $k^2+30k+225 \leq 70k+50$. Moving everything to the left side of the inequality, we get $k^2-40k+175 \leq 0$. This is can be factored using the quadratic formula, and the inequality becomes $(k-5)(k-35) \leq 0$. This implies that $5\leq k$ and $k \leq 35$.\Step 6: For the first inequality, $\sqrt{70k+50}-1 < k+15$, we first move -1 to the right hand side, to get $\sqrt{70k+50}< k+16$. Squaring both sides, we get $70k+50 < k^2+32k+256$. Moving everything to the right side of the inequality, we get $ 0 < k^2-38k+206$.\Step 7: Using the quadratic formula to solve the inequality, we get $k<19-\sqrt{155}$ or $k>19+\sqrt{155}$.\Step 8: Notice that now we have bounds on the value of k. The first bound is $5 \leq k \leq 35$, and the second bound is $k<19-\sqrt{155}$ or $k>19+\sqrt{155}$.\Step 9: Since k must be an integer, we only need to find integers that satisfy the two inequalities. \Step 10: Testing integer values of k, we get $k=5,6,32,33,34,35$ as valid values of k.\Step 11: Recall that $n=70k+50$. Substituting each of the valid values of k into the equation, we test to make sure these values of k produce valid values of n as well. \Step 12: Since the original question asked for the number of solutions to the equation, and these six values of k produce valid values of n, our answer must be 6.\Step 13: Therefore, the final answer is 6.\----------Now solve the following problem. Remember to work carefully and rigorously. Let's think step-by-step: {problem}\
CoT2	**Role**: You are an expert mathematician specializing in AMC problem-solving. Solve all problems with extreme precision using this framework:### **Problem-Solving Framework***1. **DECOMPOSE*** (3-5 bullet points):- Identify core variables/quantities and their relationships- List all explicit constraints and implicit assumptions - Flag any units/conversion- Define success criteria: "The goal is to find...")2. **STRATEGY SELECTION*** (Justify choice):- Consider: Algebra, Combinatorics, Geometry, Number Theory, or Hybrid - Evaluate efficiency: "This method is optimal because..." - For alternator/liar problems: Explicitly model truth tables/For inequalities: Define boundary analysis approach/For combinatorics: Specify counting principle3. **RIGOROUS EXECUTION*** (Never skip steps):- Derive formulas from first principles:"Starting from [core principle], we derive..." Show all calculations:"Calculate: [step1] = [step2] = [result]"- Handle casework:"Case 1: [condition] → [logic] → [result]"- Track units/dimensions through4. **VALIDATION*** (Triple-check):- Dimensional analysis: "Units consistency: [unit1] = [unit2] √"- Edge case testing: "At boundary x=[value], we verify..."- Answer reversal: "If solution is X, plugging back yields [check]"- Option comparison (if applicable): "Choice A matches because..."- Reality check: "Does this quantity make sense? [Reason]"### **Example Template*** (Adapt to specific problem)**Problem***: "Carlos drove 70 min at 40 mph. Distance traveled?"**Decompose***:- Variables: time=70 min, speed=40 mph - Constraints: unit mismatch (min vs hours) - Goal: distance in miles **Strategy***: - Use distance = speed × time with unit conversion- Optimal: Convert minutes to hours first**Work***: - Convert: 70 min ÷ 70/60 hr = 7/6 hr - Calculate: distance = 40 mph × 7/6 hr = 280/6 = 46.67 miles **Validation***: - Units: mph × hr = miles √- Reverse: 46.67 miles / 40 mph = 1.167 hr = 70 min √- Realistic: 40mph in 70min = 46-47 miles √### **Critical Rules***- For liar/truth-teller problems: "Build response tables showing all answer combinations" - For geometry: "Include coordinate system setup and diagram constraints"- For inequalities:"Test critical points and sign changes"- For combinatorics: "Justify overcounting adjustments" **Problem to Solve***: {problem}
CoT3	"""Act as an expert AMC 12 mathematician. Solve the problem rigorously with these steps:1. **Decompose***: Identify variables, constraints, units, and goals.2. **Strategy***: Choose the fastest method (e.g., algebra, combinatorics, geometry). Justify why.3. **Work***: Show every step. Re-derive formulas; never skip calculations.4. **Validation***: - Check units/edge cases. - Plug the answer into the original problem. - Compare with multiple-choice options (A)–(E). - Re-calculate inversely if numerical. **Problem:** {problem}**Step 1: Decompose*** - (Break down problem specifics)**Step 2: Strategy*** - (Method + justification)**Step 3: Work*** (Detailed steps with labels)**Step 4: Validation*** - (Error checks + option comparison)**Final Answer is **:"""
CoT4	"""Act as an expert AMC 12 mathematician. Solve the problem step by step, and verify the logic in each step rigorously. Return the final answer is: \\boxed{}.Problem: {problem}"""

Appendix B: Model specification comparison, including state-of-the-art models Gemini 2.5 pro and Deepseek v3 0324.

	Gemma 27B	Phi-4	Llama 4 Maverick	Gemini 2.5 Pro	Deepseek V3 0324
Active Parameters (Billions)	27.4	14.7	17	Undisclosed	37
Openrouter Input pricing ($/M tokens)	0.09	0.06	0.15	1.25	0.25
Openrouter Output pricing ($/M tokens)	0.17	0.14	0.60	10.00	0.85
Release Date (MM/DD/YYYY)	03/12/2025	12/12/2024	04/05/2025	06/17/2025	03/24/2025

Appendix C: Llama-4-Maverick's (baseline) response to 2022 AMC12A problem 1.

```
"""## Step 1: Understand the given problem and the expression to be evaluated\nWe need to
evaluate the expression \\[3+\\frac{1}{3+\\frac{1}{3+\\frac13}}.\\]\n\n
## Step 2: Start simplifying from the innermost fraction\nFirst, simplify the innermost fraction:
$3 + \\frac{1}{3} = \\frac{9}{3} + \\frac{1}{3} = \\frac{10}{3}$.\n\n
## Step 3: Substitute the simplified innermost fraction back into the expression\nNow the
expression becomes \\[3+\\frac{1}{3+\\frac{1}{\\frac{10}{3}}} =
3+\\frac{1}{3+\\frac{3}{10}}.\\]\n\n
## Step 4: Simplify the next level of fraction\nSimplify the next innermost fraction: $3 +
\\frac{3}{10} = \\frac{30}{10} + \\frac{3}{10} = \\frac{33}{10}$.\n\n
## Step 5: Substitute this simplified fraction back into the expression\nNow the expression
becomes \\[3 + \\frac{1}{\\frac{33}{10}} = 3 + \\frac{10}{33}.\\]\n\n
## Step 6: Simplify the expression further\n$3 + \\frac{10}{33} = \\frac{99}{33} + \\frac{10}{33}
= \\frac{109}{33}$.\n\n
## Step 7: Convert the improper fraction to a mixed fraction as required\nTo convert
$\\frac{109}{33}$ to a mixed fraction, divide the numerator by the denominator: $109 \\div 33 =
3$ with a remainder of $10$. So, $\\frac{109}{33} = 3\\frac{10}{33}$.\n\nThe final answer is:
$\\boxed{3\\frac{10}{33}}$","""
```

Acknowledgments

I am deeply grateful to my advisor, Professor Siddharth Krishnan, for his invaluable mentorship, insightful feedback, and unwavering support throughout this research. His guidance has been instrumental in shaping the conceptual framework, methodology, and presentation of this work. I attest that the ideas, graphics, and writing in this paper are entirely my own.

References

1. OpenAI; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; et al. GPT-4 Technical Report. *arXiv preprint 2023*, arXiv:2303.08774.
2. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E. H.; Le, Q. V.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *NeurIPS 2022, 36*, 2201.11903. DOI: 10.48550/arXiv.2201.11903.
3. Ahn, J.; Verma, R.; Lou, R.; Liu, D.; Zhang, R.; Yin, W. Large Language Models for Mathematical Reasoning: Progresses and Challenges. *arXiv 2024*. DOI: 10.48550/arXiv.2402.00157.
4. Yan, J.; Wang, C.; Huang, J.; Zhang, W. Do Large Language Models Understand Logic or Just Mimic Context? *arXiv 2024*. DOI: 10.48550/arXiv.2402.12091.
5. Patil, A.; Jadon, A. Advancing Reasoning in Large Language Models: Promising Methods and Approaches. *arXiv [cs.CL] 2025*, *2502.03671v2*, *License: CC BY-SA 4.0*. DOI: 10.48550/arXiv.2502.03671.
6. Bi, Z.; Han, K.; Liu, C.; Tang, Y.; Wang, Y. Forest-of-Thought: Scaling Test-Time Compute for Enhancing LLM Reasoning. *arXiv [cs.CL] 2024*, 2412.09078. DOI: doi.org/10.48550/arXiv.2412.09078. License: CC BY-SA 4.0.
7. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing reasoning and acting in language models. *arXiv:2210.*03629 2022. DOI: doi.org/10.48550/arXiv.2210.03629.
8. Wang, B.; Min, S.; Deng, X.; Shen, J.; Wu, Y.; Zettlemoyer, L.; Sun, H. Towards Understanding Chain-of-Thought Prompting: An Empirical Study of What Matters. *ACL 2023*.
9. Renze, M.; Guven, E. The Benefits of a Concise Chain of Thought on Problem-Solving in Large Language Models. *arXiv 2024*, 2401.05618. DOI: 10.48550/arXiv.2401.05618
10. John Johnson. Small Language Models (SLM): A Comprehensive Overview. *HuggingFace*, February 22, 2025 https://huggingface.co/blog/jjokah/small-language-model.
11. Aralimatti, R.; Shakhadri, S. A. G.; KR, K.; Angadi, K. B. Fine-Tuning Small Language Models for Domain-Specific AI: An Edge AI Perspective. *arXiv 2025*, 2503.01933. DOI: 10.48550/arXiv.2503.01933
12. Belcak, P.; Heinrich, G.; Diao, S.; Fu, Y.; Dong, X.; Muralidharan, S.; Lin, Y. C.; Molchanov, P. Small Language Models are the Future of Agentic AI. *arXiv 2025*, 2506.02153. DOI: 10.48550/arXiv.2506.02153.

13. Zhang, W.; Xu, H.; Wang, Z.; He, Z.; Zhu, Z.; Ren, K. Can Small Language Models Reliably Resist Jailbreak Attacks? A Comprehensive Evaluation. *arXiv 2025*, 2503.06519. DOI: 10.48550/arXiv.2503.06519.
14. Shim, J.; Dong Ho, S.; Kim, J. CoT Harms Performance of Rather Smaller Language Models. *Proceedings of the International Conference on Industrial Engineering and Operations Management (1st World Congress)* 2024, 504-508. DOI: 10.46254/WC01.20240169.
15. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models Are Unsupervised Multitask Learners. *OpenAI blog 2019*
16. Mathematical Association of America. American Mathematics Competitions. MAA Website. https://maa.org/student-programs/amc/ (accessed 2025-07-24).
17. Zhang, J.; Zhao, Y.; Zhang, L.; Hu, J.; Luan, X.; Xu, Z.; Yang, F. Psychometric-Based Evaluation for Theorem Proving with Large Language Models. *arXiv 2025*, abs/2502.00855. DOI: 10.48550/arXiv.2502.00855.
18. Google DeepMind. Gemma 3 model card. https://huggingface.co/google/gemma-3-27b-it (accessed 2025-06-30).
19. Atil, B.; Aykent, S.; Chittams, A.; Fu, L.; Passonneau, R. J.; Radcliffe, E.; Rajagopal, G. R.; Sloan, A.; Tudrej, T.; Ture, F.; Wu, Z.; Xu, L.; Baldwin, B. Non-Determinism of "Deterministic" LLM Settings. *arXiv 2024*, 2408.04667. DOI: 10.48550/arXiv.2408.04667.
20. Ahn, J.; Yin, W. Prompt-Reverse Inconsistency: LLM Self-Inconsistency Beyond Generative Randomness and Prompt Paraphrasing. *arXiv 2025*, 2504.01282. DOI: 10.48550/arXiv.2504.01282.
21. Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large Language Models Are Zero-Shot Reasoners. *arXiv 2022*, 2205.11916. DOI: 10.48550/arXiv.2205.11916.
22. Ji, Z.; Yu, T.; Xu, Y.; Lee, N.; Ishii, E.; Fung, P. Towards Mitigating Hallucination in Large Language Models via Self-Reflection. Findings of the Association for Computational Linguistics: *EMNLP 2023* 2023, Findings, 1827–1843. DOI: 10.18653/v1/2023.findings-emnlp.123.
23. Shi, F.; Chen, X.; Misra, K.; Scales, N.; Dohan, D.; Chi, E. H.; Schärli, N.; Zhou, D. Large Language Models Can Be Easily Distracted by Irrelevant Context. *Proceedings of the 40th International Conference on Machine Learning* 2023, 202, 31210-31227
24. Levy, M.; Jacoby, A.; Goldberg, Y. Same Task, More Tokens: the Impact of Input Length on the Reasoning Performance of Large Language Models. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2024), Long Papers 2024, *1 (Long Papers)*, 15339–15353. DOI: 10.18653/v1/2024.acl-long.818.
25. Zhang, Z.; Li, J.; Lan, Y.; Wang, X.; Wang, H. An Empirical Study on Prompt Compression for Large Language Models. *Building Trust Workshop at ICLR 2025*, 2025. DOI: 10.48550/arXiv.2505.00019.
26. Schreiter, D. Prompt Engineering: How Prompt Vocabulary Affects Domain Knowledge. *arXiv 2025*, 2505.17037. DOI: 10.48550/arXiv.2505.17037.
27. Cao, B.; Cai, D.; Zhang, Z.; Zou, Y.; Lam, W. On the Worst Prompt Performance of Large Language Models. *arXiv 2024*, 2406.10248. DOI: 10.48550/arXiv.2406.10248.
28. Han, D.; Xia, M.; Madrigal Diaz, D.; Kessler, S.; Mallick, A.; Zhang, X.; Garcia, M. D. C. H.; Xu, J.; Rühle, V.; Rajmohan, S. Enhancing Reasoning Capabilities of Small Language Models with Blueprints and Prompt Template Search. *arXiv 2025*, 2506.08669. DOI: 10.48550/arXiv.2506.08669.
29. Kim, Y.; Yi, E.; Kim, M.; Yun, S. Y.; Kim, T. Guiding Reasoning in Small Language Models with LLM Assistance. *arXiv 2025*, 2504.09923. DOI: 10.48550/arXiv.2504.09923.
30. Qi, Z.; Ma, M.; Xu, J.; Zhang, L. L.; Yang, F.; Yang, M. Mutual Reasoning Makes Smaller LLMs Stronger Problem-Solvers. *arXiv 2024*, 2408.06195. DOI: 10.48550/arXiv.2408.06195.
31. Liu, X.; Xu, P.; Wu, J.; Yuan, J.; Yang, Y.; Zhou, Y.; Liu, F.; Guan, T.; Wang, H.; Yu, T.; McAuley, J.; Ai, W.; Huang, F. Large Language Models and Causal Inference in Collaboration: A Comprehensive Survey. *arXiv 2024*, 2403.09606. DOI: 10.48550/arXiv.2403.09606.

Author

Eric is currently a high school student in the United States. He is very passionate about both artificial intelligence and competitive mathematics, and has a strong background in competitive math, having qualified for the USAMO in 2025, and USAJMO in both 2023 and 2024.

DOI: 10.36838/IJHSR87.20