

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Towards Enhanced Reasoning in Large Language Models

Permalink

<https://escholarship.org/uc/item/7t63t1z9>

Author

Ling, Zhan

Publication Date

2024

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Towards Enhanced Reasoning in Large Language Models

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Zhan Ling

Committee in charge:

Professor Hao Su, Chair
Professor Taylor Berg-kirkpatrick
Professor Jingbo Shang
Professor Zhuowen Tu

2024

Copyright

Zhan Ling, 2024

All rights reserved.

The Dissertation of Zhan Ling is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

TABLE OF CONTENTS

Dissertation Approval Page	iii
Table of Contents	iv
List of Figures	vi
List of Tables	vii
Acknowledgements	ix
Vita	xi
Abstract of the Dissertation	xiii
Chapter 1 Introduction	1
1.1 Reasoning with Large Language Models	1
1.2 Overview of Techniques and Contributions	2
1.2.1 Deductive Verification of Chain-of-Thought Reasoning	2
1.2.2 LLMs as Hierarchical Policy for Improved Exploration	3
1.2.3 Synthetic Long-Context Reasoning Data via Context Expansion	3
1.3 Additional Work Done During my Doctoral Career	3
Chapter 2 Deductive Verification of Chain-of-Thought Reasoning	5
2.1 Introduction	6
2.2 Related work	9
2.3 Motivation and Problem Formulation	10
2.4 Deductively Verifiable Chain-of-Thought Reasoning	12
2.4.1 Decomposition of Deductive Verification Process	12
2.4.2 Natural Program Deductive Reasoning Format	13
2.4.3 Integrating Deductive Verification with Unanimity-Plurality Voting	15
2.5 Experiments	15
2.5.1 Experimental Setup	16
2.5.2 Results	17
2.6 Additional Results	20
2.6.1 Zero-Shot vs. Few-Shot Full Reasoning Chain Verification Without Step-by-Step Decomposition or Natural Program Format	20
2.6.2 Zero-Shot vs. One-Shot Deductive Verification with Step-by-Step Decomposition and Natural Program Format	20
2.6.3 Deductive Verification with Vicuna Models	22
2.7 Additional Implementation Details	23
2.8 Limitations	24
2.9 Conclusion	25

Chapter 3	Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving	27
3.1	Introduction	28
3.2	Related Work	30
3.3	Language Model as a Hierarchical Policy for Exploration	32
3.4	Experiments	36
3.4.1	Do We Enhance the Discovery and Visibility of Correct Solutions?	37
3.4.2	Do We Improve Final Answer Accuracy for Challenging Reasoning Problems?	40
3.4.3	Ablation Study	43
3.5	Additional Results	45
3.5.1	Failure Example Analysis	45
3.5.2	Evaluation on GSM8K	45
3.6	Additional Implementation Details	47
3.7	Further Analysis and Limitations	48
3.8	Conclusion	48
Chapter 4	Long-Context Reasoning Synthesis through Context Expansion	50
4.1	Introduction	51
4.2	Related Work	54
4.3	Our Benchmark: LongReason	56
4.3.1	Long-context Reasoning Question Construction via Context Expansion ..	56
4.3.2	The Statistics of LongReason	59
4.4	Exerperiments & Results	60
4.4.1	Experimental setup	61
4.4.2	Main Results	61
4.4.3	Further Analysis	62
4.5	Model Information	64
4.6	Hyperparameters for LongReason Construction	65
4.7	Conclusion and Limitations	65
Chapter 5	Finale	67
Bibliography	69

LIST OF FIGURES

Figure 2.1.	Illustration of Natural Program.	7
Figure 2.2.	Natural Program-based deductive reasoning verification.	8
Figure 3.1.	Overview of our approach, which frames language models as a hierarchical policy for exploration.	30
Figure 3.2.	A detailed illustration of our approach that frames language models as a hierarchical policy for exploration.	32
Figure 3.3.	Illustration of the partitioning of the reasoning chain space based on the high-level tactics employed in the solution.	35
Figure 3.4.	Statistics for the “CoT Sampling + Majority Voting” baseline.	38
Figure 4.1.	Overview of our pipeline for constructing LongReason.	53
Figure 4.2.	An illustrative example in LongReason.	55
Figure 4.3.	The number of reasoning steps in the ground-truth analysis for questions in LongReason.	59
Figure 4.4.	Performance of the Qwen2.5 series on LongReason, with model sizes ranging from 7B to 72B.	63
Figure 4.5.	Comparison of the long-context reasoning performance between Gemini-1.5 Pro and Claude 3.5-Sonnet across different task categories.	63
Figure 4.6.	An example where Gemini-1.5 Pro provide a incorrect reasoning given the long-context question while the s. The key difference in reasoning is underlined.	64

LIST OF TABLES

Table 2.1.	An example question from GSM8K with a generated CoT reasoning path with GPT-3.5 (turbo).	11
Table 2.2.	Zero-shot reasoning chain verification accuracy for GPT-3.5-turbo (ChatGPT).	12
Table 2.3.	Final answer accuracy comparison on GPT-3.5-turbo (ChatGPT).	17
Table 2.4.	Deductive verification accuracy of reasoning chains for GPT-3.5-turbo (ChatGPT).	18
Table 2.5.	Impact of different values of k' on the verification accuracy of reasoning chains using our Unanimity-Plurality Voting strategy. Experiments are performed on AddSub using GPT-3.5-turbo (ChatGPT).	19
Table 2.6.	ChatGPT generates the correct final answer but provides incorrect premise numbers for grounding the first reasoning step.	19
Table 2.7.	An example question with ambiguous wordings.	20
Table 2.8.	Comparison of reasoning chain verification accuracy for GPT-3.5-turbo with zero / two-shot prompting.	21
Table 2.9.	Comparison between zero-shot and one-shot accuracy of deductive verification for GPT-3.5-turbo (ChatGPT) with step-by-step decomposition and Natural Program format.	21
Table 2.10.	One-shot Deductive Verification Accuracy of Vicuna-7B and Vicuna-13B.	23
Table 2.11.	Hyperparameters for finetuning Vicuna models with our deductive verification dataset.	23
Table 2.12.	Regular Expression for extracting the final answers of different kinds of questions.	25
Table 3.1.	An example problem from the MATH dataset.	31
Table 3.2.	Comparison of the Grouped-Majority Recall (“GMR”) and the final answer accuracy (“Acc.”) between the CoT sampling + Voting baseline and our two exploration approaches.	39

Table 3.3.	Comparison of the Grouped-Majority Recall (“GMR”) and the final answer accuracy (“Acc.”) on our 1047-question MATH Level-5 evaluation set using GPT-3.5 as the language model for the CoT Sampling + Voting baseline along with the low-level follower policy in our approaches.	42
Table 3.4.	(a) Comparison of the Grouped-Majority Recall and the final answer accuracy on three of the challenging STEM subjects from the MMLU dataset. (b) Final answer accuracy of WizardMath-7B-V1.1 on our 140-question MATH level-5 evaluation set.	42
Table 3.5.	Ablation on using (a) different models and different k (numbers of comparison repetitions); (b) different temperatures (T) during our tournament-based reasoning chain selection.	43
Table 3.6.	Effect of majority voting and our tournament-based reasoning chain selection on the final-answer accuracy of the CoT + Sampling baseline and our hierarchical policy approaches.	44
Table 3.7.	A failure case in which the GPT-4 high-level leader generates a highly-relevant hint for the question.	45
Table 3.8.	A failure case in which a very similar question along with its ground truth solution.	46
Table 3.9.	A failure case in which the GPT-4 high-level leader generates an irrelevant hint, resulting in wrong answer.	46
Table 3.10.	Final answer accuracy on GSM8K. We adopt GPT-3.5 as the low-level follower, and we sample 32 reasoning chains per problem.	47
Table 3.11.	Cost comparison between our approach and the CoT + Sampling Baseline on our 140-question MATH Level-5 evaluation set.	47
Table 3.12.	Comparison on the number of input and output tokens per-question between our approach and the CoT + Sampling Baseline on our 140-question MATH Level-5 evaluation set.	47
Table 4.1.	Comparison of LongReason with other long-context benchmarks.	51
Table 4.2.	Performance (%) of selected LLMs on LongReason.	60
Table 4.3.	Ablation study on the position of the final inquiry for selected models evaluated at context lengths ranging from 8K to 128K.	62
Table 4.4.	Information of evaluated and analyzed models in LongReason.	65

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to Prof. Hao Su for his invaluable guidance and unwavering support throughout my Ph.D. journey.

My heartfelt thanks to Prof. Su, who has been a guiding light from my undergraduate years through my Ph.D. journey, especially during moments when I felt lost. I am especially grateful for the opportunity you provided to pursue research as a Ph.D. student at UC San Diego, which has been instrumental in shaping the foundation of my aspirations. Your encouragement and mentorship helped me find my path and pursue my dreams. Your guidance and support have helped me build essential skills, strengthen my abilities, and prepare for future challenges. I deeply appreciate the time and effort you invested in my growth as a researcher.

I am deeply grateful to my Ph.D. committee members, Prof. Taylor Berg-Kirkpatrick, Prof. Jingbo Shang, and Prof. Zhuowen Tu, for their insightful contributions and steadfast support throughout this journey. Their guidance has been invaluable and instrumental in shaping my work.

I would also like to extend my heartfelt thanks to my collaborators, including my labmates, colleagues, and friends, whose inspiration and support have significantly enriched my research experience. In particular, I am profoundly grateful to Zhiao Huang, Tongzhou Mu, Xuanlin Li, and Yunhao Fang for their encouragement and collaboration. Furthermore, I am immensely thankful to Roland Memisevic, Mingu Lee, Reza Pourreza, Jiecao Chen, and Kang Liu for their insightful mentorship during my internships at Qualcomm and ByteDance, which played a pivotal role in deepening my understanding of reasoning with large language models.

Lastly, I owe a special debt of gratitude to my parents for their unwavering and unconditional support. Their love and belief in me have been my guiding light through every challenging moment, especially during the COVID-19 pandemic. Thank you for standing by me and for your endless encouragement.

Chapter 2, in full, is a reprint of the material published in the 2023 Neural Information Processing Systems (NeurIPS): “Deductive Verification of Chain-of-Thought Reasoning” (Zhan

Ling*; Yunhao Fang*; Xuanlin Li; Zhiao Huang; Mingu Lee; Roland Memisevic; Hao Su). The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of material from the publicly available preprint: “Unleashing the Creative Mind: Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving” (Zhan Ling; Yunhao Fang; Xuanlin Li; Tongzhou Mu; Mingu Lee; Reza Pourreza; Roland Memisevic; Hao Su). The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of a work currently under preparation for submission: “LongReason: A Synthetic Long-Context Reasoning Benchmark via Context Expansion” (Zhan Ling, Kang Liu, Kai Yan, Yifan Yang, Weijian Lin, Ting-Han Fan, Lingfeng Shen, Zhengyin Du, Jiecao Chen). The dissertation author was the primary investigator and author of this paper.

VITA

- 2015–2019 Bachelor of Engineering, Tsinghua University
2019–2023 Master of Science, University of California San Diego
2019–2024 Doctor of Philosophy, University of California San Diego

PUBLICATIONS

1. Fangchen Liu, **Zhan Ling**, Tongzhou Mu, Hao Su. State Alignment-based Imitation Learning. ICLR 2020.
2. Tongzhou Mu*, **Zhan Ling***, Fanbo Xiang*, Derek Yang*, Xuanlin Li*, Stone Tao, Zhiao Huang, Zhiwei Jia, Hao Su. ManiSkill: Generalizable Manipulation Skill Benchmark with Large-Scale Demonstrations. NeurIPS Track 2021.
3. Xiaoshuai Zhang*, Rui Chen*, Ang Li**, Fanbo Xiang**, Yuzhe Qin**, Jiayuan Gu**, **Zhan Ling****, Minghua Liu**, Peiyu Zeng**, Songfang Han***, Zhiao Huang***, Tongzhou Mu***, Jing Xu, Hao Su. Close the Visual Domain Gap by Physics-Grounded Active Stereovision Depth Sensor Simulation. T-RO & IROS 2023.
4. Xinyue Wei*, Minghua Liu*, **Zhan Ling**, Hao Su. Approximate Convex Decomposition for 3D Meshes with Collision-Aware Concavity and Tree Search. SIGGRAPH 2022.
5. Zhiwei Jia, Xuanlin Li, **Zhan Ling**, Shuang Liu, Yiran Wu, Hao Su. Improving policy optimization with generalist-specialist learning. ICML 2022.
6. Minghua Liu*, Xuanlin Li*, **Zhan Ling***, Yangyan Li, Hao Su. Frame Mining: a Free Lunch for Learning Robotic Manipulation from 3D Point Clouds. CoRL 2022.
7. Jiayuan Gu*, Fanbo Xiang*, Xuanlin Li**, **Zhan Ling****, Xiqiang Liu**, Tongzhou Mu**, Yihe Tang**, Stone Tao**, Xinyue Wei**, Yunchao Yao**, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, Hao Su. ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills. ICLR 2023.
8. Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, **Zhan Ling**, Fatih Porikli, Hao Su. PartSLIP: Low-Shot Part Segmentation for 3D Point Clouds via Pretrained Image-Language Models. CVPR 2023.
9. Zhiao Huang, Litian Liang, **Zhan Ling**, Xuanlin Li, Chuang Gan, Hao Su. Reparameterized Policy Learning for Multimodal Trajectory Optimization. ICML 2023 Oral.
10. **Zhan Ling***, Yunchao Yao*, Xuanling Li, Hao Su. On the Efficacy of 3D Point Cloud Reinforcement Learning. arXiv:2306.06799

11. Xuanlin Li*, Yunhao Fang*, Minghua Liu, **Zhan Ling**, Zhuowen Tu, Hao Su, Distilling Large Vision-Language Model with Out-of-Distribution Generalizability. ICCV 2023.
12. **Zhan Ling***, Yunhao Fang*, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, Hao Su. Deductive Verification of Chain-of-Thought Reasoning. NeurIPS 2023.
13. **Zhan Ling**, Yunhao Fang, Xuanlin Li, Tongzhou Mu, Mingu Lee, Reza Pourreza, Roland Memisevic, Hao Su. Unleashing the Creative Mind: Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving. Arxiv: 2311.00694
14. Ying Su, **Zhan Ling**, Haochen Shi, Cheng Jiayang, Yauwai Yim, Yangqiu Song. ActPlan-1K: Benchmarking the Procedural Planning Ability of Visual Language Models in Household Activities. EMNLP 2024 Main.
15. **Zhan Ling**, Kang Liu, Kai Yan, Yifan Yang, Weijian Lin, Ting-Han Fan, Lingfeng Shen, Zhengyin Du, Jiecao Chen. LongReason: A Synthetic Long-Context Reasoning Benchmark via Context Expansion. In submission.

FIELDS OF STUDY

Major Field: Computer Science (Natural Language Processing, Robotics, Reinforcement Learning)

ABSTRACT OF THE DISSERTATION

Towards Enhanced Reasoning in Large Language Models

by

Zhan Ling

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Hao Su, Chair

Large Language Models (LLMs) have demonstrated remarkable progress across diverse natural language tasks. Recently, Chain-of-Thought methods have been introduced to enhance reasoning by generating detailed and comprehensive reasoning processes. However, challenges such as hallucinations, error accumulation, and limited exploration hinder their effectiveness on complex tasks. Additionally, the near-exhaustion of high-quality natural language data on the internet poses a significant barrier to further improving the reasoning capabilities of LLMs.

To address these challenges, this dissertation investigates two key directions: enhancing inference techniques and synthesizing reasoning data to strengthen LLM reasoning capabilities. First, it introduces a deductive verification method that enables self-verification of reasoning

chains generated by LLMs, ensuring more rigorous and accurate reasoning during inference. Second, it addresses the limitations in exploring diverse reasoning strategies by framing reasoning as a hierarchical policy, where high-level tactics guide detailed low-level problem-solving through in-context learning with LLMs. In addition, it explores data synthesis for long-context reasoning tasks, which is particularly challenging even for human and very rare natural data on the internet. It proposes a novel data synthesis method that can generate long-context reasoning data with diverse and realistic reasoning patterns. The evaluation of the generated long-context reasoning dataset using this method reveals that even state-of-the-art LLMs struggle to perform robustly, highlighting the potential of the synthetic data strategy for enhancing LLM training.

This dissertation contributes to advancing LLM reasoning abilities through novel methods that address critical limitations in both training and inference. These advancements provide valuable insights and pave the way for stronger and more reliable reasoning in LLMs.

Chapter 1

Introduction

1.1 Reasoning with Large Language Models

Large language models (LLMs) [91, 101, 5, 120], trained on vast amounts of high-quality internet data and incorporating knowledge accumulated by humanity, have demonstrated remarkable capabilities in understanding human language and solving natural language tasks. Recently, the chain-of-thought (CoT) prompting [125, 65] has been introduced, enabling LLMs to reason through multiple steps. This significantly enhances their performance on tasks requiring reasoning abilities, such as math word problems and logical inference.

Despite these advancements, LLMs continue to face significant challenges when handling complex reasoning tasks during inference. While CoT prompting allows models to generate detailed reasoning processes, it can inadvertently lead to issues such as hallucinations and compounded errors in intermediate reasoning steps—particularly when addressing problems requiring numerous reasoning stages. Another major limitation is their constrained exploration ability for solving challenging reasoning tasks. Even with techniques like sampling [123] or search-based methods [133], LLMs often fail to explore diverse strategies, which may result in the inability to sample any correct reasoning paths. This limitation persists even when human intervention is available to select the best path as the final output, rendering the models ineffective in such scenarios.

Moreover, while current LLMs are trained on nearly all available high-quality internet

data, their performance remains constrained on some challenging reasoning tasks. The scarcity of high-quality data tailored to complex reasoning may impede the models’ ability to learn intricate reasoning patterns, making it difficult to further enhance their reasoning capabilities.

1.2 Overview of Techniques and Contributions

This dissertation aims to tackle the challenge of enhancing reasoning capabilities in LLMs from both inference and training perspectives. First, we explore methods to improve the accuracy and reliability of reasoning chains generated by LLMs through self-verification. In Chapter 2, we present an approach for step-by-step verification of reasoning chains to ensure their validity. Second, we focus on enhancing the exploration capabilities of LLMs by framing reasoning as a hierarchical policy. In Chapter 3, we introduce techniques to encourage LLMs to sample more diverse reasoning strategies, particularly within the tactical space. Lastly, we propose a novel data synthesis method to address the scarcity of rare reasoning data in current datasets. Specifically, in Chapter 4, we detail how to synthesize long-context reasoning question-answering data by leveraging existing short-context data. These contributions are comprehensively discussed in the following chapters.

1.2.1 Deductive Verification of Chain-of-Thought Reasoning

In “Deducting Verification of Chain-of-Thought Reasoning” [67], we first propose a novel framework, Natural Program, that enables LLMs to perform explicit and rigorous deductive reasoning while ensuring the trustworthiness of their processes through self-verification. Natural Program decomposes reasoning verification into step-by-step subprocesses, enhancing the precision and grounding of reasoning steps. This approach enables models to perform self-verification at each stage, significantly improving correctness and trust in solving complex reasoning tasks.

1.2.2 LLMs as Hierarchical Policy for Improved Exploration

In "Unleashing the Creative Mind: Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving" [67], we address LLMs' limitations in exploring diverse reasoning strategies by framing them as a hierarchical policy via in-context learning. This framework consists of a visionary "leader" proposing diverse high-level problem-solving strategies, followed by a "follower" executing detailed reasoning processes guided by the leader's instructions. By sampling multiple reasoning chains and using a tournament-based approach to evaluate and select the best solutions, we enhance exploration and improve the accuracy of solutions to challenging problems.

1.2.3 Synthetic Long-Context Reasoning Data via Context Expansion

In "LongReason: A Synthetic Long-Context Reasoning Benchmark via Context Expansion", we propose a new dataset, LongReason, generated through a novel data synthesis method that expands shorter reasoning problems into long-context scenarios. We evaluate various LLMs on LongReason and demonstrate that it poses significant challenges, even for state-of-the-art models. This highlights the potential of our synthetic dataset to serve as a valuable resource for improving yeLLMs' reasoning capabilities.

1.3 Additional Work Done During my Doctoral Career

My dissertation primarily focuses on enhancing reasoning abilities in large language models. Beyond this, I have been fortunate to explore diverse topics within artificial intelligence, contributing to areas such as robotics, reinforcement learning, imitation learning, computer vision, and vision foundation models.

I have devoted significant effort to generalizable robot manipulation. I co-led the ManiSkill [89] project and served as a primary contributor in ManiSkill2 [40], which are unified simulation platforms designed for studying and improving generalizable robot manipulation

skills. Additionally, I contributed to developing algorithms that build more realistic simulation environments for robot manipulation [126, 138]. My work also explores integrating 3D representation with deep reinforcement learning, studying the effects of 3D representation, 3D augmentation [69], and frame selection [73] to improve sample efficiency in reinforcement learning.

Beyond robotics, I have collaborated on developing new reinforcement learning and imitation learning algorithms. In [50], we proposed a generalist-specialist learning framework for deep reinforcement learning. In [70], we introduced a state-alignment-based imitation learning algorithm capable of cross-modality imitation. In [48], we explored a novel model-based reinforcement learning algorithm leveraging multimodal representations for enhanced exploration capabilities. In the field of computer vision, I have contributed to distilling vision foundation models [63], developing part detection methods [74], and evaluating vision-language models [116]. These efforts have broadened my expertise and deepened my understanding across various areas of AI.

Chapter 2

Deductive Verification of Chain-of-Thought Reasoning

Large Language Models (LLMs) significantly benefit from Chain-of-thought (CoT) prompting in performing various reasoning tasks. While CoT allows models to produce more comprehensive reasoning processes, its emphasis on intermediate reasoning steps can inadvertently introduce hallucinations and accumulated errors, thereby limiting models' ability to solve complex reasoning tasks. Inspired by how humans engage in careful and meticulous deductive logical reasoning processes to solve tasks, we seek to enable language models to perform *explicit and rigorous deductive reasoning*, and also ensure the *trustworthiness* of their reasoning process through self-verification. However, directly verifying the validity of an entire deductive reasoning process is challenging, even with advanced models like ChatGPT. In light of this, we propose to decompose a reasoning verification process into a series of step-by-step subprocesses, each only receiving their necessary context and premises. To facilitate this procedure, we propose **Natural Program**, a *natural language-based* deductive reasoning format. Our approach enables models to generate precise reasoning steps where subsequent steps are more rigorously grounded on prior steps. It also empowers language models to carry out reasoning self-verification in a *step-by-step* manner. By integrating this verification process into each deductive reasoning stage, we significantly enhance the rigor and trustfulness of generated reasoning steps. Along this process, we also improve the answer correctness on complex reasoning tasks.

2.1 Introduction

The transformative power of large language models, enhanced by Chain-of-Thought (CoT) prompting [125, 54, 144, 111], has significantly reshaped the landscape of information processing [35, 76, 124, 136, 34, 135, 58, 80], fostering enhanced abilities across a myriad of disciplines and sectors. While CoT allows models to produce more comprehensive reasoning processes, its emphasis on intermediate reasoning steps can inadvertently introduce hallucinations [14, 82, 41, 49] and accumulated errors [14, 127, 6], thereby limiting models' ability to produce cogent reasoning processes.

In fact, the pursuit of reliable reasoning is not a contemporary novelty; indeed, it is an intellectual endeavor that traces its roots back to the time of Aristotle's ancient Greece. Motivated by the desire to establish a rigorous reasoning process, in his "Organon," Aristotle introduced principles of logic, in particular, syllogism, a form of logical argument that applies deductive reasoning to arrive at a conclusion based on two or more propositions assumed to be true. In disciplines that rigorous reasoning is critical, such as judicial reasoning and mathematical problem solving, documents must be written in a formal language with a logical structure to ensure the validity of the reasoning process.

We yearn for this sequence of reliable knowledge when answering questions. Our goal is to develop language models that can propose potential solutions through reasoning in logical structures. Simultaneously, we aim to establish a verifier capable of accurately assessing the validity of these reasoning processes. Despite recent significant explorations in the field, such as [123]'s emphasis on self-consistency and [78, 19]'s innovative use of codes to represent the reasoning process, these approaches still exhibit considerable limitations. For example, consistency and reliability are not inherently correlated; as for program codes, they are not powerful enough to represent many kinds of reasoning process, e.g., in the presence of quantifiers ("for all", "if there exists") or nuances of natural language (moral reasoning, "likely", ...).

We propose leveraging the power of natural language to achieve the deductive reasoning

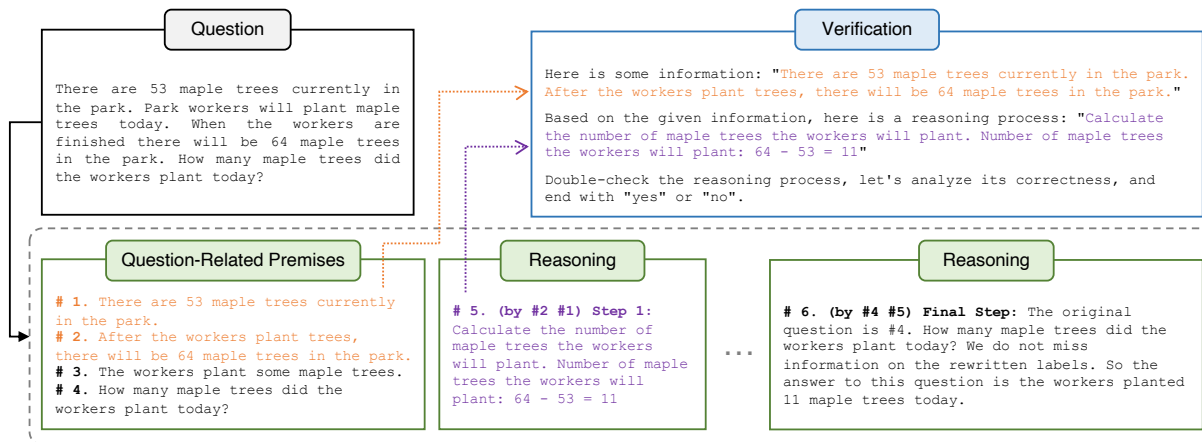


Figure 2.1. Illustration of Natural Program (bottom 3 boxes), a natural language-based deductive reasoning format for LLM reasoning chain generation. Natural Program allows individual reasoning steps (an example in purple) along with their minimal set of premises (an example in yellow) to be easily extracted, which facilitates the verification of deductive reasoning processes.

emphasized in ancient Greek logic, introducing a “*natural program*”. This involves retaining natural language for its inherent power and avoiding the need for extensive retraining with large data sets. A natural program represents a rigorous reasoning sequence, akin to a computer program. We expect implementations of the idea to have two properties: 1) that natural programs are generated with minimal effort from an existing language model capable of CoT reasoning, preferably through in-context learning; 2) that the natural program can be easily verified for reliability in the reasoning process.

Through a step-by-step investigation, we discovered that large language models have the potential to meet our expectation. Naïve CoT prompts like “Let us think step by step.” has many flaws, and entrusting the entire verification process to a large model like ChatGPT can still lead to significant error rates. However, we found that, if the reasoning process is very short, and only based on necessary premises and contexts, the verification of existing large language models is already quite reliable. Therefore, our approach is to design prompts that induce CoT processes comprised of rigorous premises/conditions and conclusions with statement labels, and verification can be done by gradually isolating very few statements within the long thought chain. Experimentally, we found that nearly all reasoning that passed the verification was rigorous, and

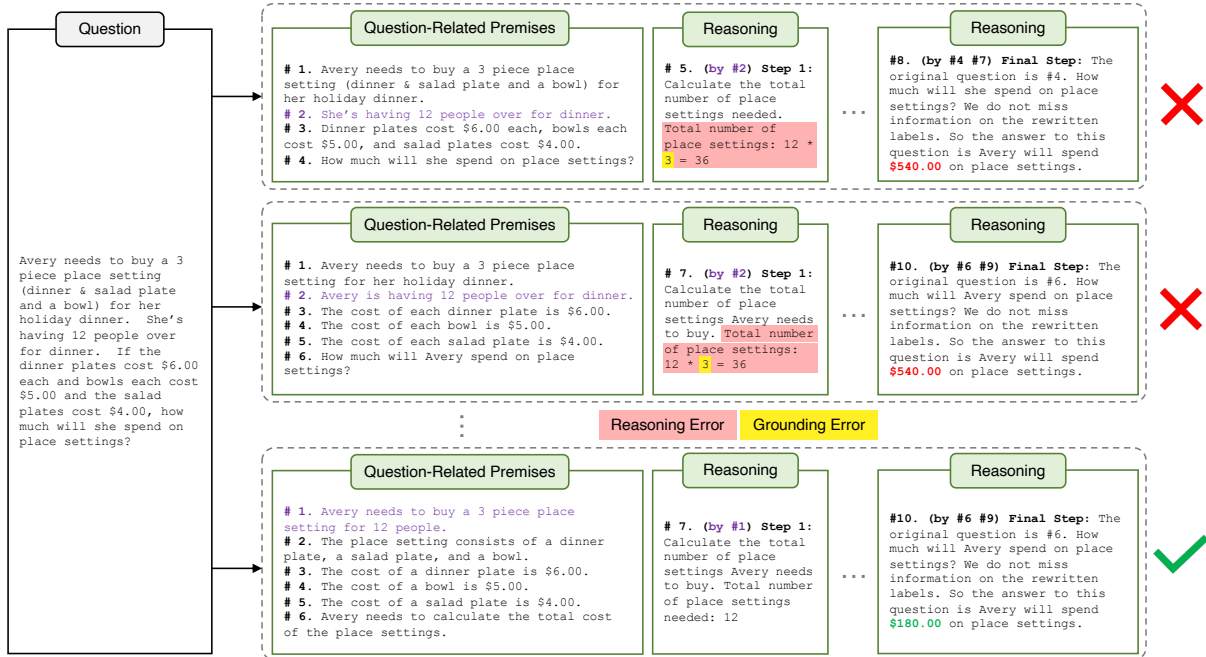


Figure 2.2. Through our Natural Program-based deductive reasoning verification approach, we alleviate LLM’s reasoning and grounding errors in problem solving steps, thereby enhancing the rigorousness, trustworthiness, and interpretability of generated reasonings.

nearly all that did not pass had elements of imprecision in the reasoning process, even if they occasionally arrived at correct answers.

It is worth emphasizing that, we are not looking for a method to just maximize the correctness rate of final answers; instead, we aspire to generate a cogent reasoning process, which is more aligned with the spirit of judicial reasoning. When combined with sampling-based methods, our method can identify low-probability but rigorous reasoning processes. When repeated sampling fails to yield a rigorous reasoning process, we can output “unknown” to prevent hallucinations that mislead users.

We demonstrate the efficacy of our natural program-based verification approach across a range of arithmetic and common sense datasets on publicly-available models like OpenAI’s GPT-3.5-turbo (175B). Our key contributions are as follows:

1. We propose a novel framework for rigorous deductive reasoning by introducing a “Natural Program” format (Fig. 2.1), which is suitable for verification and can be generated by

just in-context learning;

2. We show that reliable self-verification of long deductive reasoning processes written in our Natural Program format can be achieved through step-by-step subprocesses that only cover necessary context and premises;

3. Experimentally, we demonstrate the superiority of our framework in improving the rigor, trustworthiness, and interpretability of LLM-generated reasoning steps and answers (Fig. 2.2).

2.2 Related work

Reasoning with large language models. Recent large language models (LLMs) [13, 23, 137, 121, 106, 45, 24, 105] have shown incredible ability in solving complex reasoning tasks. Instead of letting LLMs directly generate final answers as output, prior work have shown that by encouraging step-by-step reasoning through proper prompting, such as Chain-of-Thought (CoT) prompting [125] and many others [54, 144, 141, 113, 123, 145, 75, 134], LLMs exhibit significantly better performance across diverse reasoning tasks. To further improve the step-by-step reasoning process, some recent studies have investigated leveraging external solvers such as program interpreters [107, 19, 78], training and calling external reasoning modules [26], or performing explicit search to generate deductive steps [11, 118]. Parallel to these works, we do not rely on external modules and algorithms, and we directly leverage the in-context learning ability of LLMs to generate more precise and rigorous deductive reasoning steps.

Large language models as verifiers. Using language models to evaluate model generations has been a long standing idea [56, 104, 109, 14]. As LLMs exhibit impressive capabilities across diverse tasks, it becomes a natural idea to use LLMs as evaluation and verification tools. For example, [25, 26, 95] finetune LLMs to verify solutions and intermediate steps. LLMs aligned with RLHF [94, 91, 123] have also been employed to compare different model generations. In addition, recent works like [112, 128, 79, 20] leverage prompt designs to allow LLMs

to self-verify, self-refine, and self-debug without the need for finetuning. However, these works do not focus on the rigorousness and trustworthiness of the deductive reasoning processes at every reasoning step. In this work, we propose a natural language-based deductive reasoning format that allows LLMs to self-verify *every* intermediate step of a deductive reasoning process, thereby improving the rigorousness and trustfulness of reasoning

2.3 Motivation and Problem Formulation

A reasoning-based question-answering (QA) task can be defined as a tuple (Q, C, O, A) [103], where Q is the target question; C is the context of a question, such as the necessary background for answering a question; $O = (o_1, o_2, \dots, o_k)$ are optional answer choices if Q is a K-way multiple choice problem; and A is the ground-truth answer. Given Q and C as inputs, large language models (LLMs) [13, 23, 121] generate a sequence of tokens $T = (t_1, t_2, \dots, t_n)$ to answer the question. Recent works like Chain-of-Thought (CoT) [125, 54] leverage prompt engineering in the context C to encourage models to generate the intermediate reasoning process in T , which benefits LLM performance across diverse reasoning tasks. In this case, T consists of a set of m intermediate reasoning steps, which we denote as $S = (s_1, s_2, \dots, s_m)$. Each step s_i can be represented by a subsequence of the generated tokens $(t_i, t_{r_i}) \subseteq T$. In much prior work, a generated solution is “correct” if and only if the predicted final answer in s_m matches the ground truth A , which we call **answer correct(ness)**.

We observe that for all cases where LLMs produce erroneous final answers, there exists at least one mistake among the intermediate reasoning steps S . Moreover, even when the final answer is correct, there might still exist some mistakes among S . This phenomenon, as illustrated in Tab. 2.1, occurs for all LLMs we tested, including state-of-the-art models such as ChatGPT [94]. Since later reasoning steps are conditioned on prior reasoning steps, these mistakes often initiate a snowball effect, causing subsequent mistakes to compound. This significantly diminishes the likelihood of correct problem-solving and impedes the progress

Table 2.1. An example question from GSM8K with a generated CoT reasoning path with GPT-3.5 (turbo), where the output provides a wrong reasoning chain with the correct answer.

Question: Jerry is rolling a six-sided die. How much more likely is it (expressed as a percentage) that he rolls a number greater than 3 than that he rolls two even numbers in a row?

Ground Truth: There are 3 numbers greater than 3 on the dice, so the chances of rolling one of them are $3 / 6 = 50\%$. The chance of rolling one even number is 50%, so the chance of rolling two in a row is $50\% * 50\% = 25\%$. The difference between these two probabilities is $50\% - 25\% = 25\%$. ##### 25

Output: There are two possible outcomes for rolling a number greater than 3 out of 6 total outcomes, so the probability is $2 / 6$ or $1 / 3$. For rolling two even numbers in a row, there are three possible outcomes (2-4, 4-6, 6-2) out of 36 total outcomes, so the probability is $3 / 36$ or $1 / 12$. To find the difference in probability, we subtract the probability of rolling two even numbers in a row from the probability of rolling a number greater than 3: $1 / 3 - 1 / 12 = 1 / 4$. This means that it is 25% more likely to roll a number greater than 3 than to roll two even numbers in a row.

towards achieving human-level complex reasoning.

Therefore, in this work, we place significant emphasis on ensuring the validity of every reasoning step, not just the correctness of the final answer. In particular, we focus on the validity of *deductive reasoning*, an essential component of a logical reasoning process. In deductive reasoning, we are given a (premise, conclusion) pair, and we are interested in determining whether the conclusion follows from the premises. In the context of reasoning-based QA tasks, for each reasoning step s_i , we define its *deductive validity* $V(s_i)$ as a binary variable. A reasoning step is **deductively valid** ($V(s_i) = 1$) if and only if s_i can be logically deduced from its corresponding premises p_i , which consist of the context C , the question Q , and all the previous reasoning steps $s_j (j < i)$. Then, we can also define the deductive validity for the entire reasoning chain S as $V(S) = \bigwedge_{i=1}^M V(s_i)$. Compared to evaluating answer correctness, which can be accomplished by simple functions such as exact string match, evaluating deductive validity is a lot more challenging. Thanks to the recent progress on LLMs, which demonstrate impressive in-context learning capabilities across diverse scenarios, we propose to use LLMs to examine reasoning chains and predict the deductive reasoning validity.

Table 2.2. Zero-shot reasoning chain verification accuracy for GPT-3.5-turbo (ChatGPT). To generate verification inputs, for each dataset, we perform Chain-of-Thought (CoT) prompting and randomly sample 50 resulting reasoning chains providing correct answers and 50 reasoning chains providing wrong answers. We then prompt the model with “Do you think the above reasoning process is correct?Let’s think step by step” such that the model outputs whether any mistake exists in the input reasoning process. We observe that when given an *entire* reasoning process, where the deductive graphs for all reasoning steps are entangled together, it is challenging even for strong language models like ChatGPT to verify its validity.

Answer Correctness	GSM8K	AQuA	MATH	AddSub	Date	Last Letters
Correct	0.98	0.96	1.00	0.98	0.98	1.00
Wrong	0.04	0.06	0.04	0.02	0.04	0.04
(Average)	0.51	0.51	0.52	0.50	0.51	0.52

2.4 Deductively Verifiable Chain-of-Thought Reasoning

In this section, we introduce our specific approaches to performing deductive verification of reasoning chains. Specifically, we first introduce our motivation and method for decomposing a deductive verification process into a series of step-by-step processes, each only receiving contexts and premises that are necessary. Then, we propose **Natural Program**, a natural language-based deductive reasoning format, to facilitate local step-by-step verification. Finally, we show that by integrating deductive verification with unanimity-plurality voting, we can improve the trustworthiness of reasoning processes along with final answers. An overview of our approach is illustrated in Fig. 2.1 and Fig. 2.2.

2.4.1 Decomposition of Deductive Verification Process

Given a reasoning chain $S = (s_1, s_2, \dots, s_n)$, a straightforward idea to verify its deductive validity is to ask LLMs to examine the *entire* reasoning chain at once. We thus conduct a preliminary experiment: for a dataset problem and its reasoning chain S generated by ChatGPT, we instruct ChatGPT to determine whether there exists any mistake among any reasoning step in S . Note that the model does not need to output the specific reasoning step that is mistaken, and only needs to output an overall “yes/no”. However, as demonstrated in Tab. 2.2, ChatGPT

struggles at finding out mistaken reasonings, and it persistently outputs "Correct" for most reasoning chain queries regardless of their actual validity.

We conjecture that such phenomenon is caused by the abundance of irrelevant premise for each reasoning step. Recall that a premise p_i for a reasoning step s_i consists of the question Q , the question context C , along with the prior reasoning steps $s_{\leq j} = \{s_j : j < i\}$. For Q and C , we can further extract and decompose $Q \cup C$ into a set of "question-related premises" $QC = \{qc_1, qc_2, \dots, qc_m\}$, where qc_i is a premise or condition inferred from $Q \cup C$. Then, it is often the case that most elements of $p_i = QC \cup s_{\leq j}$ are irrelevant to the validity of s_i , leading to erroneous language model. An example is illustrated in Appendix. A very recent work [110] also observes a similar phenomenon where LLMs are easily distracted by irrelevant context.

Therefore, we propose to decompose the reasoning chain verification process into a series of step-by-step processes, each receiving only the premises that are necessary. Recall that the validity of the entire reasoning chain is defined as $V(S) = \bigwedge_{i=1}^M V(s_i)$, so we can naturally decompose $V(S)$ into $\{V(s_i)\}$. For each $s_i \in S$, we would like to ensure that it explicitly lists the minimal subset of premises $\bar{p}_i \subseteq p_i$ necessary for its deductive reasoning. This motivates us to introduce a natural-language based deductive reasoning format in Sec.2.4.2.

2.4.2 Natural Program Deductive Reasoning Format

As previously mentioned in Sec. 2.4.1, we desire LLMs to output deductive reasoning processes that can be easily verified by themselves, specifically by listing out the minimal set of necessary premises p_i at each reasoning step s_i . To accomplish its goal, we propose to leverage the power of natural language, which is capable of rigorously representing a large variety of reasoning processes and can be generated with minimal effort. In particular, we introduce **Natural Program**, a novel deductive reasoning format for LLMs. More formally, Natural Program consists of the following components:

- An instruction for models to extract question-related premises QC . We use the following instruction: "First, let's write down all the statements and relationships

in the question with labels”.

- A numbered-list of question-related premises, each prefixed with “#{premise_number}”.
- An instruction for models to generate the reasoning chain S based on the question-related premises QC . We use the following instruction: “Next, let’s answer the question step by step with reference to the question and reasoning process”.
- A list of prefixed reasoning steps S_i . The prefix has the following format:
#{number} (by {list_of_premises_used}). Here “number” equals $|QC| + i$, and “list_of_premises_used” consists of numbers from the smallest subset of premises among $QC \cup s_{\leq j}$ that are used for the deductive reasoning of s_i . In addition, for the last reasoning step s_m , we ensure that it (1) includes a special tag `Final Step`; (2) refers to the premise number of the target question to be answered; (3) explicitly gives the final answer to a question.

Given that LLM’s reasoning outputs follow the Natural Program format, we can then verify the deductive validity of a *single* reasoning step s_i through the a prompt that consists of (1) the full descriptions of premises used for the reasoning of s_i ; (2) the full description of s_i ; (3) an instruction for validity verification, such as “Double-check the reasoning process, let’s analyze its correctness, and end with "yes" or "no".” Note that throughout this verification process, we only keep the minimal necessary premise and context for s_i , thereby avoiding irrelevant context distraction and significantly improving the validation efficacy.

An illustration of model’s deductive reasoning chain in the Natural Program reasoning format, along with the corresponding deductive verification process, is presented in Fig. 2.1. We also illustrate a one-shot prompt for Natural Program in the Appendix, which we use in our experiments. Through Natural Program, we will show that LLMs are capable of performing explicit, more rigorous, and more cogent deductive reasoning. Moreover, Natural Program facilitates LLMs to more effectively self-verify their reasoning processes, leading to better reliability

and trustworthiness in the generated responses.

2.4.3 Integrating Deductive Verification with Unanimity-Plurality Voting

Given that we can *effectively* verify a deductive reasoning process, we can naturally integrate verification with LLM’s sequence generation strategies to enhance the trustworthiness of both the intermediate reasoning steps and the final answers. In this work, we propose Unanimity-Plurality Voting, a 2-phase sequence generation strategy described as follows. Firstly, similar to prior work like [123], we sample k multiple reasoning chain candidates along with their final answers. In the unanimity phase, we perform deductive validation on each reasoning chain. Recall that a chain S is validated (i.e., $V(S) = 1$) if and only if all of its intermediate reasoning steps are validated (i.e., $\forall i, V(s_i) = 1$). For each intermediate reasoning step s_i , we perform majority voting over k' sampled single-step validity predictions to determine its final validity $V(s_i)$. We then only retain the verified chain candidates $\{S : V(S) = 1\}$. In the plurality voting stage, we perform majority-based final answer voting among the verified chain candidates. We will show that our approach improves the reliability and the correctness of final answers.

2.5 Experiments

In this section, we perform evaluations to demonstrate the effectiveness of our Natural Program-based deductive reasoning verification approach over diverse reasoning datasets. We first show that by integrating our deductive verification strategy with Unanimity-Plurality Voting (Sec. 2.4.3), we can simultaneously enhance answer correctness on challenging benchmarks. Next, we assess the accuracy of our deductive reasoning verification approach on reasoning chains.

2.5.1 Experimental Setup

Benchmarks. We evaluate the deductive verification accuracy and the answer correctness of reasoning chains over a diverse set of reasoning tasks: arithmetic reasoning, symbol manipulation, and date understanding. For arithmetic reasoning, we utilize the following benchmarks: 1) AddSub [46]; 2) GSM8K [25]; 3) MATH [44]; 4) AQuA [66]. Among these benchmarks, the AddSub and GSM8K datasets involve middle school-level multi-step calculations to arrive at a single number as the final answer. The MATH dataset presents more challenging problems that require expressing the answer as a mathematical expression in LaTeX format. These problems involve concepts from linear algebra, algebra, geometry, calculus, statistics, and number theory. AQuA also features similarly challenging problems, except that questions are in a multiple-choice format. For symbol manipulation, we use Last Letter Concatenation [125], where the model is tasked with concatenate the last letters of all the words provided in the question. For date understanding, we use the one from BIG-bench [114]

Deductive verification evaluation setup. For each of the above benchmarks, we select 100 reasoning chains, where 50 of them are deductively valid and 50 of them exhibit reasoning mistakes. The ground-truth deductive validity of each reasoning chain is determined by human annotators. To validate each reasoning step, we follow the format in Natural Program, except that in different experiments, the premises for a reasoning step s_i can be the entire $p_i = QC \cup s_{\leq j}$ or only the smallest subset of premises $\bar{p}_i \subseteq p_i$ necessary.

Answer extraction. To extract answers from reasoning solutions, we first perform text split based on answer prefix patterns such as "answer is" or "option is". Based on each problem type, we then use regular expressions to extract the final answer. To extract the validity results from deductive verification processes, we only keep the last sentence of model response. Then, we extract the validity answer with regular expressions to obtain attitude words, e.g., "yes" or "no". Sometimes, the language models may not provide a direct answer and instead output phrases like "not applicable" at the end of the response. In such cases, we set the answer from

Table 2.3. Final answer accuracy comparison on GPT-3.5-turbo (ChatGPT).

Methods	Arithmetic				Commonsense	
	GSM8K	AQuA	MATH*	AddSub	Date	Last Letters
CoT + Voting	87.62%	53.18%	35.93%	92.36%	69.97%	81.60%
Faithful CoT + Voting	75.80%	53.50%	31.78	88.35%	73.50%	-
Ours (Natural Program + NP + UPV)	87.05%	70.34%	36.75%	93.67%	72.49%	92.98%

the model as "yes". Please refer to Appendix for more details.

Model. We conduct our main experiments with GPT-3.5-turbo (ChatGPT) [94]. We also present results for the LLama model-family [121]) in Appendix. For ChatGPT, we use a generation temperature of $T = 0.7$. For Unanimity-Plurality Voting, we set $k = 10$ and $k' = 3$ by default. We use 1-shot prompting for both reasoning chain generation and deductive verification (except reasoning chain generation for the date understanding task where we use 2-shot). See Appendix for more details.

2.5.2 Results

Answer correctness. Following Sec. 2.4.3, we integrate our Natural Program (NP)-based deductive verification approach with Unanimity-Plurality Voting (UPV) with the hope of improving the reliability and correctness of final answers. As a reference, we also report the performance of Chain-of-Thought (CoT) [125] and Faithful CoT [78] on different reasoning tasks. For these baselines, we perform simple answer-based majority voting with $k = 10$ for fair comparison. Results are shown in Tab. 2.3. Though our major goal is to improve the trustworthiness and reliability of deductive reasoning, we also observe on-par or better final answer accuracy than baselines over reasoning tasks, demonstrating the effectiveness of our approach.

Reasoning chain deductive verification accuracy. We then evaluate the reasoning chain verification accuracy of our approach introduced in Sec. 2.4. A higher verification accuracy reflects better model reliability and trustworthiness. Results are shown in the second row of

Table 2.4. Deductive verification accuracy of reasoning chains for GPT-3.5-turbo (ChatGPT). To validate each reasoning step, we follow the format of our Natural Program, except that we either use the full premises $p_i = QC \cup S_{\leq j}$ or the minimal subset of premises $\bar{p}_i \subseteq p_i$ necessary. For each dataset, we randomly sample 50 reasoning chains that are deductively valid and 50 reasoning steps exhibiting incorrect reasonings.

Premise Context	Reasoning Correctness	GSM8K	AQuA	MATH	AddSub	Date	Last Letters
Full Premises	Correct	64%	54%	58%	95%	26%	96%
	Wrong	56%	68%	56%	24%	76%	5%
Minimal Premises	Correct	84%	72%	70%	95%	90%	96%
	Wrong	84%	62%	76%	40%	56%	6%

Tab. 2.4. We observe that our approach exhibits good reasoning verification accuracy across most datasets. Notably, our approach is capable of identifying erroneous reasoning in many faulty reasoning chains, while maintaining a low rate of false positives for reasoning chains that are already valid. Though, our approach’s efficacy is limited on tasks such as “Last Letters”. We hypothesize that this is due to the nature of the task, where each subsequent reasoning step is conditioned on all previous steps. This results in a significant number of dependent premises for later steps, presenting greater challenges for reasoning verification.

Ablation Study. In addition, we perform several ablation studies to gain further insights into our approach. In Tab. 2.4, we compare two different approaches to verify a single reasoning step $s_i \in S$ following our Natural Program format. The first approach utilizes all of p_i for verification regardless of their relevance to s_i , potentially introducing irrelevant contexts. The second approach follows our design in Sec. 2.4.1 and only includes the necessary context and premises for s_i . We observe that removing irrelevant premises significantly improves the reasoning chain verification accuracy on many datasets, highlighting the importance of this technique.

We also ablate on our Unanimity-Plurality Voting strategy by investigating the impact of different k' . Recall that k' determines the number of votes to produce validity predictions of single-step reasoning. Results are shown in Tab. 2.5. We observe that increasing k' generally enhances reasoning validation accuracy, though we note that this is at the expense of more

Table 2.5. Impact of different values of k' on the verification accuracy of reasoning chains using our Unanimity-Plurality Voting strategy. Experiments are performed on AddSub using GPT-3.5-turbo (ChatGPT).

Answer Correctness	$k' = 1$	$k' = 3$	$k' = 5$	$k' = 10$
Correct	86%	90%	90%	92%
Wrong	38%	38%	38%	40%

Table 2.6. ChatGPT generates the correct final answer but provides incorrect premise numbers for grounding the first reasoning step. In "Step 1", the correct premise numbers should be #1 and #2 for deduction. Our deductive reasoning verification approach effectively identifies these reasoning errors, enhancing the trustworthiness of the reasoning process. It is worth noting that removing a correct answer from the candidate reasoning chains can slightly impact the overall final answer correctness.

Question: In March it rained 0.81 inches. It rained 0.35 inches less in April than in March.

How much did it rain in April?

Ground Truth: It rained $0.81 - 0.35 = 0.46$ inches.

GPT-3.5-turbo (ChatGPT):

Question-related premises:

#1. It rained 0.81 inches in March.

#2. It rained 0.35 inches less in April than in March.

#3. How much did it rain in April?

The first reasoning step:

#4. (by #2) Step 1: Calculate how much it rained in April by subtracting the difference of 0.35 inches from the amount in March.

Amount of rain in April: $0.81 - 0.35 = 0.46$ inches

compute.

Qualitative Analysis. We perform qualitative analysis to illustrate the effectiveness of our deductive reasoning verification framework. Tab. 2.6 shows an example, where ChatGPT generates the correct final answer but provides incorrect premise numbers to ground the first reasoning step. We note that for many of such cases, our approach effectively identifies these reasoning errors, thereby enhancing the reliability and trustfulness of LLM’s reasoning processes. Though, it is worth noting that since we remove a correct answer from the candidate reasoning chains, this has a slightly negative impact on the overall final answer correctness.

Table 2.7. The term "pennies" in this question can be interpreted as either a type of coin or a unit of currency. In this particular question, "pennies" is treated as a type of coin. However, the initial reasoning step by ChatGPT mistakenly treats "pennies" as a unit of currency, resulting in the conversion of all Melanie's money into "pennies" (highlighted in red). Consequently, all subsequent reasoning steps follow this flawed logic, leading to an incorrect reasoning trace. Our deductive verification is not yet able to detect such errors.

Question: Melanie had 10 quarters and 17 pennies in her bank. Her dad gave her 27 pennies and her mother gave her 19 pennies. How many pennies does Melanie have now?

Ground Truth: Melanie have $17 + 27 + 19 = 63$ pennies.

ChatGPT's reasoning step:

#5. (by #1) Step 1: Calculate the number of pennies Melanie had initially.

Number of pennies in 10 quarters: $10 * 25 = 250$

Number of pennies initially: $250 + 17 = 267$

2.6 Additional Results

2.6.1 Zero-Shot vs. Few-Shot Full Reasoning Chain Verification Without Step-by-Step Decomposition or Natural Program Format

In Table 2 and Section 4.1, we demonstrated the impracticability of zero-shot verification of a complete reasoning chain S without decomposition. To complement this result, we conduct a few-shot reasoning chain verification experiment. We use two-shot examples generated by GPT-4, where one example verifies a correct reasoning chain and another example verifies an incorrect reasoning chain. However, as shown in Tab. 2.8, the verification accuracy is 50% for most datasets, and GPT-3.5 tends to output "yes" for the correctness of deductive reasoning all the time. Thus, it's still challenging to directly verify complex reasoning chains without decomposing them into step-by-step processes, even with two-shot demonstrations.

2.6.2 Zero-Shot vs. One-Shot Deductive Verification with Step-by-Step Decomposition and Natural Program Format

In this section, we conduct an ablation study where we use zero-shot vs. one-shot prompting for the deductive verification of reasoning chains using the process we introduced in Section 4.2 and Figure 1 of the main paper. In our one-shot prompting example (which is

Table 2.8. Comparison of reasoning chain verification accuracy for GPT-3.5-turbo with zero / two-shot prompting. The entire reasoning chain is verified at once without step-by-step decomposition and without our Natural Program (complementing our results in Table 2 of the main paper). For each of the 6 datasets, we perform Chain-of-Thought (CoT) prompting and randomly sample 50 resulting reasoning chains providing correct answers and 50 reasoning chains providing wrong answers for verification.

Prompting	Reasoning Correctness	GSM8K	AQuA	MATH	AddSub	Date	Last Letters
Zero-shot	Correct	0.98	0.96	1.00	0.98	0.98	1.00
	Wrong	0.04	0.06	0.04	0.02	0.04	0.04
	(Average)	0.51	0.51	0.52	0.50	0.51	0.52
Two-shot	Correct	0.98	0.96	1.00	0.92	1.00	0.96
	Wrong	0.02	0.04	0.0	0.06	0.26	0.06
	(Average)	0.50	0.50	0.50	0.49	0.63	0.51

Table 2.9. Comparison between zero-shot and one-shot accuracy of deductive verification for GPT-3.5-turbo (ChatGPT) with step-by-step decomposition and Natural Program format. For each dataset, we randomly sample 50 reasoning chains that are deductively valid and 50 reasoning steps exhibiting incorrect reasonings.

Method	Reasoning Correctness	GSM8k	AQuA	MATH	AddSub	Date	Last Letters	Overall
Zero-shot	Correct	84%	78%	90%	96%	90%	12%	75%
	Wrong	26%	12%	28%	20%	20%	80%	31%
	(Average)	55%	45%	59%	58%	55%	46%	53%
One-shot	Correct	84%	72%	70%	95%	90%	96%	85%
	Wrong	84%	62%	76%	40%	56%	6%	54%
	(Average)	84%	67%	73%	68%	73%	51%	69%

also used in our main paper), for more effective verification, we prompt the model to check the deductive validity from three perspectives: ungrounded information, erroneous reasoning, and wrong calculation. Any failure from any of these perspectives results in failure in verification result.

The zero-shot vs. one-shot comparison results are shown in Table 2.9. We observe that our one-shot prompt significantly improves the deductive verification accuracy. Notably, we discover many more reasoning errors for reasoning chains containing mistakes, while maintaining a low rate of false positives among reasoning chains that are already correct.

2.6.3 Deductive Verification with Vicuna Models

We further explore the efficacy of deductive verification for open-source models. We select two popular models: Vicuna-7B and Vicuna-13B [22]. These models are fine-tuned versions of LLaMA-7B and LLaMA-13B [121] using the ShareGPT data¹. We use the same one-shot verification method we used in Appendix 2.6.2 and Table 4 of the main paper. Results are shown in the first and the third rows of Table 2.10. We observe for the *original Vicuna models without finetuning*, Vicuna-7B exhibits poor performance in deductive verification and fails to find out reasoning mistakes, while the larger Vicuna-13B exhibits better verification accuracy.

We therefore conduct an additional experiment to investigate if the verification accuracy of Vicuna models can be improved by fine-tuning. To this end, we generate a deductive verification dataset, which consists of 2000 reasoning steps evenly distributed between correct and incorrect categories. We automatically generate this dataset using GPT-3.5-turbo since it exhibits a very high accuracy of single-step verification. We first use GPT-3.5-turbo to generate solutions for problems in GSM8K’s training set. We then execute step-by-step deductive verification on these solutions using GPT-3.5-turbo. For solutions that result in correct final answers, we retain the reasoning steps that pass deductive verification. For solutions that yield incorrect final answers, we retain the reasoning steps that cannot pass deductive verification. After constructing our dataset, we then fine-tune the Vicuna models using the verifications of the 2000 reasoning steps. Models were fine-tuned with 4 A100-80GB over 3 epochs. Training parameters are shown in Table 2.11.

As shown in Tab. 2.10, we observe that fine-tuning with our dataset can enhance the deductive verification accuracy of Vicuna models not only on the dataset where the training dataset is constructed (GSM8K), but also on many other datasets. However, the accuracy is still worse than non-finetuned GPT-3.5, which suggests that model capacity has a significant impact on deductive verification capabilities.

¹<https://github.com/domeccleston/sharegpt>

Table 2.10. One-shot Deductive Verification Accuracy of Vicuna-7B and Vicuna-13B. The models are evaluated without finetuning or with finetuning on our deductive verification dataset. For each dataset, we randomly sample 50 reasoning chains that are deductively valid and 50 reasoning steps exhibiting incorrect reasonings.

Models	Reasoning Correctness	GSM8K	AQuA	MATH	AddSub	Date	Last Letters	Overall
Vicuna-7B	Correct	80%	86%	96%	98%	96%	80%	89%
	Wrong	14%	22%	16%	6%	20%	34%	19%
	(Average)	47%	54%	56%	52%	58%	57%	54%
Vicuna-7B (fine-tuned)	Correct	68%	48%	46%	76%	46%	32%	53%
	Wrong	72%	86%	54%	60%	72%	68%	69%
	(Average)	70%	67%	50%	68%	61%	50%	61%
Vicuna-13B	Correct	86%	82%	92%	96%	72%	74%	84%
	Wrong	32%	36%	20%	20%	34%	30%	29%
	(Average)	59%	59%	56%	58%	53%	52%	57%
Vicuna-13B (fine-tuned)	Correct	74%	50%	56%	86%	72%	12%	58%
	Wrong	72%	76%	72%	68%	62%	96%	74%
	(Average)	73%	63%	64%	77%	67%	54%	66%
GPT-3.5	Correct	84%	72%	70%	95%	90%	96%	85%
	Wrong	84%	62%	76%	40%	56%	6%	54%
	(Average)	84%	67%	73%	68%	73%	51%	69%

Table 2.11. Hyperparameters for finetuning Vicuna models with our deductive verification dataset.

Hyperparameters	Value
Optimizer	AdamW
Learning rate	1×10^{-5}
Weight decay	0.00
Num epochs	3
Batch size	64
Learning rate schedule	Linear

2.7 Additional Implementation Details

In this section, we describe our process to extract the final answer from language models’ responses. The process begins by selecting the last three non-empty lines. Then, these lines are processed through the following pipeline:

1. Firstly, we use a list of regular expressions to identify "No-Answer" patterns within the text, such as "we cannot answer (this—the) question". This process helps us ascertain whether the model can provide a conclusive answer. If any such patterns appear in the text, we mark "No answer!" as the final answer. However, if we don't detect these patterns, we proceed to the next steps for extracting the final answer.
2. Secondly, if any "Answer-Split" patterns are found in the text, we divide the text into several blocks using the identified pattern. The last block of text is then utilized for extracting the answer.
3. Lastly, we use regular expressions, as outlined in Tab. 2.12, to scan the remaining text for possible final answers. If multiple matches are found for the pattern, we select the first match as the final answer. If no pattern matches are found in the remaining text, we default the final response to "No answer!".

"No-Answer" Patterns: "we cannot provide an answer to this question with (this|the) given information", "we cannot answer (this|the) question", "we cannot determine", "we can't determine", "we do not have enough information to answer (this|the) question", "we do not have enough information to provide a definitive answer to (this|the) question", "the answer(.*?)is unknown", "answer is not listed among the answer choices".

"Answer-Split" Patterns: "answer is", "final answer:", "answer to the question is", "answer to this question is", "concatenated letters are", "concatenate the letters -", "The answer of".

2.8 Limitations

While we have demonstrated the effectiveness of Natural Program-based deductive reasoning verification to enhance the trustworthiness and interpretability of reasoning steps and final answers, it is important to acknowledge that our approach has limitations. In this section,

Table 2.12. Regular Expression for extracting the final answers of different kinds of questions.

Answer Type	Regular Expression
Number	(-?\d[\d,\.]*)
Fractional number	(-?\d+(\d+\d+\d+)\d+ -?\d+\d+\d+)
Date	(\d\d\d\d\d\d\d\d)
Yes or No	(?:Yes No yes no NO YES)

we analyze a common source of failure cases to gain deeper insights into the behaviors of our approach. The failure case, as shown in Tab. 2.7, involves the ambiguous interpretation of the term “pennies,” which can be understood as either a type of coin or a unit of currency depending on the context. The ground truth answer interprets “pennies” as coins, while ChatGPT interprets it as a unit of currency. In this case, our deductive verification process is incapable of finding such misinterpretations. Contextual ambiguities like this are common in real-world scenarios, highlighting the current limitation of our approach.

2.9 Conclusion

In this paper, we aim to enable Large Language Models (LLMs) to perform explicit and rigorous deductive reasoning while ensuring the trustworthiness of their reasoning processes through self-verification. To this end, we have proposed a novel framework based on “Natural Program”, a natural language-based deductive reasoning format that facilitates reasoning verification and can be easily generated through in-context learning. Within this framework, we decompose the verification process of complex reasoning chains into step-by-step subprocesses that focus solely on necessary context and premises, allowing us to significantly enhance the accuracy of verification. Additionally, we introduce a Unanimity-Plurality Voting strategy to further improve verification accuracy and simultaneously enhance the correctness of final answers in complex reasoning tasks. Experimentally, we demonstrate the superiority of our framework in improving the rigor, trustworthiness, and interpretability of reasoning steps and answers.

Acknowledgements

Chapter 2, in full, is a reprint of the material published in the 2023 Neural Information Processing Systems (NeurIPS): “Deductive Verification of Chain-of-Thought Reasoning” (Zhan Ling*; Yunhao Fang*; Xuanlin Li; Zhiao Huang; Mingu Lee; Roland Memisevic; Hao Su). The dissertation author was the primary investigator and author of this paper.

Chapter 3

Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving

Large Language Models (LLMs) have achieved tremendous progress, yet they still often struggle with challenging reasoning problems. Current approaches address this challenge by sampling or searching detailed and low-level reasoning chains. However, these methods are still limited in their exploration capabilities, making it challenging for correct solutions to stand out in the huge solution space. In this work, we unleash LLMs’ creative potential for exploring multiple diverse problem solving strategies by framing an LLM as a *hierarchical policy* via in-context learning. This policy comprises of a *visionary leader* that proposes multiple diverse high-level problem-solving tactics as hints, accompanied by a *follower* that executes detailed problem-solving processes following each of the high-level instruction. The follower uses each of the leader’s directives as a guide and samples multiple reasoning chains to tackle the problem, generating a solution group for each leader proposal. Additionally, we propose an effective and efficient *tournament-based approach* to select among these explored solution groups to reach the final answer. Our approach produces meaningful and inspiring hints, enhances problem-solving strategy exploration, and improves the final answer accuracy on challenging reasoning datasets across different domains, such as the MATH dataset and the STEM subjects from MMLU.

3.1 Introduction

Large language models (LLMs) [13, 23, 121, 91] have demonstrated remarkable potential across a myriad of disciplines such as common sense understanding [43, 114] and code generation [17, 61]. Yet, LLMs often struggle with challenging reasoning tasks, such as writing mathematical proof and solving advanced reasoning problems. These tasks are inherently creative, as the path to a solution isn't immediately evident, requiring the exploration of numerous problem-solving tactics before discovering a successful path towards the end goal.

While recent works have investigated enhancing LLMs' exploration ability in problem solving through sampling and search [123, 133, 10], these approaches still exhibit considerable limitations. Before we describe such limitations, let's think of how humans approach mathematical proofs: one typical methodology is that we begin by connecting the target proof statement to our prior experiences such as proofs with similar routines (e.g., divide-and-conquer) or relevant techniques (e.g., root-finding). From this reservoir of knowledge and familiarity, humans try multiple **“high-level”** proof tactics that possess the potential to reach the goal, and subsequently develop detailed **“low-level”** proof details based on them. It should be noted that the quality of “high-level” strategies and thinking processes can exert a substantial impact on the effectiveness, efficiency, and likelihood of successfully solving these problems, as illustrated in Tab. 3.1. In cognitive science, such advanced higher-order thinking skills are referred to as *Metacognition* [30, 84, 9]. It is widely acknowledged that metacognition ability leads people to effective problem-solving strategies and successful task completion [117, 2].

Compared to human exploration of complex problem solution spaces, the aforementioned sampling and search methods in NLP have primarily focused on delving into the detailed, “low-level” reasoning steps, often overlooking the “high-level” strategies and cognitive processes. We, therefore, aspire to unleash LLMs' potential for creative exploration of high-level tactics and hints, enabling them to tackle challenging reasoning problems with similar ingenuity and proficiency as humans.

To this end, we draw inspiration from the concept of a “hierarchical policy” in the decision-making literature [7, 64, 55], and we propose to define LLM as a **hierarchical policy** for problem solving, which consists of a visionary high-level leader policy and a low-level follower policy. In our framework, the high-level leader establishes connections between the target problem and the LLM’s extensive knowledge and prior problem-solving experiences. It leverages this information to propose various high-level problem-solving tactics and directions for exploration. The low-level follower policy then utilizes each of these hints as an in-context guidance throughout the detailed step-by-step problem-solving processes. Furthermore, we desire implementations of this idea to be achieved through minimal effort. Indeed, this can be achieved by leveraging off-the-shelf pretrained LLMs and in-context learning. Finally, after we obtain an array of diverse reasoning chains through LLM’s creative exploration process, we propose an effective and efficient tournament-based method to select among these chains to reach the final answer. An overview of our approach is shown in Fig. 3.1.

Experimentally, we demonstrate that our high-level leader policy is capable of exploring and producing meaningful and inspiring hints and guidance for the low-level follower policy, thereby making it easier for correct reasoning chains and answers to stand out. Our reasoning chain selection approach effectively identifies desired reasoning chains, enhancing the final answer accuracy on challenging reasoning tasks. Our key contributions are as follows:

1. To effectively explore expansive solution spaces in complex problems, we propose framing language models as a hierarchical policy, encompassing both “high-level” and “low-level” cognitive processes, facilitating a more efficient exploration of distinct high-level ideas.
2. Within our hierarchical policy, we present two effective approaches for the visionary high-level leader to generate diverse tactics and hints that guide the low-level follower during exploration.
3. We propose an effective and efficient tournament-based approach for selecting desired reasoning chains among those generated during exploration, facilitating the attainment of the final answer.

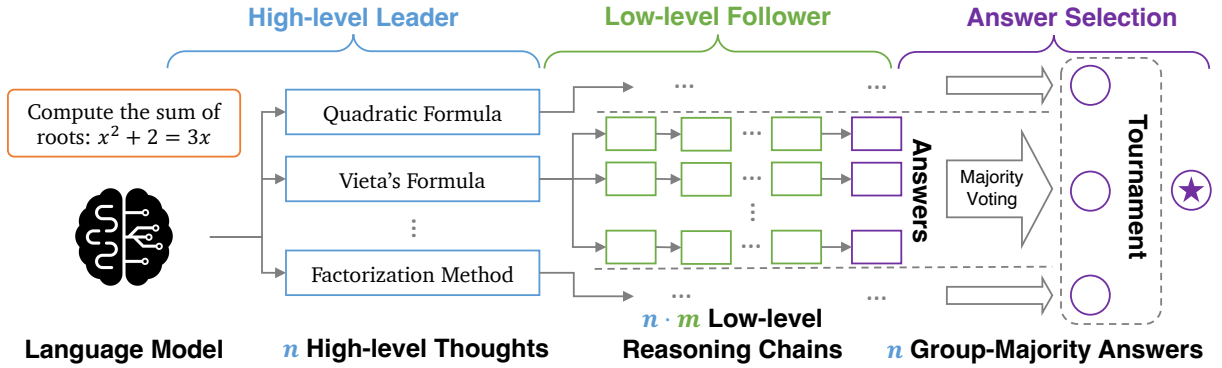


Figure 3.1. Overview of our approach, which frames language models as a hierarchical policy for exploration. The visionary high-level leader connects the target problem with the LLM’s knowledge, proposing multiple diverse tactics and hints for exploration. The low-level follower leverages these hints as in-context guidance to execute detailed problem-solving processes. Finally, we employ an tournament-based approach to select the desired reasoning chains and reach the final answer.

4. Experimentally, we demonstrate that our approach produces high-level hints and guidance that are meaningful and inspiring, enhances problem-solving strategy exploration, leads to better discovery and visibility of correct solutions, and improves the final answer accuracy on challenging reasoning problems across different datasets and domains, such as MATH and the STEM subjects from MMLU.

3.2 Related Work

Reasoning and Exploration with Language Models. Recent large language models (LLMs) have demonstrated remarkable potentials in solving complex reasoning tasks. A key strategy is to encourage LLMs to generate detailed step-by-step reasoning processes through in-context learning techniques, including but not limited to Chain-of-Thought (CoT) prompting [125] and numerous other approaches [54, 144, 141, 113, 123, 145, 75, 143, 134]. For many challenging reasoning tasks such as mathematical problem solving, proof writing, and inductive reasoning, it is often challenging for LLMs to obtain the correct solution in a single attempt. Therefore, to further enhance LLMs’ problem solving capabilities, it is highly beneficial to encourage LLMs to search and explore over diverse problem-solving strategies and reasoning

Table 3.1. An example problem from the MATH dataset, which can be solved through two distinct high-level approaches: (1) Identify when the discriminant is a perfect square; (2) Apply Vieta’s formulas and the factorization trick. The first high-level approach does not effectively reduce the problem into a simpler form, as solving for all a where $a^2 - 20a$ is a perfect square remains a highly challenging task, which causes the problem solving process to become stuck. Conversely, adopting the second strategy leads to an elegant solution.

Question: For how many integer values of a does the equation $x^2 + ax + 5a = 0$ have integer solutions for x ?

Ground Truth Solution (High level approach: Vieta’s Formulas + Factorizing Trick): Suppose the roots of the quadratic are given by m and n . Note that $(x - m)(x - n) = x^2 - (m + n)x + mn = x^2 + ax + 5a$, and setting coefficients equal, it follows that $m + n = -a$ and $mn = 5a$ (This also follows directly from Vieta’s formulas.) Notice that the a can be canceled by either dividing or noting that $0 = 5a + 5 \cdot (-a) = mn + 5(m + n)$. Simon’s Favorite Factoring Trick can now be applied: $mn + 5m + 5n + 25 = (m + 5)(n + 5) = 25$. It follows that $m + 5$ and $n + 5$ are divisors of 25, whose pairs of divisors are given by $\pm\{(1, 25), (5, 5), (25, 1)\}$. Solving, we see that (m, n) is in the set $\{(-4, 20), (0, 0), (20, -4), (-6, -30), (-10, -10), (-30, -6)\}$. However, the two pairs of symmetric solutions yield redundant values for a , so it follows that the answer is **4**.

Example Reasoning from GPT-4 (High level approach: Discriminant is a perfect square):

First, we note that the discriminant of a quadratic equation must be a perfect square for the roots of the equation to be integers. The discriminant of our equation $x^2 + ax + 5a = 0$ is $a^2 - 20a$. Setting this equal to k^2 for some integer k , we get $a^2 - 20a - k^2 = 0$. This is a quadratic equation in a , and its roots are $a = 10 \pm \sqrt{100 + k^2}$. Again, for a to be an integer, $\sqrt{100 + k^2}$ must be an integer. Let $\sqrt{100 + k^2} = m$. Then $m^2 = 100 + k^2 \implies k^2 = m^2 - 100$. This means that $m^2 - 100$ must be a perfect square. The possible values of m which satisfy this condition are $m = 10, 11, 12, 13, 14, 15, 20$. For each of these 7 values of m , we get 2 values of a (one with the $+$ sign and one with the $-$ sign). However, we have to be careful with $m = 10$, because in this case $k = 0$, and we get only one value of $a = 10$. So, the total number of integer values of a for which the equation $x^2 + ax + 5a = 0$ has integer solutions for x is $2 \times 6 + 1 = \mathbf{13}$.

steps. Recent works like [11, 118, 132, 27] perform step-by-step search to construct deductive natural language proofs given premises. [123] samples and explores multiple reasoning chains, then performs majority voting to obtain the final answer. [128, 65, 67] introduce verification filtering to the reasoning chain exploration. [133] and [10] perform tree-based and graph-based search over reasoning steps with backtracking and refinement. However, much prior work limits reasoning exploration to specific and detailed reasoning steps, and the high-level strategies and thinking processes are often overlooked. In this study, we frame LLMs as hierarchical policies, enhancing problem-solving by exploring diverse high-level strategies and tactics.

Hierarchical Policy. The concept of hierarchical policy was originally proposed in rein-

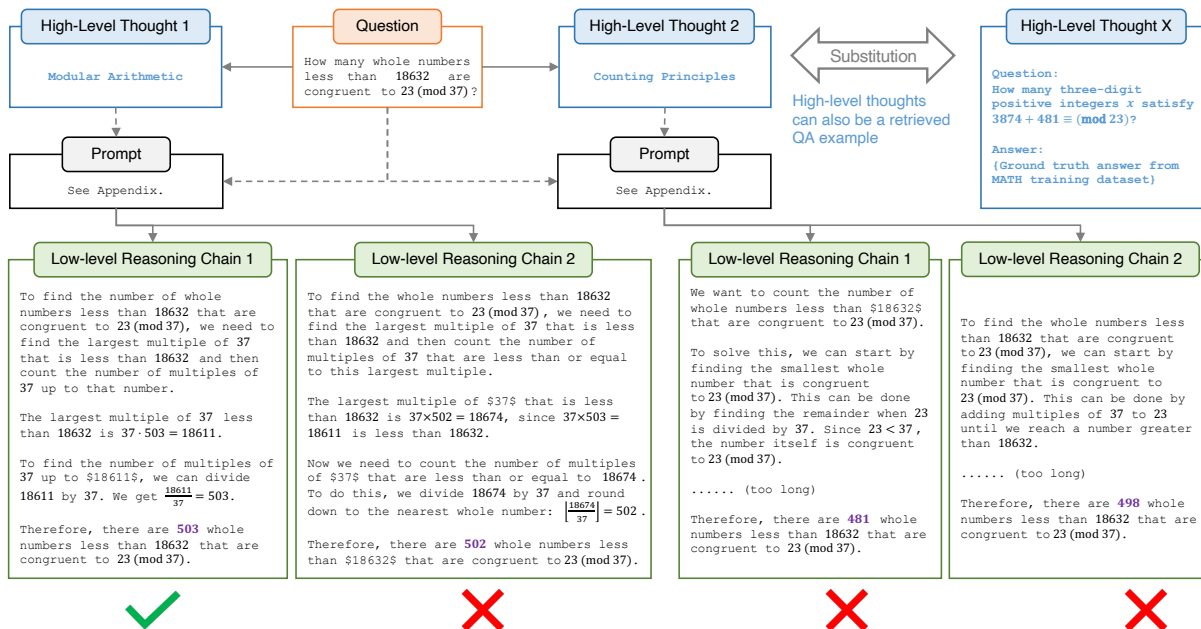


Figure 3.2. A detailed illustration of our approach that frames language models as a hierarchical policy for exploration. In this example, “Modular Arithmetic” is highly relevant to the target question, and the low-level follower successfully finds the correct answer in some generated reasoning chains. On the other hand, “Counting Principles” is irrelevant to the target question, and the low-level follower struggles to reach the correct solution. Later, we propose an effective and efficient approach to select the desired reasoning chains from those generated by the low-level follower.

forcement learning (RL) and imitation learning (IL) as a multi-level decision-making approach to tackle complex, long-horizon tasks [7, 64, 55, 90, 42, 99]. In hierarchical RL, a high-level policy proposes latent features and subgoals that guide low-level policy actions. Prior work has also investigated enhancing the exploration abilities of hierarchical policies [59, 37, 98, 48]. However, few prior works has framed LLMs as hierarchical policies through in-context learning to improve their exploration capabilities in problem solving, which is the main focus of our work.

3.3 Language Model as a Hierarchical Policy for Exploration

A natural language reasoning process can be defined as a tuple (Q, T, A) , where Q is the target question; A is the ground-truth answer in the format of a number, an answer choice,

or a statement to proof or conclude; T is the set of locally-valid reasoning chains that reach the ground-truth answer, i.e., $T = \{\tau = (\tau_1, \tau_2, \dots, \tau_s) : \text{valid}(\tau) = 1, \tau_1 = Q, \tau_s = A\}$. A large language model (LLM), denoted as π , takes a question Q and a prompt P as input to generate a step-by-step reasoning chain $\tau = \pi(P, Q)$ that attempts to solve the problem. In the quest to improve LLMs’ exploration abilities in problem solving, much prior work focuses on exploring, sampling, and searching for specific reasoning steps [123, 133, 10]. Yet, these methods tend to neglect the higher-order cognitive processes inherent in human problem solving. Successful problem-solving often relies on a guiding strategy and hint, and overlooking this aspect could potentially lead to inefficient and ineffective exploration.

To address these limitations, we propose to formulate LLM as a **hierarchical policy** $\pi = (\pi_{high}, \pi_{low})$ for problem solving through in-context learning. Following the convention of Markov Decision Process, π , π_{high} , and π_{low} are probabilities over token sequences. The visionary high-level leader policy π_{high} takes in a question Q and a prompt P_{high} as input and returns a set of possible high-level tactics and hints $H = \{h_1 \dots h_n\}$, where $H \sim \pi_{high}(P_{high}, Q)$ (we emphasize that a sample from π_{high} returns a *tactic set* instead of a single tactic; we will also use tactic / hint interchangeably from here on). Then, the low-level follower policy π_{low} utilizes *each* h_i as an in-context guidance to execute specific problem-solving processes by sampling or searching reasoning chains, yielding a **group** of reasoning chains $T_i = \{t_{i,1}, \dots, t_{i,m}\}$, where $t_{i,j} \sim \pi_{low}(h_i, Q)$.

To instantiate our hierarchical approach, there are **two crucial design components** we need to address: **(1)** How to encourage the leader π_{high} to generate appropriate tactics and hints H that serve as the guidance for the follower π_{low} ; **(2)** Given groups of reasoning chains $\{T_i\}_{i=1}^n$ generated by π_{low} , how to effectively and efficiently choose the desired reasoning chains to obtain the final answer.

Generating high-level tactics and hints for exploration. Given a question Q , our goal is to encourage the leader π_{high} to effectively establish the connection between Q and relevant language model knowledge, from which it proposes high-level hints and directions that holds

significant potential for reaching the goal. We would also like to limit irrelevant hints that potentially mislead the low-level policy, since previous work [110] has shown that LLMs can be brittle to irrelevant context. We therefore propose **two approaches** for π_{high} to generate high-level problem-solving tactics and hints (an illustration in Fig. 3.2):

(a) **Prompt** an LLM, such as GPT-4, to generate a list of relevant techniques and concepts for a target question. We aim for these hints to be clear, concise, and easily interpretable, e.g., “*Angle Bisector Theorem*”, “*Minimization with Derivatives*”, such that they can serve as effective guidance for the low-level policy π_{low} . See Appendix for detailed prompts.

(b) Use a sequence-embedding language model, such as SentenceBERT [102], to **retrieve** a set of relevant problems with their step-by-step solutions. Each relevant problem and solution then inspires π_{low} to utilize similar tactics and strategies when exploring and generating reasoning chains.

Probabilistic Interpretation: Next, we build a connection between our method and hierarchical policies in the Markov Decision Process (MDP) framework, and we use the MDP framework to explain the improved exploration ability. To make it intuitive, we use Fig. 3.3 to illustrate the idea. In the low-level reasoning chain space $T = \{t_1, t_2, \dots\}$ given Q , we group reasoning chains based on the high-level tactics they employ. For example, $t_{1,1}$ and $t_{1,2}$ both employ tactic h_1 . Here, the size of the region corresponds to the marginal conditional probability of h given Q when we sample the low-level reasoning chain **without** providing a high-level tactic prompt. We denote this marginal probability as $\Pr(h|Q)$. Note that the tactic with the highest $\Pr(h|Q)$ may *not* necessarily lead to the correct reasoning chains. This should not be counter-intuitive, especially for hard reasoning problems that require out-of-the-box thinking. In practice, for a specific reasoning question that receives an incorrect answer from GPT-3.5/4, we have observed that the generated reasoning chains frequently rely predominantly on a *single* tactic. Instead, our leader-follower strategy takes two steps to generate the low-level reasoning

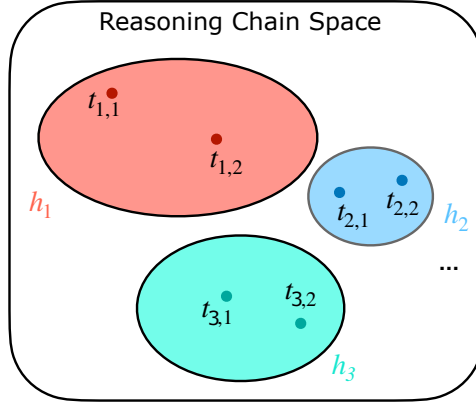


Figure 3.3. Illustration of the partitioning of the reasoning chain space based on the high-level tactics employed in the solution.

chain, which can be formulated using the marginal probability formula:

$$\Pr(A|Q) = \sum_h \Pr(A|h, Q) \cdot \text{Unif}(h \in H) \cdot \Pr(H|Q)$$

Here, $\Pr(A|h, Q)$ corresponds to π_{low} , $\Pr(H|Q)$ corresponds to π_{high} , and $\text{Unif}(h \in H)$ denotes a uniform distribution among $h \in H$. Note that $\text{Unif}(h \in H) \cdot \Pr(H|Q) \neq \Pr(h|Q)$. To illustrate with Fig. 3.3, suppose $H = \{h_1, h_2, h_3\}$ and $\Pr(H; \pi_{high}) = 1$, then sampling by $\Pr(h_i|Q)$ corresponds to sampling by the area of h_i , whereas sampling by $\text{Unif}(h \in H) \cdot \Pr(H|Q)$ corresponds to uniformly sampling among h_1, h_2 , and h_3 , i.e., regarding them as if they were having the same area. **For general cases, our strategy samples all the different hints returned by π_{high} with equal probabilities.** Our strategy, therefore, aligns with the spirit of common practice in reinforcement learning to encourage the exploration behavior by making the density of actions more uniform (e.g., ϵ -greedy policy reduces the chance of the best action and increases the chance of worse actions).

Effectively and efficiently selecting final reasoning chains. Given n high-level tactics and hints $\{h_i\}_{i=1}^n$ proposed by the leader π_{high} , the low-level follower policy π_{low} produces n groups of reasoning chains $\{T_i\}_{i=1}^n$. Throughout our experiments, we maintain a constant size of reasoning chains for all groups, i.e., $\forall i, |T_i| = m$. As it is a common scenario that not all of the

suggested tactics are relevant to problem solving, and irrelevant hints could make the low-level policy more susceptible to reasoning mistakes, we would like to *effectively* select the desired reasoning chains among those generated by the low-level policy. We would also like to make the selection process *efficient*, reducing the need to invoke a large number of LLM calls.

To this end, we propose the following **tournament-based approach** to select reasoning chains: Within each group of reasoning chains T_i , we conduct majority voting to establish the group-majority answer A_i . Then, for every group, we randomly select a *single* reasoning chain from those that reach the group-majority answer, and we add it to a “selection” set S (as a result, $|S| = n$). Next, denote the “final” reasoning chain as τ_{final} , which we initialize as first reasoning chain of S . We then initiate a “tournament”: Over $n - 1$ iterations, for each iteration i , we prompt GPT-4 to compare the current τ_{final} with the $(i + 1)$ -th reasoning chain in S to determine which is better. If the latter is better, we set it as τ_{final} . Empirically, we also gather comparison results through majority voting conducted over k repetitions.

The above approach requires a small number of $n \times k$ language model calls to select a desired reasoning chain. For instance, when we have $n = 4$ reasoning groups, each containing $m = 16$ reasoning chains, and we perform $k = 3$ comparison repetitions, it takes only 12 calls to GPT-4.

3.4 Experiments

In this section, we perform quantitative and qualitative evaluations to demonstrate the effectiveness of our approach. We first investigate whether our approach successfully enhances the discovery and visibility of correct solutions, introducing a quantitative metric for this assessment. Subsequently, we assess whether our approach improves final answer accuracy in challenging reasoning problems across various domains including mathematics and science. Lastly, we conduct ablation studies over tournament-based answer selection and further analyze our approaches and discuss its limitations.

Experiment setup. We conduct our primary analysis on the MATH dataset [43]. We also demonstrate our approach’s effectiveness on three of most challenging STEM subjects (high-school chemistry, college physics and machine learning) from the MMLU dataset [43], as well as GSM8K (refer to Appendix 3.5.2), which includes simpler mathematics problems compared to the MATH dataset.

For the MATH dataset [43], we adopt its **Level-5** test set, i.e., the hardest subset of questions, to evaluate different approaches. For the high-level policy π_{high} , we adopt the two approaches outlined in Sec. 3.3: (1) prompt GPT-4 to generate at most n relevant hints and tactics for a target question; (2) use SentenceBERT to retrieve n most-relevant problems from the MATH training set. For the low-level policy π_{low} , we adopt either GPT-3.5 or GPT-4. Unless otherwise specified, we establish the following parameters: $n = 4$ high-level hints (or reasoning groups) per question, $m = 16$ generated reasoning chains per group, and $k = 1$ comparison repeats for our tournament-based reasoning chain selection process (which is performed using GPT-4). We set temperature to be 0.3 during reasoning chain selection and 0.7 otherwise.

Our evaluation set of MATH questions is constructed as follows: For experiments in Sec. 3.4.1 and the first half of Sec. 3.4.2, we randomly sample 20 questions for each of the 7 categories in the MATH Level-5 test set, resulting in an evaluation set of 140 questions. We opted for this smaller evaluation set due to the cost associated with evaluating our approach when either GPT-3.5 or GPT-4 serves as π_{low} , which amounted to approximately \$150 for these 140 questions. Evaluating GPT-4 as π_{low} on the full test set would be very expensive. Later, in the second half of Sec. 3.4.2, we employ the *entire* subset of MATH Level-5 questions (excluding those requiring visual understanding) to assess our approach when GPT-3.5 serves as π_{low} . The findings remain consistent.

3.4.1 Do We Enhance the Discovery and Visibility of Correct Solutions?

In this section, we investigate whether by framing LLMs as a hierarchical policy and encouraging them to explore multiple diverse problem-solving tactics and hints, we improve

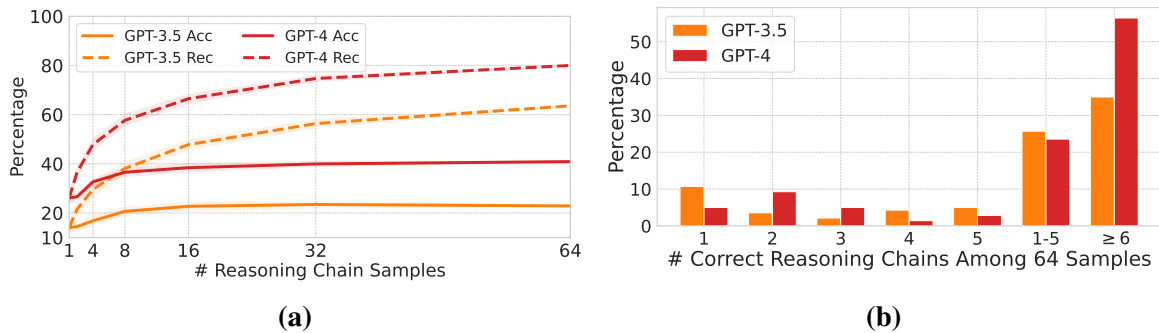


Figure 3.4. Statistics for the “CoT Sampling + Majority Voting” baseline. **(a)** Accuracy and recall as a function of the number of sampled reasoning chains. **(b)** Percentage of questions where $x \in \mathbb{N}$ of the 64 sampled reasoning chains are correct.

the discovery and visibility of desired solutions that reach the correct answers. We compare our approach with the Chain-of-Thought (CoT) Sampling [125] + Majority Voting [123] baseline, which samples the same amount of detailed reasoning chains as ours and performs majority voting to obtain the final answer.

We would like to assess the “visibility” of the correct answers among solutions generated along the exploration process. A correct answer is “visible” if it not only exists in at least one of reasoning chains, but also occupies a substantial proportion of them, even though it might not be the majority answer. An intuitive way of measurement is to compare the accuracy and recall of correct answers between our approach and the baseline. **However, we find that standard metrics like accuracy and recall do not perfectly align with our goal.** In particular, for our CoT Sampling baseline, as the number of reasoning chains goes up, the recall steadily improves, but the final-answer accuracy after majority voting plateaus after a few reasoning chains, as illustrated in Fig. 3.4a. This suggests that the standard recall metric poorly correlates the prominence of correct answers. On the other hand, the standard accuracy metric does not reflect the scenarios where LLM identifies correct solutions by a reasonable chance (and does so multiple times), but these correct answers become submerged during majority voting. Such scenarios often occur, and we illustrate this phenomenon in Fig. 3.4b. Therefore, the visibility of desired solutions is not adequately captured by the standard accuracy or recall metric.

Table 3.2. Comparison of the Grouped-Majority Recall (“GMR”) and the final answer accuracy (“Acc.”) between the CoT sampling + Voting baseline and our two exploration approaches outlined in Sec. 3.3. We use GPT-3.5 or GPT-4 as the language model for the CoT Sampling + Voting Baseline, along with the low-level follower policy in our approaches. Metrics are obtained using $n = 4$ reasoning groups and $m = 16$ reasoning chains per group on our 140-question MATH Level-5 evaluation set.

Model	Method	Alg.		Count.		Geom.		Int. Alg.		Num. Th.		Prealg.		Preal.		Overall	
		Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR
GPT-3.5	CoT Sampling + Voting	30.00	46.67	25.00	32.83	10.00	11.67	15.00	17.50	5.00	29.84	60.00	69.58	10.00	10.31	22.14	31.19
	ToT + Voting	15.00	-	0.00	-	5.00	-	20.00	-	0.00	-	49.17	-	0.00	-	13.45	-
	Ours - Prompt for Tactics	40.00	41.67	30.00	32.83	15.00	19.38	0.00	15.00	20.00	35.00	65.00	75.00	5.00	10.00	25.00	32.69
	Ours - Retrieval	50.00	50.00	25.00	31.98	15.00	15.31	0.00	15.00	30.00	45.31	70.00	72.50	5.00	10.00	27.85	34.23
GPT-4	CoT Sampling + Voting	50.00	62.50	36.25	55.67	47.50	50.00	10.00	24.83	55.00	61.67	75.00	88.75	12.50	12.00	40.89	50.78
	Ours - Prompt for Tactics	65.00	65.00	35.00	52.83	40.00	51.25	15.00	29.31	70.00	79.08	80.00	91.25	5.00	12.50	44.29	54.46
	Ours - Retrieval	50.00	67.50	45.00	45.00	40.00	53.50	30.00	31.00	50.00	60.17	70.00	95.00	10.00	10.00	42.14	51.73

We thus invent a “microscope” to inspect the visibility of the correct answers. **We propose the following “Grouped-Majority Recall” metric to better quantify the visibility of correct solutions.** The calculation takes two steps. First of all, recall that our proposed method produces n groups each containing m reasoning chains. The CoT Sampling baseline can be viewed as a special case of our method with $n = 1$ group of empty high-level tactics. To calculate the Grouped-Majority Recall metric, we first perform majority voting in each group to obtain n majority answers, and then we calculate the percentage of questions whose ground truth answer exist in *at least* one of the n group-majority answers. Sometimes, a group will contain multiple majority answers, in which case we calculate the expected value of our metric. To compare our approach with the CoT Sampling baseline, we randomly partition $n \times m$ CoT samples into n groups and calculate our metric.

Intuition behind the Grouped-Majority Recall: In contrast to the standard accuracy metric, rare correct answers that might be obscured amidst all the $n \cdot m$ samples could emerge as the majority in one of the groups, thereby being recognized by our new metric. This occurrence, as noted in our observations, is not uncommon: when a high-level tactic is accurate yet seldom sampled, it frequently yields a substantial number of correct low-level solutions that can dominate in a group, despite remaining a minority among all samples. Our new metric aptly acknowledges

such instances. Additionally, *in contrast to the standard recall metric*, for a correct answer to be recognized by our new metric, it should not only appear in at least one reasoning chain, but also take up the majority in a group, necessitating its appearance in multiple reasoning chains.

Results. We report the Grouped-Majority Recall metrics in Tab. 3.2 (“GMR” columns). We find that our exploration approaches effectively enhance Grouped-Majority Recall when either GPT-3.5 or GPT-4 serves as our low-level policy, demonstrating that our methods improve the discovery and visibility of solutions leading to the correct answers. Additionally, we observe that among our two exploration approaches, using concise technique-based hints underperforms using retrieved problem-solution hints when GPT-3.5 is used as the follower, but outperforms the latter for GPT-4. We conjecture that this is caused by the stronger ability for the GPT-4 follower to understand and effectively utilize the hints in specific problem-solving processes. On the other hand, for the weaker GPT-3.5 follower, even if the high-level hints are already inspiring and meaningful, it may not effectively utilize the hint to solve the target problem.

3.4.2 Do We Improve Final Answer Accuracy for Challenging Reasoning Problems?

Next, we investigate whether our exploration and tournament-based reasoning chain selection approach enhance the final answer accuracy on challenging reasoning problems. We first conduct an experiment on our 140-question Level-5 test set of the MATH dataset, and we compare our approach with the CoT Sampling + Majority Voting baseline in Tab. 3.2 (“Acc.” columns). We find that both of our exploration approaches successfully improve the final answer accuracy, demonstrating that our approach effectively selects among the explored reasoning chains to retain the high-quality ones.

We also implement Tree-of-Thoughts (ToT) [133] for mathematics problem solving following the original paper, and present its results on GPT-3.5 as a reference¹. We perform breadth-first search (BFS) in ToT, where at each step we expand 8 children and keep the best

¹Running ToT is especially expensive and costs over \$100 on GPT-3.5. Running ToT on GPT-4 would incur thousands of dollars of expense, so we would like to leave it for future work.

5 candidates at each depth level. We limit the tree depth to 16. However, we observe that the final accuracy for ToT is significantly worse than the CoT Sampling + Voting baseline. Upon further analysis, we find that the average number of reasoning chains ToT produces is 8.41, which is significantly fewer than the 64 reasoning chains in our baseline, potentially harming its performance.

Evaluation on a larger set of MATH questions. As stated in our experiment setup, our experiments in Sec. 3.4.1 and Sec. 3.4.2 were conducted using a smaller sample of 140 questions from the MATH Level-5 test-set **due to the cost associated with evaluating GPT-4**. In this section, we expand our evaluation by using GPT-3.5 as the low-level follower π_{low} and evaluating on a larger set of 1047 questions from the MATH Level-5 test set. This evaluation set encompasses all questions from the MATH Level-5 test set, except those that include Asymptote (a Vector Graphics Language) code in the question or answer, as these questions require visual comprehension. Additionally, unlike the 140-question evaluation set, which has an equal distribution of 20 questions from each of the 7 categories, the 1047 questions evaluated in this section have an uneven distribution among categories.

We present the Grouped-Majority Recall and the final answer accuracy results in Tab. 3.3. The findings are consistent with those we obtained in Sec. 3.4.1 and Sec. 3.4.2. We also observe that the overall Grouped-Majority Recall and the final answer accuracy are higher than those we obtained on the 140-question evaluation set, which is due to the higher portion (429 / 1047) of algebra and prealgebra questions in our 1047-question evaluation set, which are considered easier.

Evaluation on STEM subjects from the MMLU dataset. Furthermore, to showcase our approach’s broad applicability beyond mathematics, we conduct experiments using three of most challenging STEM subjects (high-school chemistry, college physics and machine learning) from the MMLU dataset [43]. Results are shown in Tab. 3.4a. We find that our approach continues to achieve better final-answer accuracy and grouped-majority recall, demonstrating the versatility and the effectiveness of our approach across diverse domains.

Table 3.3. Comparison of the Grouped-Majority Recall (“GMR”) and the final answer accuracy (“Acc.”) on our 1047-question MATH Level-5 evaluation set using GPT-3.5 as the language model for the CoT Sampling + Voting baseline along with the low-level follower policy in our approaches.

Method	Alg.		Count.		Geom.		Int. Alg.		Num. Th.		Prealg.		Precal.		Overall	
	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR	Acc.	GMR
CoT Sampling + Voting	56.73	65.84	26.57	39.97	16.00	19.33	6.56	11.06	23.27	38.75	60.74	69.11	4.49	4.74	32.22	40.63
Ours - Prompt for Tactics	55.00	64.77	33.96	42.84	24.00	26.00	8.61	12.23	32.47	42.47	63.76	71.91	6.74	11.40	35.15	42.58
Ours - Retrieval	57.86	69.81	29.25	40.00	20.00	22.25	11.07	15.26	31.82	43.43	66.44	72.65	5.62	8.87	36.10	44.29

Table 3.4. (a) Comparison of the Grouped-Majority Recall and the final answer accuracy on three of the challenging STEM subjects from the MMLU dataset. We adopt GPT-3.5 as the low-level follower, and we sample 32 reasoning chains per problem. For “CoT + Voting”, we directly perform majority final-answer voting over the 32 reasoning chains. For “Ours - Prompt for Tactics”, we adopt $n = 4$ groups, each having $m = 8$ reasoning chains. (b) Final answer accuracy of WizardMath-7B-V1.1 on our 140-question MATH level-5 evaluation set. We adopt WizardMath-7B-V1.1 as the low-level follower, and we sample 32 reasoning chains per problem with the forementioned setup.

Method	h.-s. chem.		c. phys.		m. learning	
	Acc.	GMR	Acc.	GMR	Acc.	GMR
CoT Sampling + Voting	65.02	75.12	66.48	78.02	54.46	63.39
Ours - Prompt for Tactics	73.89	80.75	75.82	84.52	65.18	70.98

(a) Results of three STEM subjects from MMLU

Method	CoT Sampling	Ours - Tactics	Ours - Retrieval
Answer Accuracy	9.88	10.71	13.57

(b) Results of WizardMath-7B-V1.1 on our 140-question MATH Level-5 evaluation set.

Is our approach still effective on smaller, open-source models? Previously, we conducted our experiments on proprietary language models, including GPT-3.5 and GPT-4. In this section, we further investigate whether the effectiveness of our approach generalizes to smaller and open-source models. We conduct our experiments on WizardMath-7B-V1.1 [77], a finetuned version of Mistral-7B-v0.1 [52]. Results are shown in Tab. 3.4b. We find that our approaches continue to achieve better final-answer accuracy, demonstrating the effectiveness of our approach across different model scales.

Table 3.5. Ablation on using **(a)** different models and different k (numbers of comparison repetitions); **(b)** different temperatures (T) during our tournament-based reasoning chain selection.

Selection	Ours - Retrieval						CoT Sampling
	Tourn. w/ GPT-3.5			Tourn. w/ GPT-4			Voting
k	1	3	5	1	3	5	-
Accuracy	21.43	22.86	27.14	27.85	27.85	27.85	22.14

(a) Ablation on using different models and different k

Temperature	$T = 0$	$T = 0.3$	$T = 0.7$	$T = 1.0$
Answer Accuracy	27.14	27.85	27.85	27.85

(b) Ablation on using different temperatures (T)

3.4.3 Ablation Study

Ablation on tournament-based reasoning chain selection. We perform ablation studies on different design decisions of our tournament-based reasoning chain selection approach, where we adopt GPT-3.5 for the low-level policy and compute the metric over our 140-question evaluation set. Specifically, **a)** we conduct an experiment where we use either GPT-3.5 or GPT-4 to perform our tournament selection process, while varying the value of k , the number of comparison repetitions, in each of iteration of the tournament. Results are shown in Tab. 3.5a. We find that when k is small, using GPT-4 for tournament-based reasoning chain selection leads to a significant improvement compared to using GPT-3.5. Notably, GPT-4 demonstrates strong performance as a reasoning chain selector even with $k = 1$, resulting in a noteworthy enhancement of final answer accuracy over the CoT Sampling baseline. **b)** We also conduct an experiment where we vary the decoding temperature (T) of GPT-4 during our tournament-based reasoning chain selection process ($k = 1$ in this experiment). Results are shown in Tab. 3.5b. We observe that when $T \in [0, 1]$, different tournament selection temperatures have little effect on the final answer accuracy.

Does final answer accuracy improvement come from our hierarchical policy framework and / or our tournament selection process? We perform an ablation experiment where

Table 3.6. Effect of majority voting and our tournament-based reasoning chain selection on the final-answer accuracy of the CoT + Sampling baseline and our hierarchical policy approaches. For “Voting over Samples”, we directly perform majority voting over the 64 sampled reasoning chains per problem. For “Voting over Groups” and “Tournament”, we adopt $n = 4$ groups, each having $m = 16$ reasoning chains. We use GPT-3.5 for low-level follower and the CoT Sampling Baseline.

Method	CoT Sampling	Ours - Tactic	Ours - Retrieval
Voting over Samples	22.14	21.79	23.57
Voting over Groups	19.70	20.24	23.39
Tournament	22.86	25.00	27.85

we investigate the impact of our hierarchical policy framework and our tournament-based reasoning chain selection process in improving the final answer accuracy. Results are shown in Tab. 3.6. We find that **a)** For our hierarchical policy approaches, adopting our tournament-based reasoning chain selection process yields better final answer accuracy than using majority voting, *demonstrating the effectiveness of our tournament selection process*. Intuitively, this is because not all of the tactics and hints produced by our high-level leader are helpful, and some of them might mislead the low-level follower, potentially causing it to generate a consistent but wrong answer under a misleading high-level guidance. By evaluating reasoning chains using our tournament-based selection approach, we effectively remove those that exhibit reasoning mistakes and keep those that are more trustful. **b)** Additionally, for the CoT + Sampling baseline, adopting our tournament-based reasoning chain selection process does not outperform our approaches that employ the hierarchical-policy framework. This shows that our hierarchical policy plays a significant role in enhancing LLM’s ability to solve challenging problems.

Cost analysis. We present a cost analysis in Appendix. Our approach does not incur much extra cost compared to baseline approaches like CoT Sampling + Majority Voting.

3.5 Additional Results

3.5.1 Failure Example Analysis

In this section, we present failure cases of our approach. This include cases where the low-level follower fails to effectively follow the high-level tactics (Tab. 3.7)), as well as cases where the high-level policy fails to provide informative tactics (Tab. 3.9) and demonstrations (Tab. 3.8).

Table 3.7. A failure case in which the GPT-4 high-level leader generates a highly-relevant hint for the question, but the GPT-4 low-level follower involves incorrect intermediate steps when following the high-level hint the and results in wrong answer.

Question: Suppose that the minimum value of $f(x) = \cos 2x - 2a(1 + \cos x)$ is $-\frac{1}{2}$. Find a .

High-Level Hint: Trigonometric Identities

Guidance: Use principles of “Trigonometric Identities” to approach this question. Ensure you think step-by-step, presenting a clear, logical solution pathway.

GPT-4 Follower Output: First, we can use the double angle formula $\cos 2x = 1 - 2\sin^2 x$ to express $f(x)$ all in terms of $\sin x$: $f(x) = 1 - 2\sin^2 x - 2a(1 + \cos x) = 1 - 2\sin^2 x - 2a + 2a\cos x$. Then use the identity $\sin^2 x + \cos^2 x = 1$ to express $f(x)$ in terms of $\sin x$: $f(x) = 1 - 2\sin^2 x - 2a + 2a(1 - \sin^2 x) = 1 - 2\sin^2 x - 2a + 2a - 2a\sin^2 x$. Simplify: $f(x) = 1 - 2a - 2\sin^2 x(1 + a) = 1 - 2a - 2(1 + a)\sin^2 x$. This is a quadratic in $\sin x$, and we want to find the minimum. This occurs when the derivative is 0, or when $\sin x = 0$. However, we want the minimum value of $f(x)$, so we need to consider the endpoints of the interval $[-1, 1]$ for $\sin x$. When $\sin x = \pm 1$, $f(x) = 1 - 2a - 2(1 + a)(\pm 1)^2 = 1 - 2a - 2(1 + a) = -a - 1$. We are given that the minimum value of $f(x)$ is $-\frac{1}{2}$, so $-a - 1 = -\frac{1}{2}$, which gives $a = \boxed{\frac{1}{2}}$.

3.5.2 Evaluation on GSM8K

We also investigate the efficacy of our approach on easier datasets as GSM8K. Results are shown in Tab. 3.10. We find that while GSM8K features easier mathematics problems than the MATH Level-5 questions used in our previous experiments, our approach continues to achieve better final-answer accuracy. This suggests that our approach retains its effectiveness across datasets with varying degrees of difficulty.

Table 3.8. A failure case in which a very similar question along with its ground truth solution (from the MATH training set) are retrieved and padded as prompt for the GPT-3.5 low-level follower. The solution of the retrieved example is not closely related to the new question given, and thus the low-level follower fails to produce a correct final answer.

Retrieved Question: The least common multiple of two numbers is 3780, and the greatest common divisor is 18. Given that one of the numbers is 180, what is the other number?

Answer: We use the identity $\gcd(a, b) \cdot \text{lcm}[a, b] = ab$ for all positive integers a and b . We are told that $\gcd(a, b) = 18$, and $\text{lcm}[a, b] = 3780$, so $ab = 18 \cdot 3780$. If one number is 180, then the other number is $18 \cdot 3780 / 180 = \boxed{378}$.

Question: What is the smallest integer greater than 200 for which the greatest common divisor of that integer and 36 is 18?

GPT-3.5 Follower Output: Let the integer be x . We are given that $\gcd(x, 36) = 18$. We know that $\gcd(a, b) \cdot \text{lcm}[a, b] = ab$ for all positive integers a and b . Since $\gcd(x, 36) = 18$, we have $\text{lcm}[x, 36] = \frac{36 \cdot 18}{18} = 36$. The least common multiple of two numbers is always greater than or equal to the maximum of the two numbers. Therefore, $\max(x, 36) = 36$. Since we want to find the smallest integer greater than 200, we want to find the smallest value of x such that $x > 200$. Since $\max(x, 36) = 36$, we have $x = 36$. However, $x = 36$ is not greater than 200, so we need to find the next smallest value of x that satisfies $\max(x, 36) = 36$. The next smallest value of x is $x = \boxed{216}$, which is indeed greater than 200. Pred Answer: 216

Table 3.9. A failure case in which the GPT-4 high-level leader generates an irrelevant hint, resulting in wrong answer.

Question: Given that $n > 1$, what is the smallest positive integer n whose positive divisors have a product of n^6 ?

High-Level Hint: Arithmetic Mean-Geometric Mean Inequality (AM-GM Inequality)

Guidance: Use principles of “Arithmetic Mean-Geometric Mean Inequality (AM-GM Inequality)” to approach this question. Ensure you think step-by-step, presenting a clear, logical solution pathway.

GPT-4 Follower Output: Let’s denote the positive divisors of n as d_1, d_2, \dots, d_k . We are looking for the smallest n for which $\sqrt[k]{d_1 d_2 \dots d_k} = n^2$ (since the geometric mean of the divisors is equal to n^2). Applying the AM-GM inequality, we have $\frac{d_1 + d_2 + \dots + d_k}{k} > \sqrt[k]{d_1 d_2 \dots d_k}$, which simplifies to $\frac{n + d_2 + \dots + d_k}{k} > n^2$, where $d_1 = n$ (since n is a divisor of itself). Rearranging the inequality, we have $d_2 + \dots + d_k > k \cdot n^2 - n$. We know that d_2, \dots, d_k are all positive integers, and they are all divisors of n , so the smallest they can be is 1. Therefore, we have $k > k \cdot n^2 - n + 1$. Since k is the number of divisors of n , we know it must be at least 2 (since every positive integer has at least two divisors: 1 and itself). Therefore, we have $2 > 2 \cdot n^2 - n + 1$, which simplifies to $n^2 - n + 1 \leq 1$. Rearranging the inequality, we have $n^2 - n \leq 0$, which simplifies to $n(n - 1) \leq 0$. Since $n > 1$, we have $n - 1 > 0$, so we must have $n \leq 0$, which is a contradiction. Therefore, no such n exists. However, if we relax the condition that n must be an integer, the smallest possible value of n is $\frac{1}{2}$, which occurs when n is a perfect square and all of its divisors are equal to \sqrt{n} . So, the smallest integer n whose positive divisors have a product of n^6 is $\boxed{1}$, because $1^6 = 1$ and the product of the divisors of 1 is 1.

Table 3.10. Final answer accuracy on GSM8K. We adopt GPT-3.5 as the low-level follower, and we sample 32 reasoning chains per problem. For "CoT + Voting", we directly perform majority final-answer voting over the 32 reasoning chains. For tournament, we adopt $n = 4$ groups, each having $m = 8$ reasoning chains.

Method	CoT + Voting	CoT + Tourn.	Ours - Retrieval + Tourn.
Answer Accuracy	88.45	88.85	89.91

3.6 Additional Implementation Details

We use "gpt-3.5-turbo-0613" and "gpt-4-0613" as the GPT-3.5 and GPT-4 version used throughout the experiments in our paper. We set the decoding temperature to 0.3 for tournament-based reasoning chain selection and 0.7 otherwise (i.e., for hint generation from the high-level leader, reasoning chain generation from the low-level follower, along with the baselines).

Table 3.11. Cost comparison between our approach and the CoT + Sampling Baseline on our 140-question MATH Level-5 evaluation set. We generate $n \times m = 4 \times 16 = 64$ reasoning chains per question. GPT-4 is utilized to perform tournament selection in our approaches.

Method Low-Level Follower	CoT Baseline	Ours - Tactics	Ours - Retrieval	CoT Baseline	Ours - Tactics	Ours - Retrieval
	GPT-3.5	GPT-3.5	GPT-3.5	GPT-4	GPT-4	GPT-4
Sampling	10.72	11.71	9.83	204.16	191.11	188.47
Tournament	None	2.42	4.45	None	4.52	5.92
Total	10.72	14.12	14.28	204.16	195.63	194.39

Table 3.12. Comparison on the number of input and output tokens per-question between our approach and the CoT + Sampling Baseline on our 140-question MATH Level-5 evaluation set. We generate $n \times m = 4 \times 16 = 64$ reasoning chains per question. GPT-4 is utilized to perform tournament selection in our approaches. Tournament tokens are included in the table.

Method Low-Level Follower	CoT Baseline	Ours - Tactics	Ours - Retrieval	CoT Baseline	Ours - Tactics	Ours - Retrieval
	GPT-3.5	GPT-3.5	GPT-3.5	GPT-4	GPT-4	GPT-4
Avg. #Input Tokens Per Question	0.11K	0.73K	1.88K	0.11K	0.88K	2.00K
Avg. #Output Tokens Per Question	38.20K	41.61K	34.32K	24.25K	22.85K	22.14K

3.7 Further Analysis and Limitations

In this section, we provide a further analysis into the success and failures of our approach, and we discuss the potential limitations of our approach along this process.

As shown in Fig. 3.2, we observe that our high-level leader policy is capable of producing many insightful and inspiring hints, even though it sometimes produces irrelevant hints. To further analyze the quality of generated high-level hints, we perform an experiment, where we aggregate all the high-level techniques and concepts produced by our first hint generation approach into a set. For each hint, we obtain its corresponding problem and ground truth answer, and then prompt GPT-4 to generate 10 “ground-truth hints” given both information. Subsequently, we calculate the percentage of hints that match at least one of its corresponding “ground-truth hints”, serving as a measure of hint quality. We find that **94.3%** of hints are among the ground-truth hints, highlighting the effectiveness of our approach in proposing pertinent, insightful hints.

Next, we delve into the analysis of common sources of failure cases to gain a deeper understanding of our approach’s behavior (due to space constraints, we present the detailed examples in Appendix. One significant factor contributing to these failures is that, despite the high-level leader producing hints being both inspiring and meaningful, the low-level follower may not closely follow these hints to solve the target problem, resulting in reasoning errors. Sometimes, it might even ignore the hints. Furthermore, even when the follower effectively incorporates the hints into its problem-solving processes, reasoning errors can still occur. The inconsistent adherence to high-level hints and the dependence on the capabilities of the follower highlight the current limitations of our approach.

3.8 Conclusion

In this work, we propose to frame LLMs as a hierarchical policy to effectively explore the expansive solution spaces in challenging mathematical reasoning problems. Our hierarchical

policy framework consists of a visionary “high-level” leader policy, which establishes connections between the target problem and the language model knowledge to generate hints, along with a “low-level” follower policy that leverages these hints as an in-context guidance throughout detailed problem-solving processes. Within this framework, we introduce two effective approach for the high-level leader policy to generate a diverse set of problem-solving tactics and hints for exploration. Additionally, we present an effective and efficient tournament-based method to select desired reasoning chains among the explored ones to attain the final answer. Through experiments, we demonstrate that our approach enhances problem-solving strategy exploration, improves the discovery and visibility of correct solutions, and enhances the final answer accuracy on challenging problems across different datasets and domains, such as the MATH dataset and the STEM subjects from MMLU.

Acknowledgements

Chapter 3, in full, is a reprint of material from the publicly available preprint: “Unleashing the Creative Mind: Language Model As Hierarchical Policy For Improved Exploration on Challenging Problem Solving” (Zhan Ling; Yunhao Fang; Xuanlin Li; Tongzhou Mu; Mingyu Lee; Reza Pourreza; Roland Memisevic; Hao Su). The dissertation author was the primary investigator and author of this paper.

Chapter 4

Long-Context Reasoning Synthesis through Context Expansion

Large language models (LLMs) have demonstrated remarkable progress in understanding long-context inputs. However, benchmarks for evaluating the long-context reasoning abilities of LLMs fall behind the pace. Existing benchmarks often focus on a narrow range of tasks or those that do not demand complex reasoning. To address this gap and enable a more comprehensive evaluation of the long-context reasoning capabilities of current LLMs, we propose a new synthetic benchmark, **LongReason**, which is constructed by synthesizing long-context reasoning questions from a varied set of short-context reasoning questions through context expansion. LongReason consists of 794 multiple-choice reasoning questions with diverse reasoning patterns across three task categories: reading comprehension, logical inference, and mathematical word problems. We evaluate 21 LLMs on LongReason, revealing that most models experience significant performance drops as context length increases. Our further analysis shows that even state-of-the-art LLMs still have significant room for improvement in providing robust reasoning across different tasks. We will open-source LongReason to support the comprehensive evaluation of LLMs' long-context reasoning capabilities.

Table 4.1. Comparison of LongReason with other long-context benchmarks. LongReason offers controllable context lengths and incorporating diverse and realistic tasks without the need for human annotation on long text.

Benchmark	Avg Len	Light Human Effort	Realistic Tasks	Broad Tasks	Controllable Context
ZeroSCROLLS [108]	~10K	✗	✓	✓	✗
L-Eval [3]	~8K	✗	✓	✓	✗
BAMBOO [32]	~16K	✗	✓	✓	✗
LongBench [8]	~8K	✗	✓	✓	✗
LooGLE [60]	~20K	✗	✓	✓	✗
InfiniteBench [139]	~200K	✗	✓	✓	✗
Loong [122]	~250K	✗	✓	✓	✗
Needle-in-a-haystack [53]	any	✓	✗	✗	✓
RULER [47]	any	✓	✗	✓	✓
LongReason (Ours)	any	✓	✓	✓	✓

4.1 Introduction

In recent years, large language models (LLMs) [91, 101, 5, 83, 52, 120] have demonstrated remarkable advances in diverse natural language processing tasks. The ability to comprehend and reason over long inputs is essential for downstream applications, including multi-turn conversations [119], document understanding [81] retrieval-augmented generation [134, 131], and language agents [142, 140]. Meanwhile, extensive efforts in deep learning system [29, 28, 21, 100] research have been devoted to optimizing computational overhead to support increasing numbers of input tokens, which has led to growing attention on long-context LLMs. Now, both proprietary and open-source LLMs can support up to millions of input tokens [101, 86, 38].

However, despite the rapid development of long-context language models, benchmarks have lagged behind. One of the key challenges is dataset construction, as long-context question-answering data is relatively scarce on the internet. To address this, prevalent long-context benchmarks have utilized synthetic tasks like passkey retrieval [88], needle-in-a-haystack (NIAH) [53,

142], and variable tracking [47] to evaluate long-context LLMs. However, these tasks are often unrealistic and involve reasoning processes that differ significantly from those in real-world applications. Alternatively, some research efforts have involved human annotation of realistic questions and gold answers over one or multiple long documents [33, 122, 60]. However, creating realistic long-context tasks from extensive texts is both challenging and time-consuming, even for human experts [122]. This limitation restricts the expansion of datasets to accommodate arbitrary context lengths and the ability to support controllable context. As shown in Table 4.1, existing benchmarks either rely on a limited number of synthetic tasks, demand significant human effort to read long contexts, or lacking controllable contexts and support for arbitrary context lengths. Furthermore, existing datasets [33, 122, 60] often utilize documents from specific domains, such as financial reports or legal cases, as input, which can inherently limit the diversity of task categories. Consequently, they tend to focus on a narrow set of tasks, such as comparison or classification, rather than evaluating more complex and challenging tasks that require chain-of-thought reasoning.

To address these challenges, we introduce a new long-context reasoning benchmark, LongReason, featuring diverse and realistic reasoning tasks to assess the long-context reasoning abilities of LLMs. To create the dataset efficiently and effectively, we first had human annotators collect short reasoning questions from the internet, cleaning them to avoid data contamination and forming the seed dataset. This seed dataset contains reasoning questions with diverse patterns from three major task categories: reading comprehension, logical inference, and mathematical word problems. We chose to use multiple-choice problems for easy evaluation, avoiding the use of LLMs or inaccurate metrics like Rouge score and F1 to assess the correctness of reasoning. Then, we utilize an automatic pipeline that synthesizes multi-hop long-context reasoning questions from the collected short-context problems. To ensure quality, we leverage LLMs to automatically verify the generated questions, ensuring they retain the same logic as their shorter counterparts. Ultimately, we retain 794 questions that pass these checks. For each question, we can generate long-context versions of arbitrary lengths; however, since most existing models support contexts

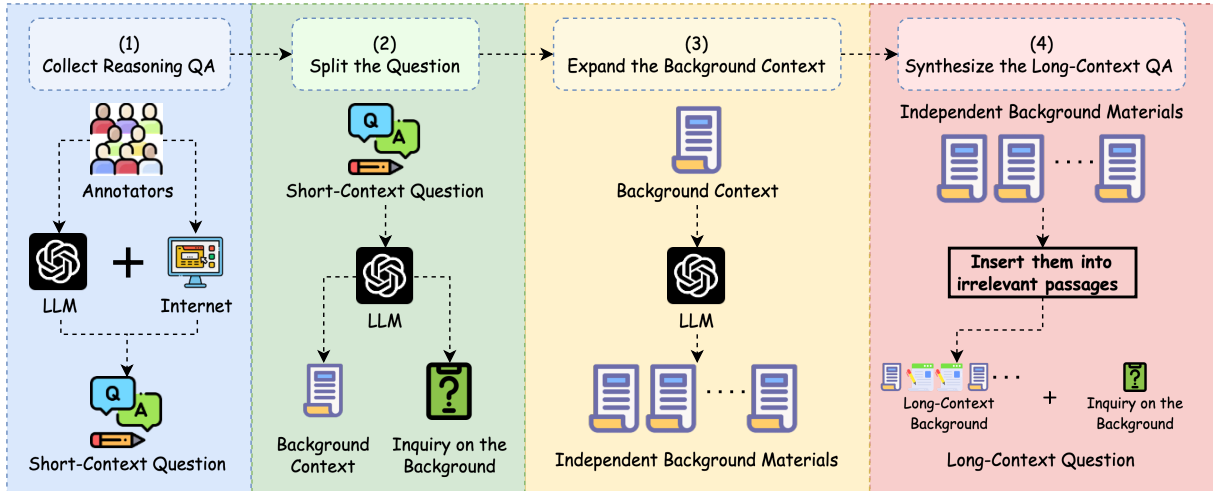


Figure 4.1. Overview of our pipeline for constructing LongReason. Given a short reasoning question Q_{short} , the pipeline first separates it into a background context C_{short} and a final question I . Next, multiple paragraphs are synthesized from the background context C_{short} . These synthesized paragraphs are then embedded within irrelevant passages to create a long-context background. Finally, the constructed context is combined with the final question to generate the long-context reasoning question Q_{long} .

up to 128K tokens, we focus our evaluation within this limit. This synthetic pipeline supports converting one short reasoning question into different lengths, enabling fine-grained assessment of LLMs across various context lengths and reasoning tasks.

To assess the current progress in the long-context reasoning abilities of existing LLMs, we evaluated 21 models of varying scales and architectures, sourced from both open-source and closed-source communities. While most of these models achieve near-saturated performance on previous synthetic tasks such as NIAH, nearly all exhibit significant performance degradation on LongReason as the context length increases. Further analysis reveals that even state-of-the-art LLMs show varying degrees of performance decline across different task categories, underscoring the importance of evaluating diverse reasoning tasks to fully understand the long-context reasoning capabilities of LLMs.

Our key contributions are summarized as follows:

- We present LongReason, a new synthetic long-context reasoning benchmark that encompasses a diverse range of task categories and supports controllable context lengths.

- We propose an innovative synthesis algorithm that generates long-context reasoning questions from existing short questions, reducing the need for labor-intensive human annotation for long-context data.
- We perform an extensive analysis of current LLMs, benchmarking their performance in long-context reasoning and offering valuable insights to enhance long-context reasoning capabilities.

4.2 Related Work

Long-Context Large Language Models Recent advancements in deep learning system have significantly propelled the development of long-context large language models (LLMs). One of the key challenges in scaling these models is the quadratic time and space complexity inherent in computing self-attention over long sequences. To mitigate this computational burden, efficient self-attention algorithms [29, 28, 71] have been introduced, reducing memory overhead, and novel training methods [62, 21] facilitate the training of these long-context models. As Rotary Position Embedding (RoPE) [115] is widely used for positional encoding in many open-source models [83, 120, 85], recent research [18, 130, 97, 72, 31, 146] has focused on adapting RoPE from pre-trained short-context models to effectively handle longer sequences. Moreover, new architectures [39, 96, 16, 15, 12] have been developed to efficiently process long-context inputs. Consequently, state-of-the-art language models [93, 92, 101, 4, 83, 85, 120, 38] now support context windows ranging from 128K to millions of tokens, enabling the exploration of reasoning abilities over extensive contexts with LLMs.

Long-Context Benchmarks As the context window of current LLMs expands rapidly, numerous benchmarks have been proposed to evaluate their capabilities. In early benchmarks such as ZeroSCROLLS [108], L-Eval [3], BAMBOO [33], LongBench [8], and LooGLE [60], the average input length remains under 25K tokens, which is far shorter than the context window size supported by existing LLMs. Recently, some research has begun to explore using synthetic

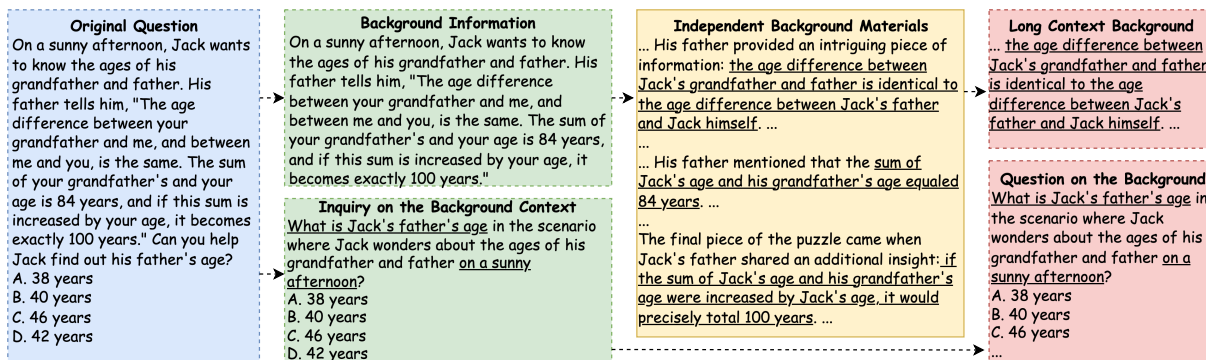


Figure 4.2. An illustrative example in LongReason. The original question is first decomposed into a separate background passage and an inquiry based on it. The inquiry includes keywords such as “Jack’s father’s age” and a time reference like “on a sunny afternoon” from the background passage, ensuring a clear connection to the passage. Subsequently, the background passage is expanded into multiple independent materials while preserving these key keywords. Finally, these independent materials are combined with some unrelated passages to create the final long-context reasoning question.

datasets, which can support controllable context lengths, to evaluate the long-context abilities of LLMs. Needle-in-a-Haystack and its variants [53, 142] primarily evaluate retrieval abilities by inserting relevant information into extensive irrelevant corpora and testing the LLMs’ capacity to extract it. Additionally, RULER [47] constructs synthetic tasks based on code-like flexible configurations to assess LLM performance over long contexts. While synthetic tasks can support the evaluation of arbitrarily long contexts, they are limited in scope, focusing on a narrow set of tasks and failing to comprehensively evaluate the reasoning abilities of LLMs in realistic scenarios. Other benchmarks like InfiniteBench [139] and Loong [122] use human annotations to create questions from given long texts, which contain more diverse tasks but are both time-consuming and costly. Our proposed benchmark, LongReason, focuses on evaluating the long-context reasoning abilities of LLMs, which are created automatically from short reasoning questions without heavy human effort in reading the long context. We conduct a detailed comparison with existing benchmarks in Table 4.1.

4.3 Our Benchmark: LongReason

In this section, we provide a detailed overview of LongReason, our synthetic long-context reasoning benchmark. This includes the problem formulation, the dataset construction process, and an analysis of the statistics of LongReason.

4.3.1 Long-context Reasoning Question Construction via Context Expansion

Problem Formulation The primary goal of LongReason is to assess the long-context reasoning abilities of LLMs. To achieve this, we first define the reasoning task as follows: Given a reasoning question Q , LLMs need to reason over Q to produce a reasoning chain S that leads to the final answer A . In this work, the focus is on scenarios where the question Q can be divided into a background context C and a final inquiry I based on that context, denoted as $Q = (C, I)$. In LongReason, the context C can be long, comprising multiple paragraphs from diverse sources, while only a small subset of the information in the context C is directly relevant to answering I . To simplify evaluation, LongReason employs close-ended multiple-choice questions for I . The dataset construction begins with a set of questions $\mathbf{Q}_{\text{short}}$, consisting of questions Q_{short} with relatively short question statements. For each Q_{short} , our proposed context expansion pipeline utilizes LLMs to automatically generate a long-context version of the question, $Q_{\text{long}} = (C_{\text{long}}, I)$. The detailed construction pipeline is illustrated in Figure 4.1.

Short-Context Reasoning Question Collection We begin by asking human annotators to create a dataset $\mathbf{Q}_{\text{short}}$, comprising short questions Q_{short} across various domains and diverse task categories. Annotators collect example questions from the internet and utilize an LLM to refine these questions, ensuring they are free from data contamination. To ensure that each short question requires reasoning, we prompt an LLM to evaluate the number of reasoning steps in its corresponding ground-truth reasoning chain, denoted as \bar{S} . We include only those questions that require at least two reasoning steps to arrive at the final answer in LongReason, thereby filtering

out straightforward common-sense problems that lack significant reasoning depth.

Automatic Short-Context Reasoning Question Decomposition with LLMs For each short reasoning question Q_{short} , we prompt an LLM to decompose the question into a background context C_{short} and an inquiry I . This decomposition needs to ensure that the final inquiry I is clearly linked to the background context C_{short} , enabling the LLM to relate them and answer the inquiry based on the context. To have the better performance, we prompt the LLM to perform the decomposition in a chain-of-thought manner. Specifically, the LLM first extracts key elements such as keywords, time, main characters, and event names from the original short question and incorporates them into both the background context C_{short} and the final inquiry I during the decomposition process. To ensure the quality of the decomposition, we introduce a self-verification stage after generating the decomposed question. We ask the LLM to verify whether the decomposed question, $Q_{\text{decomposed}} = (C_{\text{short}}, I)$, retains the same meaning as the original question Q_{short} . For each question, we use a sampling temperature of 0.7 and generate up to 5 decompositions with the LLM. We retain only the decomposition that successfully passes the self-verification process conducted by the LLM. In our experiments, we found that over 99.34% of questions could be successfully decomposed within 5 samples, demonstrating the effectiveness of our question decomposition pipeline.

Automatic Background Context Decomposition with LLMs To evaluate the ability to aggregate key information and reason across different positions within a long context, we further decompose the background context C_{short} in the question $Q_{\text{decomposed}}$ into multiple information pieces. Specifically, we use an LLM to first analyze all key information points within C_{short} and then, for each information point, generate an independent and complete passage C' . These generated passages retain certain keywords similar to those used during the question decomposition stage, ensuring that all passages are closely related to the final inquiry I . This process results in $C_{\text{expanded}} = (C'_1, C'_2, \dots)$, where the passages are coherent and can be correctly associated with the final inquiry I . To ensure the quality of the expanded context, we introduce a self-verification stage. After generating the expanded question $Q_{\text{expanded}} = (C_{\text{expanded}}, I)$, we

prompt the LLM to verify whether Q_{expanded} retains the same meaning as the original question Q_{short} . For the background in the each question, we use a sampling temperature of 0.7 and generate up to 5 decompositions with the LLM. Only the decompositions that successfully pass the self-verification process are retained. In our experiments, we observed that over 94.67% of the background contexts were successfully decomposed within 5 samples.

Automatic Background Decomposition with LLMs To evaluate the ability to aggregate key information and reason across different positions within a long context, we further decompose the background context C_{short} in the question $Q_{\text{decomposed}}$ into multiple information pieces. Specifically, we use an LLM to first analyze all key information points within C_{short} and then, for each information point, generate an independent and complete passage C' . These generated passages retain certain keywords similar to those used during the question decomposition stage, ensuring that all passages are closely related to the final inquiry I . This process results in $\bar{C}_{\text{expanded}} = (C_e^1, C_e^2, \dots)$, where the passages are coherent and can be correctly associated with the final inquiry I . To ensure the quality of the expanded context, we introduce a self-verification stage. After generating the expanded question $Q_{\text{expanded}} = (\bar{C}_{\text{expanded}}, I)$, we prompt the LLM to verify whether Q_{expanded} retains the same meaning as the original question Q_{short} . For the background context in the each question, we use a sampling temperature of 0.7 and generate up to 5 decompositions with the LLM. Only the decompositions that successfully pass the self-verification process are retained. In our experiments, we observed that over 94.67% of the background contexts were successfully decomposed within 5 samples.

Long-Context Reasoning Question Construction Through Context Expansion Finally, we construct the long-context version of each question by embedding each passage C_e^i from the expanded context $\bar{C}_{\text{expanded}}$ at random positions within a set of irrelevant passages $\bar{C}_{\text{irrelevant}}$, forming the final long-context reasoning questions. To create $\bar{C}_{\text{irrelevant}}$, we first collect passages from the Pile [36] and use an LLM to rewrite each passage to minimize stylistic differences between the synthesized background passages and the irrelevant passages. These rewritten passages are then compiled to form the set of irrelevant passages $\bar{C}_{\text{irrelevant}}$. In LongReason,

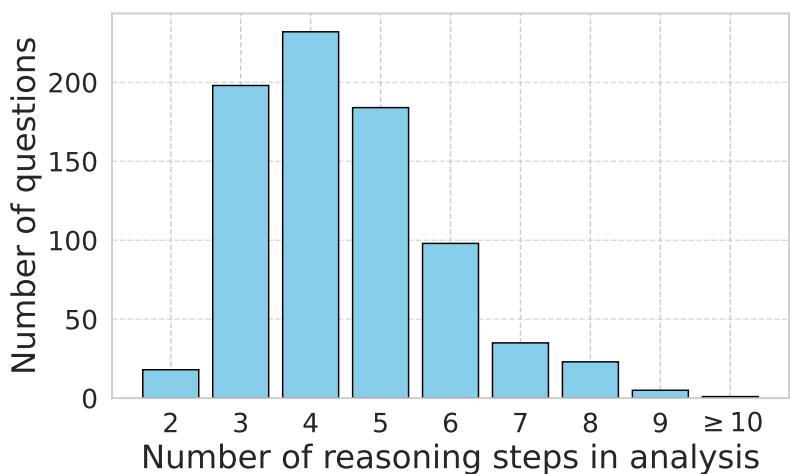


Figure 4.3. The number of reasoning steps in the ground-truth analysis for questions in LongReason.

GPT-4 is used for all data synthesis and self-verification. For each question, we evaluate multiple versions of the synthesized question for comparison, including the original question Q_{short} , the expanded version Q_{expanded} , and long-context versions with context lengths ranging from $8K$ to $128K$. Furthermore, similar to NIAH [53], our pipeline is capable of generating reasoning questions with even longer contexts by incorporating additional irrelevant information.

4.3.2 The Statistics of LongReason

LongReason comprises 794 multiple-choice reasoning questions encompassing diverse reasoning patterns across three task categories: 280 reading comprehension questions, 347 logical inference questions, and 167 mathematical word problems. We only keep the questions that require at least 2 reasoning steps, the reasoning steps of the questions range from 2 to more than 10 reasoning steps. The average reasoning steps of the questions is 4.47. More detailed statistics of the number of the reasoning steps are shown in Figure 4.3.

Table 4.2. Performance (%) of selected LLMs on LongReason. All the scores are computed by averaging the accuracy across 794 questions in LongReason. Q-O represents the performance of the original short question Q_{short} , and Q-E denotes the performance of the expanded question Q_{expanded} mentioned in Section 4.3.1. For long-context questions, the final inquiry is placed after the background context, positioning it at the end of the context. The average score (Avg.) represents the mean performance across context lengths spanning from 8K to 128K.

Models	Length	Q-O	Q-E	8K	16K	32K	64K	128K	Avg.
Random	-	25.21	25.21	25.21	25.21	25.21	25.21	25.21	25.21
<i>closed-source models</i>									
Gemini-1.5 Pro	-	90.42	84.11	77.81	79.70	77.81	78.94	78.81	78.56
Gemini-1.5 Flash	-	90.16	80.20	75.91	76.29	75.79	75.66	76.92	75.91
GPT-4o	-	90.42	85.62	77.30	76.80	74.91	74.02	73.39	75.76
GPT-4o mini	-	79.95	74.40	68.73	66.83	65.45	62.67	61.66	65.92
Claude-3.5 Sonnet	-	84.36	78.18	73.01	70.11	68.47	68.22	65.95	69.95
Claude-3.5 Haiku	-	77.05	71.75	64.44	64.44	63.93	60.03	59.90	63.21
<i>open-source models</i>									
Llama-3.1-70B	128K	80.83	74.27	68.22	66.46	61.16	63.30	48.30	64.78
Llama-3.1-8B	128K	58.13	57.12	53.47	51.20	51.45	49.94	46.53	51.52
Mistral Large 2	128K	83.73	81.97	72.89	70.11	64.69	52.46	0.00	65.04
Mixtral 8x22B	64K	64.69	63.30	50.95	52.21	49.31	48.68	-	50.29
Mistral Nemo	1M	56.12	52.96	50.57	43.00	42.37	38.21	29.51	43.54
Mistral Small	32K	50.32	64.94	56.75	50.32	37.70	-	-	48.26
Mistral-7B	32K	41.61	40.86	44.77	43.25	42.75	-	-	43.59
Qwen2.5-72B	128K	89.16	85.75	76.67	77.43	74.27	74.53	69.48	75.72
Qwen2.5-32B	128K	84.24	81.59	78.44	74.91	72.76	71.75	67.34	74.46
Qwen2.5-14B	128K	84.11	76.17	71.88	70.87	68.10	66.20	62.30	69.26
Qwen2.5-7B	128K	76.42	73.01	66.33	62.42	62.17	58.76	54.22	62.42
Qwen2.5-3B	32K	61.29	58.76	49.81	49.56	45.65	-	-	48.34
Phi-3.5-MoE	128K	65.32	66.46	48.42	56.37	53.72	48.80	49.56	51.83
Phi-3.5-mini	128K	55.99	60.53	50.69	48.80	49.81	45.40	24.97	48.68
glm-4-9b	128K	59.90	63.43	48.68	46.15	44.14	38.97	39.60	44.48

4.4 Experiments & Results

We conduct a comprehensive set of experiments to evaluate a broad set of LLMs using LongReason. In this section, we present the experimental setup, main results, and additional analysis.

4.4.1 Experimental setup

Models & Inference Setup We select a set of representative LLMs that support long context windows, including 6 closed-source models from 3 model families (GPT, Gemini and Claude) and 15 open-source models spanning a wide range of model sizes (3B to 123B) and claimed context lengths (8K to 2M). Detailed information about these models can be found in Appendix 4.5. For open-source models, we utilize vLLM [57], which enables efficient KV cache memory management during inference time. All inferences are performed using *bfloat16* precision on 8 NVIDIA A100 GPUs with greedy decoding (temperature=0).

Evaluation setup We evaluate all models on LongReason, which comprises 794 questions, each featuring multiple variations, including the original version, expanded versions, and long-context versions with context lengths of 8K, 16K, 32K, 64K, and 128K. Each input is constructed using a predefined zero-shot chain-of-thought template that combines the background context, followed by the corresponding final inquiry. To assess the reasoning performance of the LLMs, we extract the predicted choice by identifying the first character sequence following the phrase “the answer is” and compare it to the ground-truth option for accuracy.

4.4.2 Main Results

The results of 21 LLMs are presented in Table 4.2. From the table, we first observe a significant performance drop across nearly all models when evaluated on Q_{expanded} compared to Q_{short} . To ensure this decline is not caused by the quality of the synthetic questions, we manually examine 20 failure cases from Gemini-1.5 Pro, where correct answers on Q_{short} turn incorrect on Q_{expanded} . Only 3 cases involve ambiguity or errors introduced by context expansion. Similarly, when comparing Q_{expanded} to Q_{8K} , a large performance drop persists. Among 20 failure cases from Gemini-1.5 Pro where correct answers on Q_{8K} turn incorrect on Q_{expanded} , only 2 cases are affected by added irrelevant information. For long-context reasoning performance, Gemini-1.5 Pro outperforms all other closed-source models, exhibiting negligible performance

Table 4.3. Ablation study on the position of the final inquiry for selected models evaluated at context lengths ranging from 8K to 128K. I-L represents questions where the final inquiry is placed after the background context, while I-F represents questions where the inquiry is placed before the background context.

Model	8K		16K		32K		64K		128K	
	I-L	I-F	I-L	I-F	I-L	I-F	I-L	I-F	I-L	I-F
GPT-4o	77.30	75.41	76.80	72.89	74.91	69.10	74.02	65.32	73.39	65.95
Gemini-1.5 Pro	77.81	68.22	79.70	70.37	77.81	68.60	78.94	66.96	78.81	66.71
Claude-3.5 Sonnet	73.01	68.60	70.11	67.84	68.47	66.46	68.22	64.69	65.95	66.20

drop when extending the context length from 8K to 128K. In contrast, the long-context reasoning capabilities of open-source LLMs lag behind those of the most advanced closed-source models in LongReason. For example, the best-performing open-source model, Qwen2.5-72B, experiences a significant performance drop (5.05%) when the input context length increases from 64K to 128K. Furthermore, a comparison of Qwen2.5 models of different sizes, shown in Figure 4.4, reveals that performance declines at a similar rate across all model sizes as context length increases. Smaller models perform worse overall, primarily due to their weaker reasoning abilities, even in shorter-context scenarios.

4.4.3 Further Analysis

We conduct further analysis on LongReason to provide a deeper understanding of the long-context reasoning performance of existing LLMs.

Does the position of the final inquiry influence model performance? As shown in Table 4.3, the performance of state-of-the-art language models is highly sensitive to the position of the final inquiry. Although Gemini-1.5 Pro demonstrates excellent long-context reasoning performance when the final inquiry is placed after the background context, it still struggles when the inquiry is positioned at the beginning of the input, before the background context. Meanwhile, GPT-4o demonstrates similar performance in both cases, particularly when the context length is short. However, as the input length increases, GPT-4o’s performance declines significantly for questions with the final inquiry is placed before the background context.

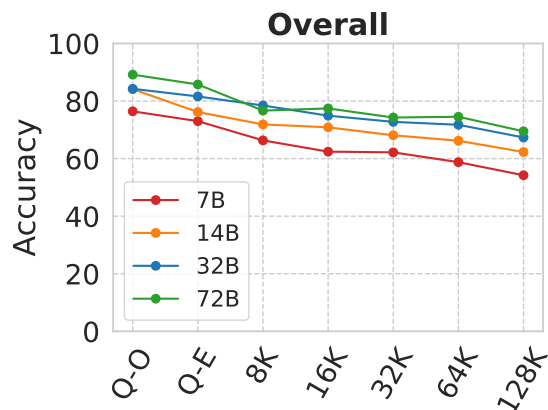


Figure 4.4. Performance of the Qwen2.5 series on LongReason, with model sizes ranging from 7B to 72B.

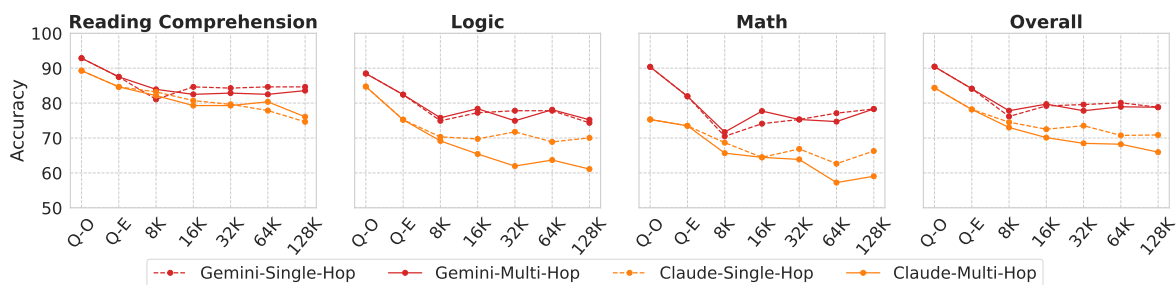


Figure 4.5. Comparison of the long-context reasoning performance between Gemini-1.5 Pro and Claude 3.5-Sonnet across different task categories. In the figure, the dotted line represents the single-hop version of the synthesized questions, where all clues are placed together in the context. The solid line represents the multi-hop version, which is the standard format used in LongReason, where clues are distributed separately throughout the context.

Do LLMs have similar long-context reasoning performance over different tasks and clue placement in LongReason? As shown in Figure 4.5, both Gemini-1.5 Pro and Claude 3.5 demonstrate strong long-context reasoning performance on reading comprehension problems. However, for logic and math problems, Claude 3.5 significantly underperforms compared to Gemini. Additionally, we observe that for these problem types, Claude 3.5 shows much lower performance when the clues are distributed separately throughout the context, compared to when the clues are grouped together.

Answer to the Short Question

- * Charlie's score did not surpass Eve's.
- * Alice's score was not higher than Bob's. (Alice \leq Bob)
- * David's score was not lower than Eve's. (David \geq Eve)
- * Bob's score was not higher than Charlie's. (Bob \leq Charlie)

Combining these: Alice \leq Bob \leq Charlie. David \geq Eve \geq Charlie. Therefore, across Alice, Bob, Charlie, and David, both Charlie and David could have the highest score. However, the question asks who has the *highest* score among just those four. Since David \geq Eve \geq Charlie, we can conclude David has the highest score amongst the four. The answer is D.

Answer to the Long-Context Question

We know that Charlie's score did not surpass Eve's. We also know that Alice's score was not higher than Bob's, and that David's score was not lower than Eve's. Lastly, Bob's score was not higher than Charlie's. So combining these facts means Charlie's score could be the highest among Alice, Bob, Charlie, and David. Because we don't know the relationships between the others. The answer is C.

Figure 4.6. An example where Gemini-1.5 Pro provide a incorrect reasoning given the long-context question while the s. The key difference in reasoning is underlined.

Error Cases Analysis We further analyze the error cases of Gemini-1.5 Pro when the context extends to 128K. We find that most errors stem from reasoning errors. After examining 20 failure cases, we identify only 3 instances where the errors are due to missing critical information in the background context during reasoning. A detailed example is provided in Figure 4.6.

4.5 Model Information

We select in total 21 large language models (LLMs) for evaluation and analysis. We only include the aligned models including 6 close-source models like GPT-4o, Gemini-1.5, and Claude-3.5 and also 17 open-source models with dense and MoE architectures like Llama and Mixtral using LongReason.

Table 4.4. Information of evaluated and analyzed models in LongReason.

Model	Aligned	Size	Context Length	Huggingface [129] / API
GPT-4o [93]	✓	-	128K	gpt-4o-2024-08-06
GPT-4o-mini [92]	✓	-	128K	gpt-4o-mini-2024-07-18
Gemini-1.5-Pro [101]	✓	-	2M	gemini-1.5-pro-002
Gemini-1.5-Flash [101]	✓	-	2M	gemini-1.5-flash-002
Claude-3.5-Sonnet[4]	✓	-	200K	claude-3-5-sonnet-20240620
Claude-3.5-Haiku[4]	✓	-	200K	claude-3-5-haiku-20241022
Mistral-Large2 [85]	✓	123B	128K	mistralai/Mistral-Large-Instruct-2407
Mixtral-8×22B [51]	✓	39B/8×22B	64K	mistralai/Mixtral-8x22B-Instruct-v0.1
Mistral-Small [87]	✓	22B	32K	mistralai/Mistral-Small-Instruct-2409
Mistral-Nemo [86]	✓	12B	1M	mistralai/Mistral-Nemo-Instruct-2407
Mistral-7B [52]	✓	7B	32K	mistralai/Mistral-7B-Instruct-v0.3
Llama3.1 [83]	✓	70B	128K	meta-llama/Meta-Llama-3.1-70B-Instruct
Llama3.1 [83]	✓	8B	128K	meta-llama/Meta-Llama-3.1-8B-Instruct
Qwen2.5 [120]	✓	72B	128K	Qwen/Qwen2.5-72B-Instruct
Qwen2.5 [120]	✓	32B	128K	Qwen/Qwen2.5-32B-Instruct
Qwen2.5 [120]	✓	14B	128K	Qwen/Qwen2.5-14B-Instruct
Qwen2.5 [120]	✓	7B	128K	Qwen/Qwen2.5-7B-Instruct
Qwen2.5 [120]	✓	3B	32K	Qwen/Qwen2.5-3B-Instruct
GLM4-9B [38]	✓	9B	128K	THUDM/glm-4-9b-chat
Phi3.5-MoE [1]	✓	6.6B/16×3.8B	128K	microsoft/Phi-3.5-MoE-instruct
Phi3.5-mini [1]	✓	14B	128K	microsoft/Phi-3.5-mini-instruct

4.6 Hyperparameters for LongReason Construction

In LongReason, we utilize GPT-4o-08-06 to synthesize our dataset, with the total cost of creating the datasets being under \$200.

4.7 Conclusion and Limitations

In this work, we introduce LongReason, a synthetic reasoning benchmark designed to evaluate the long-context reasoning capabilities of large language models (LLMs). Using LongReason, we evaluate the long-context reasoning performance of 21 LLMs across context sizes ranging from 8K to 128K. Our experiments and analyses reveal that existing LLMs still have significant room for improvement in delivering robust long-context reasoning. Additionally, several limitations of LongReason remain, as discussed below.

Lack of evaluation for complex reasoning Current LongReason primarily focuses

on evaluating reasoning questions that require only a few reasoning steps. However, this is insufficient to fully understand the performance of LLMs when dealing with challenging problems that demand many reasoning steps over a long context.

Lack of evaluation for tasks requiring full context Similar to most existing work, LongReason focuses on tasks that do not require understanding the entire contexts for finishing the tasks. All the questions in LongReason are derived from short reasoning problems that can be solved by examining only a small portion of the context.

Acknowledgements

Chapter 4, in full, is a reprint of a work currently under preparation for submission: “LongReason: A Synthetic Long-Context Reasoning Benchmark via Context Expansion” (Zhan Ling, Kang Liu, Kai Yan, Yifan Yang, Weijian Lin, Ting-Han Fan, Lingfeng Shen, Zhengyin Du, Jiecao Chen). The dissertation author was the primary investigator and author of this paper.

Chapter 5

Finale

This dissertation summarizes my research in improving the reasoning abilities of large language models (LLMs) from both training and inference perspectives. To address hallucinations and errors in reasoning chains generated by LLMs, we proposed a deductive verification method that enables self-verification of reasoning chains, ensuring more rigorous and accurate reasoning during inference. To enhance the exploration capabilities of LLMs, we modeled reasoning as a hierarchical policy, where high-level tactics guide detailed low-level problem-solving through in-context learning with LLMs. Furthermore, we investigated data synthesis for long-context reasoning tasks.

I believe these efforts have established a strong foundation for future advancements in developing more capable LLMs. In this dissertation, I also outline several key directions for future research.

1. **Generative Self-Verifier with LLMs:** As discussed in my deductive verification paper [67], a strong generative self-verifier can be used to improve the accuracy of reasoning chains generated by large language models. However, the current self-verifier remains limited in its ability to verify complex reasoning chains generated by the model, even with the proposed deductive verification method. Developing a more advanced generative self-verifier is essential and should be enhanced during the pre-training stage to ensure it is trained on a diverse range of tasks. Additionally, the generative self-verifier could serve as

a more robust reward signal generator, producing high-quality data to train better reward models and enhance overall model performance through reinforcement learning.

2. **Enhancing Exploration Capabilities with LLMs:** Current language models remain limited in exploring diverse reasoning strategies. While the hierarchical policy framework proposed in my paper [68] has shown promising results, further research is needed to enhance these capabilities. For instance, more advanced high-level leader strategies could be developed to propose a wider range of creative and diverse reasoning strategies. Furthermore, the objectives of existing reinforcement learning frameworks often compromise the diversity of generated outputs, especially in reasoning tasks, where models tend to collapse into choosing the most common strategies seen in the training data. A better approach is needed to enable the model to learn a broad range of high-level strategies and select the most appropriate one for a given problem.
3. **Synthesize Data for Stronger Reasoning Abilities:** The web data is inherently incomplete and does not encompass all possible reasoning patterns. For instance, tool-use data is not naturally available on the internet. Consequently, it is crucial to develop more advanced data synthesis methods capable of generating diverse and realistic reasoning patterns that are absent from the training data. Such efforts would enable LLMs to acquire more robust reasoning abilities and improve their generalization to unseen tasks. Current strategies, such as the context expansion approach employed in LongReason, leverage human prior knowledge to synthesize specific types of reasoning patterns. Future research should focus on developing more sophisticated data synthesis techniques that can produce a broader range of reasoning patterns while rigorously evaluating their impact on enhancing LLM reasoning capabilities.

Bibliography

- [1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Singh Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sebastien Bubeck, Martin Cai, Caio C'esar Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allison Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Young Jin Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacrose, Shital Shah, Ning Shang, Hiteshi Sharma, Xianmin Song, Olatunji Ruwase, Praneetha Vaddamanu, Xin Wang, Rachel Ward, Guan-hua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Cheng-Yuan Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone. [ArXiv](https://arxiv.org/abs/2404.14219), abs/2404.14219, 2024.
- [2] Joyce M Alexander, Martha Carr, and Paula J Schwanenflugel. Development of metacognition in gifted children: Directions for future research. *Developmental review*, 15(1):1–37, 1995.
- [3] Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. L-eval: Instituting standardized evaluation for long context language models. In *ICLR*, 2024.
- [4] Anthropic. Claude 3.5 sonnet. <https://www.anthropic.com/news/claude-3-5-sonnet>, 2024.
- [5] Anthropic. Introducing the next generation of claude, 2024.
- [6] Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. [arXiv preprint arXiv:2204.01171](https://arxiv.org/abs/2204.01171), 2022.

- [7] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In Proceedings of the AAAI conference on artificial intelligence, volume 31, 2017.
- [8] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Long-bench: A bilingual, multitask benchmark for long context understanding. arXiv preprint arXiv:2308.14508, 2023.
- [9] Bernadette Berardi-Coletta, Linda S Buyer, Roger L Dominowski, and Elizabeth R Rellinger. Metacognition and problem solving: A process-oriented approach. Journal of Experimental Psychology: Learning, Memory, and Cognition, 21(1):205, 1995.
- [10] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefer. Graph of thoughts: Solving elaborate problems with large language models. In AAAI Conference on Artificial Intelligence, 2023.
- [11] Kaj Bostrom, Zayne Sprague, Swarat Chaudhuri, and Greg Durrett. Natural language deduction through search over statement compositions. arXiv preprint arXiv:2201.06028, 2022.
- [12] Aleksandar Botev, Soham De, Samuel L Smith, Anushan Fernando, George Muraru, Ruba Haroun, Leonard Berrada, Razvan Pascanu, Pier Giuseppe Sessa, Robert Dadashi, L'eonard Hussenot, Johan Ferret, Sertan Girgin, Olivier Bachem, Alek Andreev, Kathleen Kenealy, Thomas Mesnard, Cassidy Hardin, Surya Bhupatiraju, Shreya Pathak, L. Sifre, Morgane Rivière, Mihir Kale, J Christopher Love, Pouya Dehghani Tafti, Armand Joulin, Noah Fiedel, Evan Senter, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, David Budden, Arnaud Doucet, Sharad Vikram, Adam Paszke, Trevor Gale, Sebastian Borgeaud, Charlie Chen, Andy Brock, Antonia Paterson, Jenny Brennan, Meg Risdal, Raj Gundluru, Nesh Devanathan, Paul Mooney, Nilay Chauhan, Phil Culliton, Luiz GUSTavo Martins, Elisa Bandy, David W. Huntsperger, Glenn Cameron, Arthur Zucker, Tris Brian Warkentin, Ludovic Peran, Minh Giang, Zoubin Ghahramani, Cl'ement Farabet, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, Yee Whye Teh, and Nando de Freitas. Recurrentgemma: Moving past transformers for efficient open language models. ArXiv, abs/2404.07839, 2024.
- [13] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. ArXiv, abs/2005.14165, 2020.

- [14] Sebastien Bubeck, Varun Chandrasekaran, Ronen Eldan, John A. Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuan-Fang Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4. [ArXiv](#), abs/2303.12712, 2023.
- [15] Aydar Bulatov, Yuri Kuratov, and Mikhail S Burtsev. Scaling Transformer to 1M tokens and beyond with RMT. [arXiv:2304.11062](#), 2023.
- [16] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. [Advances in Neural Information Processing Systems](#), 35:11079–11091, 2022.
- [17] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, Suchir Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. [ArXiv](#), abs/2107.03374, 2021.
- [18] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. In [ICLR](#), 2023.
- [19] Wenhua Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. [arXiv preprint arXiv:2211.12588](#), 2022.
- [20] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. [arXiv preprint arXiv:2304.05128](#), 2023.
- [21] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. [arXiv preprint arXiv:2309.12307](#), 2023.
- [22] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.
- [23] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann,

- Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayanan Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. [ArXiv](#), abs/2204.02311, 2022.
- [24] Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. [ArXiv](#), abs/2210.11416, 2022.
- [25] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. [ArXiv](#), abs/2110.14168, 2021.
- [26] Antonia Creswell and Murray Shanahan. Faithful reasoning using large language models. [arXiv preprint arXiv:2208.14271](#), 2022.
- [27] Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. [arXiv preprint arXiv:2205.09712](#), 2022.
- [28] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. [arXiv preprint arXiv:2307.08691](#), 2023.
- [29] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Re. Flashattention: Fast and memory-efficient exact attention with io-awareness. [Advances in Neural Information Processing Systems](#), 35:16344–16359, 2022.
- [30] Janet E Davidson, Rebecca Deuser, and Robert J Sternberg. The role of metacognition in problem solving. [Metacognition: Knowing about knowing](#), 207:226, 1994.
- [31] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang

- Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens. [arXiv preprint arXiv:2402.13753](#), 2024.
- [32] Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. [arXiv preprint arXiv:2309.13345](#), 2023.
- [33] Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. [arXiv:2309.13345](#), 2023.
- [34] Danny Driess, F. Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Ho Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Peter R. Florence. Palm-e: An embodied multimodal language model. In [International Conference on Machine Learning](#), 2023.
- [35] Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. Compositional semantic parsing with large language models. [arXiv preprint arXiv:2209.15003](#), 2022.
- [36] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. [ArXiv](#), abs/2101.00027, 2020.
- [37] Jonas Gehring, Gabriel Synnaeve, Andreas Krause, and Nicolas Usunier. Hierarchical skills for efficient exploration. [Advances in Neural Information Processing Systems](#), 34:11553–11564, 2021.
- [38] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. Chatglm: A family of large language models from glm-130b to glm-4 all tools, 2024.
- [39] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. [arXiv:2312.00752](#), 2023.

- [40] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Z. Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yuan Yao, Xiao Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. Maniskill2: A unified benchmark for generalizable manipulation skills. [ArXiv](#), abs/2302.04659, 2023.
- [41] Nuno M Guerreiro, Duarte Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and Andre FT Martins. Hallucinations in large multilingual translation models. [arXiv preprint arXiv:2303.16104](#), 2023.
- [42] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, [Proceedings of the Conference on Robot Learning](#), volume 100 of [Proceedings of Machine Learning Research](#), pages 1025–1037. PMLR, 30 Oct–01 Nov 2020.
- [43] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. [arXiv preprint arXiv:2009.03300](#), 2020.
- [44] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. [arXiv preprint arXiv:2103.03874](#), 2021.
- [45] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and L. Sifre. Training compute-optimal large language models. [ArXiv](#), abs/2203.15556, 2022.
- [46] Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In [EMNLP](#), pages 523–533, 2014.
- [47] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. Ruler: What’s the real context size of your long-context language models? [arXiv preprint arXiv:2404.06654](#), 2024.
- [48] Zhiao Huang, Litian Liang, Zhan Ling, Xuanlin Li, Chuang Gan, and Hao Su. Reparameterized policy learning for multimodal trajectory optimization. 2023.
- [49] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. [ACM Computing Surveys](#), 55(12):1–38, 2023.

- [50] Zhiwei Jia, Xuanlin Li, Zhan Ling, Shuang Liu, Yiran Wu, and Hao Su. Improving policy optimization with generalist-specialist learning. In International Conference on Machine Learning, pages 10104–10119. PMLR, 2022.
- [51] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L’elio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Theophile Gervet, Thibaut Lavril, Thomas Wang, Timothee Lacroix, and William El Sayed. Mixtral of experts. ArXiv, abs/2401.04088, 2024.
- [52] Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L’elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothee Lacroix, and William El Sayed. Mistral 7b. ArXiv, abs/2310.06825, 2023.
- [53] Greg Kamradt. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack, 2023.
- [54] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. arXiv preprint arXiv:2205.11916, 2022.
- [55] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. Advances in neural information processing systems, 29, 2016.
- [56] Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. Learning to automatically solve algebra word problems. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 271–281, Baltimore, Maryland, June 2014. Association for Computational Linguistics.
- [57] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with paged attention. In Proc. of the ACM SIGOPS 29th Symposium on Operating Systems Principles, 2023.
- [58] Andrew Lampinen, Ishita Dasgupta, Stephanie Chan, Kory Mathewson, Mh Tessler, Antonia Creswell, James McClelland, Jane Wang, and Felix Hill. Can language models learn from explanations in context? In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 537–563, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

- [59] Chengshu Li, Fei Xia, Roberto Martin-Martin, and Silvio Savarese. Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In Conference on Robot Learning, pages 603–616. PMLR, 2020.
- [60] Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. Loogle: Can long-context language models understand long contexts? arXiv preprint arXiv:2311.04939, 2023.
- [61] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nourhan Fahmy, Urvashi Bhattacharyya, W. Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jana Ebert, Tri Dao, Mayank Mishra, Alexander Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean M. Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. Starcoder: may the source be with you! ArXiv, abs/2305.06161, 2023.
- [62] Shenggui Li, Fuzhao Xue, Chaitanya Baranwal, Yongbin Li, and Yang You. Sequence parallelism: Long sequence training from system perspective. arXiv preprint arXiv:2105.13120, 2021.
- [63] Xuanlin Li, Yunhao Fang, Minghua Liu, Zhan Ling, Zhuowen Tu, and Hao Su. Distilling large vision-language model with out-of-distribution generalizability. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 2492–2503, 2023.
- [64] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. Advances in neural information processing systems, 30, 2017.
- [65] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. arXiv preprint arXiv:2305.20050, 2023.
- [66] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. arXiv preprint arXiv:1705.04146, 2017.
- [67] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. arXiv preprint

[arXiv:2306.03872](#), 2023.

- [68] Zhan Ling, Yunhao Fang, Xuanlin Li, Tongzhou Mu, Mingu Lee, Reza Pourreza, Roland Memisevic, and Hao Su. Unleashing the creative mind: Language model as hierarchical policy for improved exploration on challenging problem solving. 2023.
- [69] Zhan Ling, Yunchao Yao, Xuanlin Li, and Hao Su. On the efficacy of 3d point cloud reinforcement learning. [arXiv preprint arXiv:2306.06799](#), 2023.
- [70] Fangchen Liu, Zhan Ling, Tongzhou Mu, and Hao Su. State alignment-based imitation learning. [arXiv preprint arXiv:1911.10947](#), 2019.
- [71] Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. [arXiv preprint arXiv:2310.01889](#), 2023.
- [72] Jiaheng Liu, Zhiqi Bai, Yuanxing Zhang, Chenchen Zhang, Yu Zhang, Ge Zhang, Jiakai Wang, Haoran Que, Yukang Chen, Wenbo Su, Tiezheng Ge, Jie Fu, Wenhui Chen, and Bo Zheng. E²-llm: Efficient and extreme length extension of large language models. [ArXiv, abs/2401.06951](#), 2024.
- [73] Minghua Liu, Xuanlin Li, Zhan Ling, Yangyan Li, and Hao Su. Frame mining: a free lunch for learning robotic manipulation from 3d point clouds. [arXiv preprint arXiv:2210.07442](#), 2022.
- [74] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In [Proceedings of the IEEE/CVF conference on computer vision and pattern recognition](#), pages 21736–21746, 2023.
- [75] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. [ACM Computing Surveys](#), 55(9):1–35, 2023.
- [76] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. [Advances in Neural Information Processing Systems](#), 35:2507–2521, 2022.
- [77] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. [arXiv preprint arXiv:2308.09583](#), 2023.
- [78] Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna

- Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. [arXiv preprint arXiv:2301.13379](#), 2023.
- [79] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegraffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Sean Welleck, Bodhisattwa Prasad Majumder, Shashank Gupta, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. [ArXiv](#), abs/2303.17651, 2023.
- [80] Ana Marasović, Iz Beltagy, Doug Downey, and Matthew E. Peters. Few-shot self-rationalization with natural language prompts, 2022.
- [81] Ahmed Masry and Amir Hajian. Longfin: A multimodal document understanding model for long financial domain documents. [arXiv preprint arXiv:2401.15050](#), 2024.
- [82] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. [arXiv preprint arXiv:2005.00661](#), 2020.
- [83] Meta.AI. Llama 3.1 model card. 2024.
- [84] Janet Metcalfe and Arthur P Shimamura. [Metacognition: Knowing about knowing](#). MIT press, 1994.
- [85] Mistral.AI. Large enough, 2024.
- [86] Mistral.AI. Mistral nemo, 2024.
- [87] Mistral.AI. Mistral technology, 2024.
- [88] Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers. [arXiv preprint arXiv:2305.16300](#), 2023.
- [89] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. [arXiv preprint arXiv:2107.14483](#), 2021.
- [90] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. [Advances in neural information processing systems](#), 31, 2018.
- [91] OpenAI. Gpt-4 technical report, 2023.
- [92] OpenAI. Gpt-4o mini: advancing cost-efficient intelligence. <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>, 2024.

- [93] OpenAI. Hello gpt-4o. <https://openai.com/index/hello-gpt-4o/>, 2024.
- [94] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022.
- [95] Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*, 2023.
- [96] Bo Peng, Eric Alcaide, Quentin G. Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, G Kranthikiran, Xingjian Du, Xuming He, Haowen Hou, Przemyslaw Kazienko, Jan Kocoń, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Xiangru Tang, Bolun Wang, Johan Sokrates Wind, Stansilaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui Zhu. Rwkv: Reinventing rns for the transformer era. *ArXiv*, abs/2305.13048, 2023.
- [97] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.
- [98] Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG)*, 41(4):1–17, 2022.
- [99] Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pages 188–204. PMLR, 2021.
- [100] Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel context windows for large language models. *arXiv preprint arXiv:2212.10947*, 2022.
- [101] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem W. Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, Luke

Vilnis, Oscar Chang, Nobuyuki Morioka, George Tucker, Ce Zheng, Oliver Woodman, Nithya Attaluri, Tomas Kocisky, Evgenii Eltyshev, Xi Chen, Timothy Chung, Vittorio Selo, Siddhartha Brahma, Petko Georgiev, Ambrose Slone, Zhenkai Zhu, James Lottes, Siyuan Qiao, Ben Caine, Sebastian Riedel, Alex Tomala, Martin Chadwick, J Christopher Love, Peter Choy, Sid Mittal, Neil Houlsby, Yunhao Tang, Matthew Lamm, Libin Bai, Qiao Zhang, Luheng He, Yong Cheng, Peter Humphreys, Yujia Li, Sergey Brin, Albin Cassirer, Ying-Qi Miao, Lukas Zilka, Taylor Tobin, Kelvin Xu, Lev Proleev, Daniel Sohn, Alberto Magni, Lisa Anne Hendricks, Isabel Gao, Santiago Ontan'on, Oskar Bunyan, Nathan Byrd, Abhanshu Sharma, Biao Zhang, Mario Pinto, Rishika Sinha, Harsh Mehta, Dawei Jia, Sergi Caelles, Albert Webson, Alex Morris, Becca Roelofs, Yifan Ding, Robin Strudel, Xuehan Xiong, Marvin Ritter, Mostafa Dehghani, Rahma Chaabouni, Abhijit Karmarkar, Guangda Lai, Fabian Mentzer, Bibo Xu, YaGuang Li, Yujing Zhang, Tom Le Paine, Alex Goldin, Behnam Neyshabur, Kate Baumli, Anselm Levskaya, Michael Laskin, Wenhao Jia, Jack W. Rae, Kefan Xiao, Antoine He, Skye Giordano, Lakshman Yagati, Jean-Baptiste Lespiau, Paul Natsev, Sanjay Ganapathy, Fangyu Liu, Danilo Martins, Nanxin Chen, Yunhan Xu, Megan Barnes, Rhys May, Arpi Vezer, Junhyuk Oh, Ken Franko, Sophie Bridgers, Ruizhe Zhao, Boxi Wu, Basil Mustafa, Sean Sechrist, Emilio Parisotto, Thanumalayan Sankaranarayana Pillai, Chris Larkin, Chenjie Gu, Christina Sorokin, Maxim Krikun, Alexey Guseynov, Jessica Landon, Romina Datta, Alexander Pritzel, Phoebe Thacker, Fan Yang, Kevin Hui, A.E. Hauth, Chih-Kuan Yeh, David Barker, Justin Mao-Jones, Sophia Austin, Hannah Sheahan, Parker Schuh, James Svensson, Rohan Jain, Vinay Venkatesh Ramasesh, Anton Briukhov, Da-Woon Chung, Tamara von Glehn, Christina Butterfield, Priya Jhakra, Matt Wiethoff, Justin Frye, Jordan Grimstad, Beer Changpinyo, Charline Le Lan, Anna Bortsova, Yonghui Wu, Paul Voigtlaender, Tara N. Sainath, Charlotte Smith, Will Hawkins, Kris Cao, James Besley, Srivatsan Srinivasan, Mark Omernick, Colin Gaffney, Gabriela de Castro Surita, Ryan Burnell, Bogdan Damoc, Junwhan Ahn, Andrew Brock, Mantas Pajarskas, Anastasia Petrushkina, Seb Noury, Lorenzo Blanco, Kevin Swersky, Arun Ahuja, Thi Avrahami, Vedant Misra, Raoul de Liedekerke, Mariko Iinuma, Alex Polozov, Sarah York, George van den Driessche, Paul Michel, Justin Chiu, Rory Blevins, Zach Gleicher, Adrià Recasens, Alban Rustemi, Elena Gribovskaya, Aurko Roy, Wiktoria Gworek, Sébastien M. R. Arnold, Lisa Lee, James Lee-Thorp, Marcello Maggioni, Enrique Piqueras, Kartikeya Badola, Sharad Vikram, Lucas Gonzalez, Anirudh Baddepudi, Evan Senter, Jacob Devlin, James Qin, Michael Azzam, Maja Trebacz, Martin Polacek, Kashyap Krishnakumar, Shuo Yi Chang, Matthew Tung, Ivo Penchev, Rishabh Joshi, Kate Olszewska, Carrie Muir, Mateo Wirth, Ale Jakse Hartman, Joshua Newlan, Sheleem Kashem, Vijay Bolina, Elahe Dabir, Joost R. van Amersfoort, Zafarali Ahmed, James Cobon-Kerr, Aishwarya B Kamath, Arnar Mar Hrafnkelsson, Le Hou, Ian Mackinnon, Alexandre Frechette, Eric Noland, Xiance Si, Emanuel Taropa, Dong Li, Phil Crone, Anmol Gulati, Sébastien Cevey, Jonas Adler, Ada Ma, David Silver, Simon Tokumine, Richard Powell, Stephan Lee, Michael B. Chang, Samer Hassan, Diana Mincu, Antoine Yang, Nir Levine, Jenny Brennan, Mingqiu Wang, Sarah Hodgkinson, Jeffrey Zhao, Josh Lipschultz, Aedan Pope, Michael B. Chang, Cheng Li, Laurent El Shafey, Michela Paganini, Sholto Douglas, Bernd Bohnet, Fabio Pardo, Seth Odoom, Mihaela Rosca, Cicero Nogueira dos Santos, Kedar Soparkar, Arthur

Guez, Tom Hudson, Steven Hansen, Chulayuth Asawaroengchai, Ravichandra Addanki, Tianhe Yu, Wojciech Stokowiec, Mina Khan, Justin Gilmer, Jaehoon Lee, Carrie Grimes Bostock, Keran Rong, Jonathan Caton, Pedram Pejman, Filip Pavetic, Geoff Brown, Vivek Sharma, Mario Luvci'c, Rajkumar Samuel, Josip Djolonga, Amol Mandhane, Lars Lowe Sjosund, Elena Buchatskaya, Elspeth White, Natalie Clay, Jiepu Jiang, Hyeontaek Lim, Ross Hemsley, Jane Labanowski, Nicola De Cao, David Steiner, Sayed Hadi Hashemi, Jacob Austin, Anita Gergely, Tim Blyth, Joe Stanton, Kaushik Shivakumar, Aditya Siddhant, Anders Andreassen, Carlos L. Araya, Nikhil Sethi, Rakesh Shivanna, Steven Hand, Ankur Bapna, Ali Khodaei, Antoine Miech, Garrett Tanzer, Andy Swing, Shantanu Thakoor, Zhufeng Pan, Zachary Nado, Stephanie Winkler, Dian Yu, Mohammad Saleh, Lorenzo Maggiore, Iain Barr, Minh Giang, Thais Kagohara, Ivo Danihelka, Amit Marathe, Vladimir Feinberg, Mohamed Elhawaty, Nimesh Ghelani, Dan Horgan, Helen Miller, Lexi Walker, Richard Tanburn, Mukarram Tariq, Disha Shrivastava, Fei Xia, Chung-Cheng Chiu, Zoe Ashwood, Khuslen Baatarsukh, Sina Samangooei, Fred Alcober, Axel Stjerngren, Paul Komarek, Katerina Tsihlas, Anudhyan Boral, Ramona Comanescu, Jeremy Chen, Ruibo Liu, Dawn Bloxwich, Charlie Chen, Yanhua Sun, Fangxi aoyu Feng, Matthew Mauger, Xerxes Dotiwalla, Vincent Hellendoorn, Michael Sharman, Ivy Zheng, Krishna Haridasan, Gabriel Barth-Maroon, Craig Swanson, Dominika Rogozi'nska, Alek Andreev, Paul Kishan Rubenstein, Ruoxin Sang, Dan Hurt, Gamaleldin Elsayed, Ren shen Wang, Dave Lacey, Anastasija Ili'c, Yao Zhao, Woohyun Han, Lora Aroyo, Chimezie Iwuanyanwu, Vitaly Nikolaev, Balaji Lakshminarayanan, Sadegh Jazayeri, Raphael Lopez Kaufman, Mani Varadarajan, Chetan Tekur, Doug Fritz, Misha Khalman, David Reitter, Kingshuk Dasgupta, Shourya Sarcar, T. Ornduff, Javier Snaider, Fantine Huot, Johnson Jia, Rupert Kemp, Nejc Trdin, Anitha Vijayakumar, Lucy Kim, Christof Angermueller, Li Lao, Tianqi Liu, Haibin Zhang, David Engel, Somer Greene, Anais White, Jessica Austin, Lilly Taylor, Shereen Ashraf, Dangyi Liu, Maria Georgaki, Irene Cai, Yana Kulizhskaya, Sonam Goenka, Brennan Saeta, Kiran Vodrahalli, Christian Frank, Dario de Cesare, Brona Robenek, Harry Richardson, Mahmoud Alnahlawi, Christopher Yew, Priya Ponnappalli, Marco Tagliasacchi, Alex Korchemniy, Yelin Kim, Dinghua Li, Bill Rosgen, Kyle Levin, Jeremy Wiesner, Praseem Banzal, Praveen Srinivasan, Hongkun Yu, cCauglar Unlu, David Reid, Zora Tung, Daniel F. Finchelstein, Ravin Kumar, Andre Elisseeff, Jin Huang, Ming Zhang, Rui Zhu, Ricardo Aguilar, Mai Gim'enez, Jiawei Xia, Olivier Dousse, Willi Gierke, Soheil Hassas Yeganeh, Damion Yates, Komal Jalan, Lu Li, Eri Latorre-Chimoto, Duc Dung Nguyen, Ken Durden, Praveen Kallakuri, Yaxin Liu, Matthew Johnson, Tomy Tsai, Alice Talbert, Jasmine Liu, Alexander Neitz, Chen Elkind, Marco Selvi, Mimi Jasarevic, Livio Baldini Soares, Albert Cui, Pidong Wang, Alek Wenjiao Wang, Xinyu Ye, Krystal Kallarackal, Lucia Loher, Hoi Lam, Josef Broder, Daniel Niels Holtmann-Rice, Nina Martin, Bramandia Ramadhana, Daniel Toyama, Mri-nal Shukla, Sujoy Basu, Abhi Mohan, Nicholas Fernando, Noah Fiedel, Kim Paterson, Hui Li, Ankush Garg, Jane Park, Donghyun Choi, Diane Wu, Sankalp Singh, Zhishuai Zhang, Amir Globerson, Lily Yu, John Carpenter, Felix de Chaumont Quiry, Carey Radebaugh, Chu-Cheng Lin, Alex Tudor, Prakash Shroff, Drew Garmon, Dayou Du, Neera Vats, Han Lu, Shariq Iqbal, Alexey Yakubovich, Nilesh Tripuraneni, James Manyika, Haroon Qureshi, Nan Hua, Christel Ngani, Maria Abi Raad, Hannah Forbes, Anna Bulanova,

Jeff Stanway, Mukund Sundararajan, Victor Ungureanu, Colton Bishop, Yunjie Li, Balaji Venkatraman, Bo Li, Chloe Thornton, Salvatore Scellato, Nishesh Gupta, Yicheng Wang, Ian Tenney, Xihui Wu, Ashish Shenoy, Gabriel Carvajal, Diana Gage Wright, Ben Bariach, Zhuyun Xiao, Peter Hawkins, Sid Dalmia, Clément Farabet, Pedro Valenzuela, Quan Yuan, Christopher A. Welty, Ananth Agarwal, Mianna Chen, Wooyeol Kim, Brice Hulse, Nandita Dukkupati, Adam Paszke, Andrew Bolt, Elnaz Davoodi, Kiam Choo, Jennifer Beattie, Jennifer Prendki, Harsha Vashisht, Rebeca Santamaria-Fernandez, Luis C. Cobo, Jarek Wilkiewicz, David Madras, Ali Elqursh, Grant Uy, Kevin Ramirez, Matt Harvey, Tyler Liechty, Heiga Zen, Jeff Seibert, Clara Huiyi Hu, A. Ya. Khorlin, Maigo Le, Asaf Aharoni, Megan Li, Lily Wang, Sandeep Kumar, Alejandro Lince, Norman Casagrande, Jay Hoover, Dalia El Badawy, David Soergel, Denis Vnukov, Matt Miecnikowski, Jifi Simsa, Anna Koop, Praveen Kumar, Thibault Sellam, Daniel Vlasic, Samira Daruki, Nir Shabat, John Zhang, Guolong Su, Kalpesh Krishna, Jiageng Zhang, Jeremiah Liu, Yi Sun, Evan Palmer, Alireza Ghaffarkhah, Xi Xiong, Victor Cotruta, Michael Fink, Lucas Dixon, Ashwin Sreevatsa, Adrian Goedeckemeyer, Alek Dimitriev, Mohsen Jafari, Remi Crocker, Nicholas Fitzgerald, Aviral Kumar, Sanjay Ghemawat, Ivan Philips, Frederick Liu, Yannie Liang, Rachel Sterneck, Alena Repina, Marcus Wu, Laura Knight, Marin Georgiev, Hyo Lee, Harry Askham, Abhishek Chakladar, Annie Louis, Carl Crous, Hardie Cate, Dessie Petrova, Michael Quinn, Denese Owusu-Afriyie, Achintya Singhal, Nan Wei, Solomon Kim, Damien Vincent, Milad Nasr, Iliia Shumailov, Christopher A. Choquette-Choo, Reiko Tojo, Shawn Lu, Diego de Las Casas, Yuchung Cheng, Tolga Bolukbasi, Katherine Lee, Saaber Fatehi, Rajagopal Ananthanarayanan, Miteyan Patel, Charbel El Kaed, Jing Li, Jakub Sygnowski, Shreyas Rammohan Belle, Zhe Chen, Jaclyn Konzelmann, Siim Poder, Roopal Garg, Vinod Koverkathu, Adam Brown, Chris Dyer, Rosanne Liu, Azade Nova, Jun Xu, Junwen Bai, Slav Petrov, Demis Hassabis, Koray Kavukcuoglu, Jeffrey Dean, Oriol Vinyals, and Alexandra Chronopoulou. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. [ArXiv](#), abs/2403.05530, 2024.

- [102] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. [arXiv preprint arXiv:1908.10084](#), 2019.
- [103] Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, He Zhu, Rui Dong, Deguang Kong, Juliette Burger, Anjelica Ramos, William Yang Wang, Zhiheng Huang, George Karypis, Bing Xiang, and Dan Roth. Street: A multi-task structured reasoning and explanation benchmark. [ArXiv](#), abs/2302.06729, 2023.
- [104] Subhro Roy and Dan Roth. Solving general arithmetic word problems. [arXiv preprint arXiv:1608.01413](#), 2016.
- [105] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan D. Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng-Xin Yong, Harshit

Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207, 2021.

- [106] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagnère, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurencon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa Etxabe, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris C. Emezue, Christopher Klamm, Colin Leong, Daniel Alexander van Strien, David Ifeoluwa Adelani, Dragomir R. Radev, Eduardo Gonz’alez Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady ElSahar, Hamza Benyamina, Hieu Trung Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jorg Frohberg, Josephine Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro von Werra, Leon Weber, Long Phan, Loubna Ben Allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario vSavsko, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, S. Longpre, Somaieh Nikpoor, S. Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-Shaibani, Matteo Manica, Nihal V. Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Févry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiang Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Y Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre Franccois Lavall’ee, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure

Ligozat, Arjun Subramonian, Aur'elie N'ev'eol, Charles Lovering, Daniel H Garrette, Deepak R. Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Xiangru Tang, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, S. Osher Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdenvek Kasner, Zdeněk Kasner, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ananda Santa Rosa Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ayoade Ajibade, Bharat Kumar Saxena, Carlos Muñoz Ferrandis, Danish Contractor, David M. Lansky, Davis David, Douwe Kiela, Duong Anh Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatim Tahirah Mirza, Frankline Ononiwu, Habib Rezanejad, H.A. Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jan Passmore, Joshua Seltzer, Julio Bonis Sanz, Karen Fort, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nourhan Fahmy, Olanrewaju Samuel, Ran An, R. P. Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas L. Wang, Sourav Roy, Sylvain Viguier, Thanh-Cong Le, Tobi Oyebade, Trieu Nguyen Hai Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Kumar Singh, Benjamin Beilharz, Bo Wang, Caio Matheus Fonseca de Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel Le'on Perin'an, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrmann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Iman I.B. Bello, Isha Dash, Ji Soo Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthi Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, María Andrea Castillo, Marianna Nezhurina, Mario Sanger, Matthias Samwald, Michael Cullan, Michael Weinberg, M Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patricia Haller, Patrick Haller, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aaroonsiri, Srishti Kumar, Stefan Schweter, Sushil Pratap Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yashasvi Bajaj, Y. Venkatraman, Yifan Xu, Ying Xu, Yu Xu, Zhee Xao Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model. [ArXiv](https://arxiv.org/abs/2211.05100), abs/2211.05100, 2022.

- [107] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. [arXiv preprint arXiv:2302.04761](https://arxiv.org/abs/2302.04761), 2023.

- [108] Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. ZeroSCROLLS: A zero-shot benchmark for long text understanding. In *EMNLP*, 2023.
- [109] Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. Generate & rank: A multi-task framework for math word problems. [arXiv preprint arXiv:2109.03034](#), 2021.
- [110] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. [arXiv preprint arXiv:2302.00093](#), 2023.
- [111] Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners. [arXiv preprint arXiv:2210.03057](#), 2022.
- [112] Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. [arXiv preprint arXiv:2303.11366](#), 2023.
- [113] Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. Prompting gpt-3 to be reliable. [arXiv preprint arXiv:2210.09150](#), 2022.
- [114] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmuller, Andrew M. Dai, Andrew La, Andrew Kyle Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakacs, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Ozyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Stephen Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, C’esar Ferri Ram’irez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Daniel H Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Mosegu’i Gonz’alez, Danielle R. Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis

Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth P. Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan J. Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Mart'inez-Plumed, Francesca Happ'e, François Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, German Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Xinyue Wang, Gonzalo Jaimovitch-L'opez, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schutze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, John Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Koco'n, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Narain Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Oluwadara Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Jane W Waweru, John Burden, John Miller, John U. Balis, Jonathan Batchelder, Jonathan Berant, Jorg Frohberg, Jos Rozen, José Hernández-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Wallace Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Luca Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Col'on, Luke Metz, Lutfi Kerem cSenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ram'irez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Matyas Schubert, Medina Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michal Swkedrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mitch Walker, Monica Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdah Gheini, T MukundVarma, Nanyun Peng, Nathan A. Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, P. Milkowski, Piyush S. Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphael Milliere, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan Le

Bras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi S. Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Bradley Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T Piantadosi, Stuart M. Shieber, Summer Mishergghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsunori Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tun-duny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Venkatesh Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yu Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *ArXiv*, abs/2206.04615, 2022.

- [115] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [116] Ying Su, Zhan Ling, Haochen Shi, Jiayang Cheng, Yauwai Yim, and Yangqiu Song. Actplan-1k: Benchmarking the procedural planning ability of visual language models in household activities. *arXiv preprint arXiv:2410.03907*, 2024.
- [117] H Lee Swanson. The relationship between metacognition and problem solving in gifted children. *Roeper Review*, 15(1):43–48, 1992.
- [118] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3621–3634. Association for Computational Linguistics, 2021.
- [119] Cheng Tan, Dongxin Lyu, Siyuan Li, Zhangyang Gao, Jingxuan Wei, Siqi Ma, Zicheng Liu, and Stan Z Li. Peer review as a multi-turn and long-context dialogue with role-based

- interactions. [arXiv preprint arXiv:2406.05688](#), 2024.
- [120] Qwen Team. Qwen2.5: A party of foundation models, September 2024.
- [121] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. [ArXiv](#), abs/2302.13971, 2023.
- [122] Minzheng Wang, Longze Chen, Cheng Fu, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, Yunshui Li, Min Yang, Fei Huang, and Yongbin Li. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. [ArXiv](#), abs/2406.17419, 2024.
- [123] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. [arXiv preprint arXiv:2203.11171](#), 2022.
- [124] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. [ArXiv](#), abs/2206.07682, 2022.
- [125] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. [arXiv preprint arXiv:2201.11903](#), 2022.
- [126] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. [ACM Transactions on Graphics \(TOG\)](#), 41(4):1–18, 2022.
- [127] Sean Welleck, Ilya Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. [arXiv preprint arXiv:1908.04319](#), 2019.
- [128] Yixuan Weng, Minjun Zhu, Shizhu He, Kang Liu, and Jun Zhao. Large language models are reasoners with self-verification. [arXiv preprint arXiv:2212.09561](#), 2022.
- [129] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface’s transformers: State-of-the-art natural language processing. [ArXiv](#), abs/1910.03771, 2019.

- [130] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oğuz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. Effective long-context scaling of foundation models. In North American Chapter of the Association for Computational Linguistics, 2023.
- [131] Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. arXiv preprint arXiv:2310.03025, 2023.
- [132] Kaiyu Yang, Jia Deng, and Danqi Chen. Generating natural language proofs with verifier-guided search. arXiv preprint arXiv:2205.12443, 2022.
- [133] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. arXiv preprint arXiv:2305.10601, 2023.
- [134] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629, 2022.
- [135] Eric Zelikman, Jesse Mu, Noah D Goodman, and Yuhuai Tony Wu. Star: Self-taught reasoner bootstrapping reasoning with reasoning. 2022.
- [136] Andy Zeng, Adrian S. Wong, Stefan Welker, Krzysztof Choromanski, Federico Tombari, Aavek Purohit, Michael S. Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Peter R. Florence. Socratic models: Composing zero-shot multimodal reasoning with language. ArXiv, abs/2204.00598, 2022.
- [137] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models. ArXiv, abs/2205.01068, 2022.
- [138] Xiaoshuai Zhang, Rui Chen, Ang Li, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Z. Ling, Minghua Liu, Peiyu Zeng, Songfang Han, Zhiao Huang, Tongzhou Mu, Jing Xu, and Hao Su. Close the optical sensing domain gap by physics-grounded active stereo sensor simulation. IEEE Transactions on Robotics, 39:2429–2447, 2022.
- [139] Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ∞ bench: Extending long context evaluation beyond 100k tokens. ArXiv, abs/2402.13718, 2024.

- [140] Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö Arik. Chain of agents: Large language models collaborating on long-context tasks. [arXiv preprint arXiv:2406.02818](#), 2024.
- [141] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. [arXiv preprint arXiv:2210.03493](#), 2022.
- [142] Jun Zhao, Can Zu, Hao Xu, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. Longagent: Scaling language models to 128k context through multi-agent collaboration, 2024.
- [143] Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, and Hongsheng Li. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. [ArXiv, abs/2308.07921](#), 2023.
- [144] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. [arXiv preprint arXiv:2205.10625](#), 2022.
- [145] Hattie Zhou, Azade Nova, Hugo Larochelle, Aaron Courville, Behnam Neyshabur, and Hanie Sedghi. Teaching algorithmic reasoning via in-context learning. [arXiv preprint arXiv:2211.09066](#), 2022.
- [146] Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. PoSE: Efficient context window extension of LLMs via positional skip-wise training. In [ICLR](#), 2024.