INTACT: Inference-Time Agentic Chain-of-Thought

Arnav Joshi
Cornell University
aj642@cornell.edu

Abstract

This report extensively explores developing an inference-time agentic chain-of-thought framework to improve reasoning and problem-solving capabilities in small to medium-sized language models (7-70 billion parameters). The hypothesis centered around the idea that a secondary critique-centric model could enhance the performance of a primary reasoning model by iteratively generating and evaluating reasoning steps, thereby emulating the structured reasoning paths typically observed in larger models, all without requiring reinforcement learning or fine-tuning. Through rigorous testing on datasets such as MATH and PRM800K, this study evaluates the impact on accuracy, consistency, and logical coherence. The findings show that the approach improves baseline performance on some non-instruction tuned models, but significant challenges persist for all other models, particularly in the smaller models' reliability and sensitivity to prompt design, underscoring areas for future improvement. Additionally, the findings demonstrate that using the critique model as a selector for a best-of-n generator rather than against a stepwise generator overcomes many of the difficulties involved with stepwise generation and produces improved results without any fine-tuning or optimization with the tradeoff of increased inference time and token usage. The code for this project can be found at https://github.com/joshiarnav/INTACT.

1 Introduction

Large language models have demonstrated remarkable advancements in multi-step reasoning tasks with chain-of-thought prompting. However, achieving similar outcomes in smaller or medium models presents distinct challenges due to their limited reasoning and contextual understanding capabilities. This project investigates an agentic CoT framework in which a critique model continuously evaluates and corrects reasoning steps generated by a primary model. By addressing errors and reinforcing logical consistency, the critique model aims to bolster the reasoning capabilities of smaller models.

1.1 Motivation

The project's motivation was primarily grounded in models such as "o1" by OpenAI, which spends time "thinking," i.e., trades off inference-time chain-of-thought reasoning for improved performance, especially on highly logical tasks. Although models such as these likely utilize some form of optimization to improve their stepwise reasoning and incentivize them to produce long chains of thought, there is potential for room for models to improve by utilizing this same process and methodology without explicitly training or optimizing a model as well. An opportunity is presented to perform a form of automated process supervision similar to the optimization process that would produce models such as "o1" but perform this process supervision live at inference time to improve model reasoning without modifying the base model. This would make it possible to improve model performance and reasoning for smaller models by offering a tradeoff between a model's time spent performing inference (i.e., token usage and time spent generating) and its performance. Most current strategies for improving model performance involve fine-tuning or optimizing a model in some fashion, which consists of having access to model weights or hardware capable of training a model, which grows more inaccessible daily. Inference-time reasoning is also less explored than training and alignment, leaving room for improvement.

Additionally, there appear to be works that focus on agentic approaches to improving model performance, such as the one presented in this report. Still, they are generally for a niche domain, for example, exclusively software engineering, and involve a more complex agentic setup. Furthermore, several works demonstrate that LLMs are generally stronger evaluators than reasoners. This hypothesis exploits this capability by focusing on a model's ability to evaluate outcomes independently of its generation to determine whether a step is correct or incorrect. The rationale is that this would improve a generator's chain of thought or outcome as the final outputs are evaluated as the model is reasoning through a problem. Moreover, the hypotheses in this report explore a relatively linear chain-of-thought structure, as previous papers explore alternative strategies such as trees-of-thought (Yao et al., 2023). Still, the findings demonstrate that these strategies perform best on niche problems and may not provide generalizable improvements. This could be investigated in future works.

1.2 Related Work

Automated CoT and Agentic Reasoning: Chain-of-thought prompting has consistently improved large language models for tasks involving multi-step reasoning, symbolic manipulation, and arithmetic calculations. The existing literature primarily focuses on CoT techniques in larger models, but the potential of these methods in smaller models remains underexplored. Here, I will review related work relevant to agentic CoT, automated CoT prompting, and the application of critique-based CoT methods.

1.2.1 Chain-of-Thought Prompting in LLMs

Research such as "Chain of Thought Prompting Elicits Reasoning in Large Language Models" (Wei et al., 2022), "Let's Verify Step by Step" (Cobbe et al., 2021), and "PAL: Program-Aided Language Models" (Gao et al., 2022) has highlighted CoT's utility in structured tasks like mathematics, logical reasoning, and programming. This concept has been expanded in papers exploring trees of thought, such as "Tree of Thoughts: Deliberate Problem Solving with Large Language Models" (Yao et al., 2023), which explore other structures for eliciting the same outcomes from LLMs. These studies demonstrate that breaking tasks into incremental steps enables models to achieve more accurate outputs by addressing one reasoning component at a time. However, they predominantly use human-defined CoT prompts/decisions in zero- or one-shot settings, leaving a gap in automated CoT exploration for smaller models.

1.2.2 Reinforcement Learning for LLMs

The initial proposal drew inspiration from OpenAI's use of RL with CoT prompts (i.e., the logic in "Let's Verify Step by Step" (Cobbe et al., 2021), which is presumably the rationale and basis behind the o1 model, which inspired this research). However, implementing RL methods (e.g., PPO) is challenging due to computational demands and difficulties in defining rewards for complex language tasks. Reinforcement learning has improved alignment and reasoning in many LLM applications, but there are limited studies currently involved with improving reasoning at inference time without using RL. Additionally, the mentioned experiments in the feedback (e.g., fine-tuning LLaMA-13B on PRM800K) have shown that fine-tuning smaller models as verifiers don't yield significant improvements, highlighting the need for alternative methodologies.

1.2.3 Agentic Frameworks

Very recent works, such as "Improving LLM Reasoning with Multi-Agent Tree-of-Thought Validator Agent" (Haji et al., 2024), present multi-agent approaches where one model generates reasoning steps, and a second model verifies or critiques these steps. This setup leverages verification as an alternative to reinforcement learning by treating step verification as a classification problem, leaving the reasoning up to a second agent rather than a human preference reinforced by a reward model. The two-model setup enables stepwise generation with a secondary model providing corrective feedback, potentially allowing smaller models to simulate CoT without extensive RL-based fine-tuning.

Training Small Models on Larger Model Outputs

Studies focused on training small models to replicate reasoning paths generated by larger models, such as "Distilling reasoning capabilities into smaller language models" (Shridhar et al., 2023), are also relevant to this project. These studies suggest that smaller models can benefit from training on stepwise outputs from larger models, offering a foundation for investigating whether critique-based CoT might be successfully distilled from larger model outputs. This may be explored later in the project, as the initial experiments will focus on the multi-agent setup and whether it holds merit.

2 Methodology

The project's primary goals included **RG1**: Implement and evaluate an agentic strategy that iteratively generates and critiques reasoning steps. **RG2**: Measure improvements in accuracy and tradeoffs in time/token efficiency over baselines. **RG3**: Modify agentic strategy and perform model ablation study to improve performance (time-permitting).

This approach aims to provide insights into how smaller models can approximate the reasoning accuracy of larger models without the computational overhead of RL or any fine-tuning.

3 Experimental Setup & Results

3.1 Experiment 1: Feasibility Study

Preliminary experiments were conducted to evaluate the feasibility of a critique-based, inference-time CoT system. Using Llama-3.2-11B-Vision as both the primary for reasoning and the secondary model as a critique agent, the agentic CoT approach was evaluated on the MATH dataset for math reasoning. The rationale behind this specific model was that this is the size range where models begin to demonstrate enhanced reasoning utilizing CoT. Additionally, this model is available at the time of this report for free inference under the Together API, which means it can be used, albeit with a heavy rate limit, which is critical as this agentic approach is very demanding in token usage.

3.1.1 Prompt Engineering for Primary and Critique Models

For the primary model, structured prompts to encourage stepwise responses were utilized, explicitly asking the model to break down the solution process into distinct steps. This involved heavy trial and error and was a largely empirical affair. The initial prompt for the primary model was:

```
  Problem: {problem statement}
  Generate only the first step in solving the problem. Be succinct and do not explain. Do not give the final answer.
```

This is followed by the following prompt for all further generations for the same problem:

```
  Problem: {problem statement}
  Steps so far:
  {primary model step generation 1}
  …
  {primary model step generation n}
  Generate only the next step in solving the problem. Be succinct and do not explain. If this or the previous step is the final step, you must indicate it with "Final Answer" in your response.
```

For the critique model, prompts were designed to assess whether each step met accuracy criteria, providing options such as "Correct," "Incorrect," "Backtrack," and "Complete." The critique model's output then determined the primary model's subsequent actions. Eventually, "Backtrack" and "Complete" were removed for the following reasons: "Backtrack" is implied due to an incorrect step and is pointless as the model will have deemed all steps before the current step as correct, so it could not decide to backtrack to further than the prior step. "Complete" was removed for more complex generation issues. The critique model could not effectively determine when the primary model was finished generating a solution even when the primary model was certain and would lead to misleading false negatives and copious extraneous inference costs. Additionally, due to an overwhelming number of false positives, additional context to think critically was added to the "Incorrect" choice, and it was moved before the "Correct" choice. The final and currently active critique model prompt, through empirical trial and error, is:

```
  You are evaluating steps in solving the following problem:
  Problem: {problem}
  Previous steps:
  {previous steps separated by newlines}
  Current step:
  {step}
  Please respond only with one of the following:
```

```
  - 'Incorrect' if the current step has any errors. Critically check any reasoning in the step.
  - 'Correct' if the current step is accurate.
```

3.1.2 Dataset and Task Selection

Experiments were run on MATH to assess baseline CoT effectiveness. This dataset is suitable as it requires each problem to be solved in a series of logically consistent steps, allowing for the validation of agentic CoT.

3.1.3 Overall Methodology

The multi-agent system was set up and run in Python by making inference API calls to Together AI API's "Llama-Vision-Free" model (which is an aliased Llama 3.2 Vision 11B model), as this model is currently available for free usage in 2024. The system first pulls a problem from the MATH dataset and feeds it into the primary model with the prompt as mentioned earlier. Then, this first step, alongside the problem statement, is presented to the critiquer model (this could be tested in future works by only presenting the current step to isolate the step and decrease token usage). The critiquer model then chooses either incorrect or correct (with no opportunity to explain its choice, which could also change in future works). If the primary model step is deemed accurate, the primary model continues generation with the problem and all its previous steps as context. If the step is considered incorrect, the step is removed from the primary model's context, and the model is re-prompted to generate this step (i.e., the primary model moves back one step). This continues until the agents either solve the problem or reach a maximum step generation limit (set to 10 empirically due to repetitive generations and model ability to solve a reasonable question in under ten steps — this is the limit of the number of times the primary model can be prompted to generate a step). These steps are saved to be evaluated for correctness against the MATH dataset. The current evaluation for correctness is to check if the "final answer" in the MATH dataset is present in the primary model's step generations and/or final step.

3.1.4 Limitations

Initial results demonstrated low accuracy overall (demonstrated in results later), and the framework was only tested on the agentic setup, meaning baselines (standard prompting and chain-of-thought prompting) needed to be established. Additionally, the critique model performance needed to be tested as it could not be determined whether it was performing at above-random accuracy, which would undermine the setup of the entire experiment, making it meaningless. Furthermore, it appeared that the model being utilized, Llama 3.2 Vision 11B, was incredibly slow due to its nature as a free trial API, meaning that single experiments would take weeks to run. Thus, a pivot on a model was also required for future experiments.

3.2 Experiment 2: Main Experiment

3.2.1 Dataset and Task Selection

This experiment aimed to build upon the previous experiment more rigorously, utilizing new models and providing a baseline to establish whether or not this framework offers tangible benefits. The datasets utilized were as follows:

MATH: MATH was chosen as math is a domain where chain-of-thought has demonstrated improved model reasoning abilities. Additionally, it provides a diverse set of relatively difficult (for smaller to medium LLMs) math problems, making it easier to demonstrate improved model performance through experiments. Models of this size tend to fail on most of the dataset at a baseline, but they can be improved via various techniques. Additionally, MATH has the benefit of having been built upon to generate a stepwise reasoning dataset, PRM800K, which greatly benefits the core of this report.

MATH-Uniform-Subsample: The MATH dataset proved too large for the experiments run in this report and consequently needed to be subsampled to run experiments in a timely manner. This led to a more uniform random subsampling from the MATH dataset to create a ~500 (504) problem dataset that has an identical quantity of problems for each topic within the dataset. This allows for a more uniform accuracy understanding of the reasoning capabilities across different types of problems. This dataset will be released alongside this report as part of the codebase.

3.2.2 Model Selection

The experimental setup primarily tested two models, both of which were utilized as both the generator and critiquer model in the experiment. It also completed establishing baselines for a third model:

- **Llama-3.1-8B-Instruct-Turbo**: This model was selected due to its size, speed, and cost. It was relatively cheap per token, which was critical for a project involving inference at this level of tokens (tens of millions). This model was utilized to set up most experiments and run empirical trials and errors to create a final set of methodologies.

- **Llama-3.3-70B-Instruct-Turbo**: This model was chosen due to its enhanced reasoning capabilities to determine how much-enhanced reasoning capabilities are important to experiments of this nature. It is documented that models at this scale tend to begin exhibiting emergent behaviors that improve their reasoning abilities. Therefore, it was critical to ensure that an experiment succeeded or failed due to the setup and not just a model's lack of reasoning capability. Additionally, this size remained small enough to run several experiments on.

- **Llama-3.1-11B-Vision**: To complete this model, it needed an established baseline and zero-shot chain-of-thought result from the previous experiment. Therefore, it was tested in the settings established below.

3.2.3 Experimental Setup

The experimental setup involved three different settings utilizing these models for generation and critiquing: baseline, zero-shot chain-of-thought (referred to as CoT henceforth), and agentic chain-of-thought (the focus of the research). The setups are as follows:

Baseline: Direct zero-shot problem-solving with a simple prompt structure. This setting involves a single API call with a temperature of 0.7 and no max token limit. This setting does not ask for any intermediate reasoning and frequently results in shorter, straight-to-answer responses. The prompt structure is as follows:

```
"{problem}  Ensure your answer is in the format $\\boxed{answer}$."
```

Zero-shot Chain-of-Thought (CoT): Zero-shot chain-of-thought step-by-step reasoning process prompt. This setting involves a single API call with a temperature of 0.7 and no max token limit. This setting generally results in longer responses with stepwise reasoning. The prompt structure is as follows:

```
"{problem} Ensure your answer is in the format $\\boxed{answer}$. Let's think step by step:"
```

Agentic Chain-of-Thought: Structured back-and-forth agentic framework. This setting involves several API calls per problem with a temperature of 0.7 for the generator model and 0.5 for the critique model, and a max token limit of 350 tokens for the generator model and 5 tokens for the critique model. This setup also involved parallelization and was rate-limited by the API at ten requests per second, maximized through parallelization. The components of the framework are as follows:

1. **Stepwise Generator Model**: Initially prompted once to generate the initial step only. Next, the model is prompted in a different manner with all of the previous steps and the problem. Each generation adds one further step in the reasoning process (if the last step was deemed correct by the critique model). The maximum number of possible generations was set to 15, and the maximum number of allowed steps was set to 10. These were set empirically as the model almost always made major errors when it reached this many generations or steps. The prompt structure for this design is as follows:

Initial prompt:

```
"Problem: {problem}
Generate only the first step in solving the problem. Be succinct. Do not give the final answer."
```

Subsequent prompts:

```
"Problem: {problem}
Steps so far:
{steps}
"Generate only the next step in solving the problem. Be succinct. If this is the final step, format your final answer as $\\boxed{answer}$."
```

2. **Stepwise Critique Model**: The critique model is fed the outputs from the generator. It is not given the opportunity to explain itself

Model	Experiment	Baseline	CoT	Agentic CoT	Best-of-N
Llama-3.2-11B-Vision	Exp. 1	14.9%	15.5%	18.4%	-
Llama-3.1-8B-Turbo	Exp. 2	51.2%	54.0%	47.1%	58.0%
Llama-3.3-70B-Turbo	Exp. 2	80.0%	80.8%	50.2%	80.0%

Table 1: Summary of Results (Accuracy). Accuracy metrics are reported for different models across experiments, comparing Baseline, CoT (Chain-of-Thought), Agentic CoT, and Best-of-N approaches.

or elaborate and is told to respond in binary as to whether a generated step is correct or not. This critique is utilized in the framework to determine if the model should continue or backtrack and erase the previously generated step. The prompt structure for the critique model is as follows:

```
"You are evaluating steps in solving the following problem:
Problem: {problem}
Previous steps: {steps}
Current step: {step}
Please respond only with one of the following:
- 'Incorrect' if the current step has any errors. Critically check any reasoning in the step.
- 'Correct' if the current step is accurate."
```

3.3 Experiment 3: Critique Verification

3.3.1 Dataset and Task Selection

The next experiment performed was focused on verifying the critique model's accuracy. This experiment was rooted in the critique model needing to be independently gauged for accuracy to determine whether it could achieve above-random critiques of stepwise reasoning generated by the generator model. The datasets selected for this task were as follows:

PRM800K: PRM800K is a stepwise reasoning dataset critical for a stepwise agent. The dataset is used to verify the accuracy of the critique model, i.e., to determine whether the critique model can verify steps at above random accuracy. If this could not be determined and the accuracy was lower than pure chain-of-thought or baseline, then it would be unclear whether the generator or the critique model is the struggling agent, leaving too much up to anecdotal interpretation. This dataset was chosen for this experimental setup as it provided a stepwise reasoning breakdown of the MATH dataset (used for the other experiments) with correct and incorrect steps labeled for all problems. This allowed for a very close emulation of the critique model's ability to check the generator's generated steps accurately.

3.3.2 Model Selection

- **Llama-3.1-8B-Instruct-Turbo**: This model was selected as it was in the previous experiment.
- **Llama-3.3-70B-Instruct-Turbo**: This model was chosen as it was in the previous experiment.

The final model (Llama 3.1 11B Vision) was dropped at this experiment stage due to its extensive wait times and slow API rate.

3.4 Experiment 4: Best-of-N Sampling

The final experiment tests a simple automated best-of-n or pass@k CoT sampling. The generator model is prompted with the standard CoT prompt from above with a higher temperature (0.9). For the sake of this experiment, this was limited to 3 generations per problem. These are all zero-shot chain-of-thought generations identical to the CoT setting described earlier. Then, the critiquer model is given the generations and told to select the best generation.

3.5 Evaluation

Evaluation was performed on the first, second, and fourth experiments identically. The solutions generated by the model were compared to the ground truth solutions provided by the MATH dataset. Empirically, the model almost never generated a correct solution that was symbolically different from the ground truth solution, so the evaluation was kept simple. The formatting for the final solution was searched for, and then the answer was extracted from this final solution formatting and compared exactly. For the experiments above, this appeared satisfactory and similar enough to previous harnesses and established evaluation techniques. While this could be improved upon (for example, by symbolic representations), this was not explored further due to time constraints. For the

Model	Accuracy	Precision	Recall	F1 Score
Llama-3.1-8B-Turbo	59.36%	69.13%	80.77%	74.50%
Llama-3.3-70B-Turbo	72.72%	77.01%	92.89%	84.21%

Table 2: Final Critique Model Results. Metrics include accuracy, precision, recall, and F1 score for the models.

third experiment, evaluation was straightforward as the model's critique decision either matched the ground truth rating or not, which would determine experimental accuracy.

3.6 Results

The results demonstrated in Table 1 & Table 2 show that the agentic framework works to some extent in the non-instruction-tuned model. Still, this model strongly underperforms instruction-tuned models, and this performance may be a byproduct of this lack of instruction-tuning. For the instruction-tuned models, the agentic framework appears to underperform the baseline. In the next experiment, it is demonstrated that the critique model has an above-random ability to select correct and incorrect results, meaning the issue likely lies within the generator, which aligns with previous works, which generally leads to fine-tuning the generator. Finally, the best-of-n experiment demonstrated the highest performance for the 8B model but not the 70B model, potentially implying that larger models do not seem to benefit or lose performance from such a setup.

4 Discussion

4.1 Key Achievements

Regardless of the main agentic model performance, these experiments still demonstrate achievements.

4.1.1 Viability of Critiques

This study demonstrated that a critique-based chain-of-thought framework may still have viability, as shown by the critique model. While the generator suffered from various issues, it is possible that a framework utilizing a critique model against stepwise generation is possible. Potentially, through minor fine-tuning or some other strategy, it may be possible to exploit the critique capabilities to perform inference-time chain-of-thought at a higher level.

4.1.2 Best-of-N Generation

The experiments highlighted that leveraging critique models to select the best generation from multiple attempts can bypass some of the challenges of stepwise generation, such as repeated failures on a specific reasoning step.

4.2 Limitations

4.2.1 Generator Model Reliability

Throughout the experiments, the generator model struggled to understand the task of stepwise generation, even with the larger 70B model. It frequently attempted to generate several steps or just go directly to the final answer, which was a costly error in both accuracy and token usage.

4.2.2 Prompt Sensitivity

The performance of both the generator and critique models' performance was highly sensitive to prompt design. Small changes in prompt phrasing resulted in significant performance variability, exacerbating reproducibility challenges. Additionally, formatting inconsistencies, such as unexpected deviations in output structure, frequently degraded the overall system performance. These errors tend to compound if the critique model makes an error early in the generation.

4.2.3 Inference Efficiency

The iterative nature of agentic CoT significantly increased token usage and inference time, creating scalability concerns. In extreme cases, the framework required up to 30 times more tokens per problem than baseline methods, making it less practical for large-scale deployment.

4.2.4 Large Model Bias

The critique model only appears to demonstrate strong gains at a larger model size, which somewhat defeats the purpose of using this framework to boost the performance of small—to medium-sized models. The smaller 8B model only achieved somewhat above-random performance on critiquing stepwise reasoning, meaning that its performance gain has a relatively low ceiling above its baseline CoT performance.

4.3 Observations

The critique model often exhibited inconsistent behavior, misclassifying correct reasoning steps as errors and overcorrecting minor issues, which disrupted reasoning coherence. It also struggled to identify subtle errors, while its inability to provide explanatory feedback hindered the system's ability to adjust during iterative evaluations. These inconsistencies, combined with the inefficiencies of the iterative framework, resulted in nearly double the inference time compared to baseline CoT methods. This inefficiency was particularly pronounced for tasks requiring extensive reasoning steps, posing significant challenges for real-time or large-scale applications.

Hyperparameter sensitivity further complicated the framework's performance. Parameters such as maximum step limits and the number of generations per step had a substantial impact on outcomes. For example, Llama-70B tended to generate irrelevant responses under higher step limits, while Llama-8B demonstrated stable yet limited performance, particularly for complex reasoning tasks. Prompt engineering presented additional challenges, as effective designs relied heavily on empirical trial and error. Small models were especially vulnerable to the effects of minor prompt changes, often amplifying their negative impact and leading to disproportionately degraded performance. The critique model's tendency to evaluate steps at random added to the complexity, highlighting the need for more reliable evaluation mechanisms. These challenges underscore the subjectivity and labor-intensive nature of prompt engineering for smaller models, as well as the difficulty of achieving consistent performance across different tasks and configurations.

4.4 Conclusion and Future Work

The findings of this study illustrate both the promise and limitations of agentic CoT frameworks in enhancing reasoning capabilities for smaller language models. While the iterative critique process offers a promising mechanism for structured reasoning, practical challenges—such as inference inefficiency, generator model variability, and prompt sensitivity—must be addressed for broader adoption. These limitations highlight the need for further research into more robust critique mechanisms, streamlined prompt designs, and alternative evaluation strategies to reduce dependency on manual fine-tuning. Despite these challenges, the potential of agentic CoT frameworks to improve reasoning accuracy in smaller models remains an exciting avenue for future exploration.

The experiments in this study highlight the potential of inference-time agentic CoT systems as a promising approach to enhancing reasoning capabilities in smaller language models. However, to fully realize these capabilities, critical bottlenecks—including inference inefficiency, critique model inconsistency, and prompt sensitivity—must be resolved.

Future work should focus on several key areas. Reinforcement learning methods, particularly process supervision, could be explored to optimize smaller models for stepwise reasoning. Enhancing critique model performance through existing datasets, such as GSM800K or PRM800K, could improve accuracy and consistency. Streamlined prompt engineering approaches are necessary to minimize token usage and mitigate formatting errors while maintaining performance. Exploring alternative mechanisms, such as multi-agent systems or lightweight reinforcement learning frameworks, could offer viable alternatives to an agentic critique-based CoT. Lastly, testing the framework across diverse datasets, such as PIQA and HellaSwag, would help assess its adaptability and robustness.

By addressing these areas, agentic CoT frameworks have the potential to improve the reasoning capabilities of smaller language models significantly, narrowing the performance gap with their larger counterparts and broadening the scope of applications for these systems.

References

Fatemeh Haji, Mazal Bethany, Maryam Tabar, Jason Chiang, Anthony Rios, and Peyman Najafirad. 2024. Improving LLM reasoning with multi-agent tree-of-thought validator agent. *arXiv preprint arXiv:2409.11527*. https://arxiv.org/abs/2409.11527.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*. https://arxiv.org/abs/2201.11903.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. *arXiv preprint arXiv:2212.00193*. https://arxiv.org/abs/2212.00193.

Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. 2024. AutoCodeRover: Autonomous program improvement. *arXiv preprint arXiv:2404.05427*. https://arxiv.org/abs/2404.05427.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*. https://arxiv.org/abs/2305.20050.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided language models. *arXiv preprint arXiv:2211.10435*. https://arxiv.org/abs/2211.10435.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv preprint arXiv:2306.05685. https://arxiv.org/abs/2306.05685.