Are Large Language Models good at Reasoning?

Parthiv A Dholaria
2021078

Aayush Ranjan
2021003

Arnav Agarwal
2021235

Pulkit Nargotra
2021273

Harshvardhan Singh
2021052

Utsav Garg
2021108

Abstract

This survey provides an in-depth analysis of the logical reasoning capabilities of Large Language Models (LLMs), which are critical for tasks involving deductive reasoning, problem-solving, and decision-making. Despite the progress made by models such as GPT-4, ChatGPT, and Vicuna/CodeLlama, complex logical reasoning remains a significant challenge. Current methods often augment LLMs with symbolic solvers, requiring translation from natural language (NL) to symbolic language (SL), but these approaches suffer from parsing errors that hinder reasoning accuracy.

We review several emerging techniques aimed at improving LLM reasoning. These include self-correction mechanisms, where models iteratively refine their answers, and reinforcement learning (RL) strategies using Monte Carlo Tree Search (MCTS) to guide multi-step reasoning and explore potential reasoning paths. We have also examined multi-agent frameworks where multiple LLM agents interact with each other to arrive at an answer to a logical reasoning problem. We also examine multiple prompt techniques such as Least-to-Most prompting and Tree of Thought for The survey evaluates their performance across benchmarks such as Logic Bench , ProofWriter and PrOntoQA. We also evaluate metrics specifically designed for evaluating performance of LLMs on logical reasoning tasks.

CCS Concepts

• **Computing methodologies** → **Reasoning about belief and knowledge**; • **Information systems** → **Language models**; • **Social and professional topics** → *Systems analysis and design*; Model curricula.

Keywords

Large Language Models, Logical Reasoning, Natural Language Inference, Deductive Reasoning, Symbolic Solvers, Self-correction Mechanisms, Multi-step Reasoning

1 Introduction

Logical reasoning is a core component of human cognition, fundamental to our ability to solve problems, make decisions, and derive conclusions based on structured information. In the realm of artificial intelligence, specifically with Large Language Models (LLMs), the capability to emulate logical reasoning is pivotal for advancing their utility in complex applications, such as natural language inference (NLI), decision-making, and scientific problem-solving. Logical reasoning in LLMs encompasses a wide variety of tasks, including logical inference, deductive reasoning, inductive reasoning, and mathematical reasoning. Each of these types demands a structured, context-aware approach to language processing that goes beyond surface-level linguistic patterns.

1.1 Logical Reasoning Types in LLMs

Natural Language Inference (NLI): NLI is one of the most prominent logical reasoning tasks, requiring models to evaluate the logical relationship between two sentences, typically determining whether one statement entails, contradicts, or is neutral with respect to another. While LLMs have shown significant progress in NLI, challenges arise when dealing with nuanced or highly contextualized logical relationships, where human-like reasoning is crucial.

1.2 Deductive Reasoning

Deductive reasoning involves arriving at a specific conclusion based on general premises or rules. LLMs must apply structured, rule-based reasoning to determine whether a conclusion is logically sound, given a set of premises. In tasks such as theorem proving or structured legal reasoning, deductive logic plays a critical role, yet LLMs still face difficulties in maintaining logical consistency across multi-step deductions.

1.3 Mathematical Reasoning

Logical reasoning also intersects with mathematics, where LLMs are expected to handle symbolic reasoning, arithmetic operations, and more complex algebraic problems. Recent advances have enabled LLMs to process mathematical queries to some extent, but they still frequently fail at multi-step mathematical reasoning or problems that require precise logical structuring.

1.4 Spatial and Temporal Reasoning

Reasoning about the spatial and temporal relationships between objects or events presents unique challenges for LLMs. These tasks require models to track and infer dynamic relationships over time or across spatial contexts, which often involves abstract thinking not inherently present in the linguistic data these models are trained on.

1.5 Survey Focus

This survey aims to provide a comprehensive overview of the state of logical reasoning in LLMs, analyzing key tasks such as logical inference, NLI, and deductive reasoning, while exploring advanced techniques such as self-correction, symbolic solvers, reinforcement learning, and prompting methods. We review benchmarks such as ProofWriter and PrOntoQA, which assess LLMs' deductive reasoning performance, and evaluate the strengths and limitations of these models in tackling complex logical tasks. Our goal is to highlight existing gaps in logical reasoning and propose future directions for research to further enhance the reasoning capabilities of LLMs.

1.6 Contributions

We provide a comprehensive review of logical reasoning types in LLMs, covering deductive reasoning, natural language inference (NLI), inductive reasoning, spatial reasoning, and mathematical reasoning. We survey state-of-the-art techniques to improve LLM reasoning, such as self-correction, reinforcement learning (RL) with Monte Carlo Tree Search (MCTS), and advanced prompting methods like Chain-of-Thought and Least-to-Most prompting. We benchmark the performance of LLMs on key datasets like ProofWriter and PrOntoQA, offering insights into strengths, weaknesses, and potential improvements in logical reasoning tasks. We identify gaps and future directions, suggesting hybrid neural-symbolic models and better fine-tuning techniques to improve LLMs' performance on complex reasoning tasks.

2 Literature Review

2.1 Benchmarks

2.1.1 LogicBench: Systematic Evaluation of Logical Reasoning Capabilities. **LogicBench** [8] was designed to systematically evaluate the logical reasoning capabilities of Large Language Models (LLMs). It tests models across a variety of logical reasoning tasks, including:

- **Propositional logic**
- **First-order logic**
- **Non-monotonic logic**

The primary goal is to assess LLM performance on complex reasoning patterns such as negation, logical inference, and contextual understanding.

Dataset Structure.

- LogicBench consists of 25 distinct reasoning patterns, structured around context-question pairs.
- Each context contains logical premises, and models must derive conclusions based on those premises.
- The evaluation includes two primary tasks:
  - **Binary Question-Answering (BQA)**: Models determine whether a statement is logically entailed by the given context.
  - **Multiple-Choice Questions-Answering (MCQA)**: Models select the correct conclusion from several possible options based on the provided context.

Key Findings.

- LLMs struggle with **complex logical reasoning**, particularly in tasks involving negations and intricate logical rules.
- Even advanced models, such as **GPT-4 and ChatGPT**, underperform when tasked with nuanced logical contexts and detailed contextual information, highlighting limitations in their reasoning abilities.
- The results indicate a need for **further research and development** to improve LLMs' logical reasoning performance, particularly in symbolic reasoning and contextual understanding.

The findings from LogicBench reveal significant gaps in the logical reasoning capabilities of LLMs, offering insights for future model improvements.

2.1.2 Proof Writer: Evaluating Multi-Step Deductive Reasoning. ProofWriter [12] is designed to test the multi-step logical reasoning capabilities of Large Language Models (LLMs), particularly focusing on:

- **Deductive reasoning and theorem proving**
- Different levels of reasoning complexity: from single-step to 5-hop reasoning

Dataset Structure.

- Consists of natural language facts and rules, prompting models to infer conclusions or verify queries based on premises.
- Covers both **closed-world** (unknown info assumed false) and **open-world** (unknown info unresolved) settings.
- Transforms formal logic into natural language, testing reasoning chains of varying difficulty.

Key Findings.

- LLMs struggle with **multi-step reasoning**, especially in open-world scenarios.
- Performance declines significantly with increased task complexity, particularly in 3-hop and 5-hop tasks.
- Logical consistency across multiple reasoning steps remains a challenge, requiring improved fine-tuning and reasoning-specific strategies.

2.1.3 PrOntoQA: Evaluating Deductive Reasoning Over Fictional Concepts. PrOntoQA [11] is a synthetic dataset focused on evaluating deductive reasoning capabilities in LLMs, specifically testing models on:

- Logical reasoning over **fictional and abstract concepts**, avoiding biases from pre-existing knowledge.
- Logical inferences based on context under **closed-world assumptions**, where unstated information is considered false.

Dataset Structure.

- Presents natural language facts and rules about fictional entities, requiring models to infer conclusions as true, false, or unknown.
- Encourages models to rely on logical reasoning rather than memorized knowledge.

Key Findings.

- LLMs struggle with generalized deductive reasoning, particularly with fictional entities outside their pre-training.
- Models like GPT-4 perform inconsistently on multi-step reasoning tasks involving abstract concepts.
- Exposes gaps in LLMs' logical reasoning abilities and highlights the need for improved inference and contextual understanding.

2.2 Prompt Engineering

2.2.1 Chain-of-Thought (CoT) Prompting. CoT prompting encourages LLMs to break down complex problems into sequential steps, helping with logical reasoning and improving accuracy for simpler tasks.

Limitations: CoT often fails with complex reasoning tasks requiring multi-step deductions, as seen in "Tree of Thoughts" and "Easy

Problems That LLMs Get Wrong," leading to inconsistent outputs in unfamiliar scenarios.

2.2.2 Least-to-Most Prompting. This technique presented in the paper 'Least-to-Most Prompting Enables Complex Reasoning in Large Language Models' [16] enhances logical reasoning by breaking complex problems into smaller subproblems solved step-by-step, leading to improved logical consistency.
Efficacy: It outperforms CoT in multi-step reasoning tasks, maintaining a coherent thought process and showing better performance in structured problem-solving.

2.2.3 Tree of Thoughts (ToT) Prompting. ToT [17] allows LLMs to explore multiple reasoning paths, self-evaluate, and backtrack, mimicking human-like reasoning. It handles complex logical challenges effectively.
Efficacy: ToT excelled in tasks like the "Game of 24," outperforming CoT and handling intricate logical reasoning more effectively by structured exploration.

The "Easy Problems That LLMs Get Wrong" [14] paper showed that well-designed prompts significantly enhance LLM performance, with clarifying questions leading to a 40.7% improvement.

While CoT provides a basic framework, least-to-most and ToT prompting are more advanced, with ToT being the most effective for complex reasoning due to its ability to explore multiple thought paths and adapt.

2.3 Self-Correction in Large Language Models

We explore the use of self correction for logical reasoning as presented in the paper 'Large Language Models Cannot Self-Correct Reasoning Yet' [4].

Self-correction in the context of large language models (LLMs) refers to the process where a model revises its initial outputs to improve accuracy or rectify errors based on internal reasoning or external feedback. The paper distinguishes between two types of self-correction: intrinsic and extrinsic.

2.3.1 Intrinsic Self-Correction in LLMs. Intrinsic self-correction involves the model relying solely on its internal knowledge and reasoning abilities to correct its responses, without any external feedback or oracle labels.

- **LLMs Struggle with Self-Correction**: Models like GPT-3.5, GPT-4, and LLaMA-2 are largely ineffective at self-correcting reasoning tasks. In many cases, their performance stagnates or even degrades after self-correction attempts.
- **Performance Degradation**: Without external feedback, LLMs often convert correct answers into incorrect ones rather than improving on incorrect answers. For example, on the GSM8K dataset, GPT-3.5 changed 8.9% of correct answers to incorrect answers after the first round of self-correction, while only 7.6% of incorrect answers were corrected. This imbalance suggests a lack of reliable internal verification mechanisms.

2.3.2 Extrinsic Self-Correction in LLMs. Extrinsic self-correction involves the use of external feedback, such as oracle labels, to guide the correction process. Models tend to perform better when they receive explicit external feedback.

- **Improvement with Oracle Labels**: When oracle labels are available, models like GPT-4 show significant improvement. For example, on the CommonsenseQA dataset, GPT-4's accuracy increased from 82% to 85.5% with oracle feedback, while performance dropped to 79.5% without it after self-correction.

2.3.3 Prompt Design Impact on Self-Correction. The paper identifies that improvements in self-correction can often be attributed to suboptimal prompt design during initial response generation. By refining the initial prompt, better results may be achieved without the need for multiple rounds of self-correction.

2.3.4 Conclusion on Self-Correction. The research concludes that while self-correction is a promising approach, current LLMs lack the intrinsic ability to perform it effectively, especially in reasoning tasks. Key takeaways include:

- Models frequently exhibit performance degradation after self-correction attempts, turning correct answers into incorrect ones.
- External feedback plays a critical role in achieving meaningful improvements.
- Improving prompt design can lead to better initial results, reducing the need for self-correction altogether.

2.4 Multi Modality in Logical Reasoning for MLLMs

Recent advancements in Multimodal Large Language Models (MLLMs) have demonstrated various capabilities, from crafting poetry based on images to performing mathematical reasoning. However, there is still a lack of systematic evaluation of MLLMs' proficiency in logical reasoning tasks, which are critical for activities such as navigation and puzzle-solving.

Perception and Reasoning.

- Perception and reasoning are key components of high-level intelligence, working together in human problem-solving.
- While current MLLM datasets focus on perception tasks, such as retrieving facts from a scene, complex multimodal reasoning requires integrating both perception and logical reasoning for tasks like interpreting graphs and everyday problem-solving.

Current Limitations.

- Most MLLM benchmarks and datasets focus on traditional computer vision tasks like recognition, leaving a gap in evaluating reasoning abilities in multimodal LLM agents.
- Reasoning skills are essential for solving complex tasks and are a foundation for challenges that humans expect AI agents to tackle.

2.4.1 MathVista: Focused on Mathematical Multimodal Reasoning. **MathVista** [6] is a benchmark designed to assess multimodal logical reasoning in mathematical tasks:

- Covers a narrow range of reasoning skills, primarily involving numerical reasoning.

- Evaluates models' abilities to handle mathematical problem-solving in visual contexts, such as interpreting graphs or solving equations presented in images.

2.4.2 LogicVista: Comprehensive Multimodal Reasoning Benchmark. **LogicVista** [15] is designed to evaluate multimodal reasoning across a wider range of tasks compared to MathVista:

- Evaluates tasks such as deductive, numerical, and mechanical reasoning.
- Assesses models' performance on diagrammatic reasoning, which involves interpreting and reasoning about visual information presented in diagrams and patterns.

Key Findings.

- Evaluation of eight MLLMs on LogicVista revealed poor performance on reasoning tasks, with some models performing worse than random guessing.
- Models like GPT-4 Vision and LLaVA performed better, particularly in deductive, numerical, and mechanical reasoning tasks.
- Inductive and spatial reasoning posed the most challenges, indicating gaps in training datasets focusing on these reasoning types.
- Diagrammatic reasoning was a significant challenge for current models, with many MLLMs struggling to interpret and reason about visual information.
- Larger models, like GPT-4 Vision, showed a positive correlation between model size and performance, particularly on complex reasoning tasks. However, gaps remained in handling **diagram-based reasoning tasks.

2.5 Reinforcement Learning

2.5.1 Reasoning via Planning (RAP). The paper 'Reasoning with Language Model is Planning with World Mode' [13] makes use of a RL algorithm for logical reasoning tasks. RAP utilizes Monte Carlo Tree Search (MCTS) to enhance LLMs by combining them with a "world model." The LLM acts as both a reasoning agent and a simulator, exploring different paths and refining reasoning steps iteratively.

- **Performance:**
  - Achieved a 64% success rate in the Blocksworld task.
  - Outperformed GPT-4 with Chain-of-Thought (CoT) by 33% in plan generation.
  - Demonstrated better accuracy than CoT and other baselines in mathematical reasoning and logical inference tasks.
- **Limitations:**
  - Lacks a mathematically sound reward model.
  - Has high computational complexity and inference time.

2.5.2 Reinforcement Learning via Symbolic Feedback (RLSF). The paper 'Reinforcement Learning via Symbolic Feedback' [5] introduces a novel approach called RSLF. RLSF fine-tunes LLMs using detailed feedback from symbolic reasoning tools, providing token-level corrections instead of the usual scalar feedback from human evaluation.

- **Mathematical Reasoning:**
  - Uses a compiler to identify and correct errors in tasks like converting pseudo-code to C++.
  - Achieved a +52.64% improvement in compilation accuracy over traditional fine-tuning.
- **Logical Reasoning:**
  - Employs a Computer Algebra System (CAS) for tasks like the Game of 24 to check correctness at each step.
  - Achieved a 25% higher success rate compared to traditional methods.

RL approaches like RLSF and RAP significantly enhance LLMs' logical reasoning capabilities. They provide structured feedback and strategic exploration. Enable LLMs to learn from detailed corrections, adapt to complex tasks, and outperform conventional methods, resulting in more robust and accurate reasoning abilities.

2.6 Multi-Agent Systems for Enhancing Logical Reasoning in LLMs

Based on the papers *"Improving LLM Reasoning with Multi-Agent Tree-of-Thought Validator Agent"* and *"LLM Harmony: Multi-Agent Communication for Problem Solving"*, the following evaluation demonstrates how multi-agent systems improve the logical reasoning capabilities of Large Language Models (LLMs):

2.6.1 Tree-of-Thought Validator Agent. The paper 'Improving LLM Reasoning with Multi-Agent Tree-of-Thought Validator Agent' [3] introduces a multi-agent approach involving multiple Reasoner agents and a Thought Validator agent to enhance the accuracy of logical reasoning.

- **Diverse Reasoning Paths**: Reasoner agents explore different paths using a Tree-of-Thought (ToT) strategy, enabling more thorough exploration compared to a single agent.
- **Validation Mechanism**: The Thought Validator filters reasoning paths, ensuring only logical solutions are selected, thereby improving accuracy.
- **Iterative Refinement**: Feedback from the Thought Validator helps refine the reasoning process through iterative rounds.

This approach improved accuracy by 5.6% over the traditional ToT strategy.

2.6.2 LLM Harmony: Multi-Agent Communication for Problem Solving. The paper 'LLM Harmony: Multi-Agent Communication for Problem Solving' [10] presents a multi-agent framework where agents engage in role-playing to collaboratively solve problems.

- **Role-playing and Personas**: Agents take on different roles (e.g., student and teacher) to tackle problems collaboratively.
- **Problem Decomposition**: Agents guide each other by breaking down complex tasks into smaller, manageable steps.
- **Collaborative Validation**: Agents evaluate each other's solutions, enhancing overall accuracy.

This method resulted in a 15% increase in arithmetic accuracy and a 6% improvement in commonsense reasoning compared to single-agent models.

2.6.3 Overall Evaluation. Multi-agent systems enhance LLMs by providing:

- Diverse reasoning paths.
- Validation mechanisms to filter incorrect solutions.
- Iterative learning, allowing for more robust logical reasoning capabilities.

2.7 Symbolic Solvers for Logical Reasoning

2.7.1 Logic-LM: Empowering Large Language Models with Symbolic Solvers. LOGIC-LM [7] integrates LLMs with symbolic solvers to improve logical reasoning. It operates in three stages:

- **Problem Formulation**: LLMs translate natural language problems into symbolic logic.
- **Symbolic Reasoning**: A symbolic solver performs inference on the translated symbolic logic.
- **Result Interpretation**: The results from the symbolic solver are translated back into natural language by the LLM.

A key contribution of LOGIC-LM is the introduction of a self-refinement module. This module allows the LLM to refine its symbolic representations by utilizing error messages from symbolic solvers, leading to an iterative process that improves accuracy.

The integration of symbolic solvers with LLMs results in significant performance gains:

- 39.2% performance improvement over standard LLMs using simple prompting techniques.
- 18.4% improvement over Chain-of-Thought (CoT) prompting.

2.7.2 DiLA: Enhancing LLM Tool Learning with Differential Logic Layer. DiLA [18] proposes a novel method of enhancing LLM logical reasoning by incorporating a differential logic layer. This approach differs from LOGIC-LM by allowing LLMs to iteratively refine solutions using first-order logic constraints encoded into the network layers. DiLA is particularly effective in solving classical constraint satisfaction problems, including:

- Boolean satisfiability (SAT)
- Graph coloring

DiLA's embedded logical reasoning directly within the model's architecture improves both efficiency and correctness. In simple cases, it achieved 100% accuracy, outperforming existing solver-aided approaches, and demonstrated significant runtime improvements, especially for large-scale problems.

DiLA uses LLMs to translate natural language descriptions into symbolic representations (such as SAT problems). The differential logic layer then iteratively refines these solutions until they satisfy all logical constraints. It showcases its ability to handle challenging SAT instances with high clause-to-variable ratios, significantly reducing the time required to solve such problems, outperforming traditional solvers like Z3 and Kissat.

2.7.3 LoGiPT: Language Models can be Logical Solvers. The LoGiPT paper [1] addresses the limitations of Large Language Models (LLMs) in performing deductive reasoning tasks, particularly their reliance on external symbolic solvers. Traditional LLMs, when integrated with these solvers, often encounter parsing errors due to the complexity of translating natural language (NL) into symbolic language (SL), which ultimately impacts reasoning accuracy. The core objective of this work is to eliminate the dependency on external solvers and avoid errors arising from NL-to-SL translations by enabling LLMs to directly mimic symbolic solver processes.

Methodology: The authors propose a paradigm shift in LLM reasoning by directly modeling symbolic solver logic within the LLM framework. They formalized step-by-step reasoning processes typically handled by symbolic solvers and developed a custom instruction-tuning dataset that captures these reasoning steps in both NL and SL. The LLMs were fine-tuned on this dataset, allowing them to internalize solver reasoning structures and perform complex multi-step deductions autonomously, without relying on external systems.

Key Findings:

- *Performance Improvement:* LoGiPT significantly outperforms solver-augmented LLMs, including advanced models like GPT-4, improving deductive reasoning accuracy by up to 13%.
- *Symbolic Representations:* Models trained on symbolic data consistently achieved superior performance compared to those relying on natural language representations, demonstrating the effectiveness of SL for deductive reasoning.
- *Avoidance of Parsing Errors:* Fine-tuning LLMs with solver-derived data allowed them to avoid NL-to-SL parsing errors, leading to substantial improvements in reasoning consistency.

Conclusion: LoGiPT demonstrates that LLMs can be fine-tuned to mimic symbolic solver processes, resulting in improved accuracy and consistency without reliance on external solvers. The authors suggest that further research should explore hybrid models combining neural networks with explicit symbolic logic steps for even greater accuracy and reliability in logical inference tasks.

The problem translation ability of LLMs, combined with their strong natural language understanding, is fundamental in both LOGIC-LM, DiLA and LoGiPT. LLMs excel at converting complex natural language problems into structured symbolic logic, such as SAT or FOL, enabling them to tackle a wide range of reasoning tasks. This translation capability is essential for leveraging external tools in LOGIC-LM or enabling iterative refinement in DiLA.

2.8 Metrics

2.8.1 ROSCOE: Fine-Grained Evaluation of Step-by-Step Reasoning. **ROSCOE** [2] provides a detailed framework for evaluating the step-by-step reasoning process in LLMs, focusing on four key metrics:

- **Semantic Alignment**: Ensures each reasoning step is grounded in the context, maintaining logical consistency with the original information.
- **Logical Inference**: Detects contradictions and assesses whether conclusions logically follow from premises, ensuring internal consistency.
- **Semantic Similarity**: Measures the similarity between reasoning steps or reference answers, ensuring concise reasoning without redundancy or hallucinations.
- **Language Coherence**: Evaluates fluency and clarity, ensuring logical steps are articulated coherently.

2.8.2 RECEVAL: Evaluation of Reasoning Chains. **RECEVAL** [9] is designed to evaluate the correctness and informativeness of reasoning chains in LLMs, focusing on two key metrics:

- **Correctness**: Ensures each reasoning step logically follows from previous steps or input context, maintaining intra-step and inter-step consistency.
- **Informativeness**: Assesses how much new and useful information is added by each step toward the final conclusion, ensuring each step advances the reasoning process.

RECEVAL evaluates entire reasoning chains, detecting errors like hallucinations, redundancy, or logical inconsistencies. It provides a detailed, step-by-step analysis, offering deeper insights beyond just evaluating final answers.

3 Challenges Ahead

Overfitting. LLMs often produce verbose and incorrect responses by defaulting to solutions found in common online versions of puzzles, rather than accurately addressing modified benchmark questions. This indicates a tendency towards overfitting to web-based training data.

Lack of Logic or Common Sense. LLMs sometimes provide illogical responses or show commonsense inconsistencies, failing to maintain logical coherence in their reasoning.

Lack of Spatial Intelligence. Simple spatial reasoning tasks, such as understanding directions or movements, pose difficulties for LLMs, often resulting in incorrect spatial judgments.

Incorrect Mathematical Reasoning. LLMs frequently display poor understanding of basic mathematical principles, leading to arithmetic errors or incorrect counting.

Poor Linguistic Understanding. LLMs struggle with nuanced language tasks, reflecting gaps in their ability to understand linguistic subtleties or follow specific linguistic constraints.

Popular Science Misunderstandings. LLMs often misapply scientific knowledge, misunderstanding fundamental principles in physics, chemistry, or biology-related tasks.

Relational Misunderstandings. LLMs fail to correctly process relational contexts, such as hierarchical relationships or cause-and-effect dynamics, which leads to incorrect conclusions.

Illogical Chain of Thought. In the Chain of Thought (CoT) reasoning process, LLMs may exhibit logical inconsistencies or contradictions, demonstrating fragility in handling complex reasoning tasks.

4 Future Discussion

As Large Language Models (LLMs) continue to evolve, several key areas of improvement in logical reasoning should be prioritized:

- **Hybrid Systems**: Integrating symbolic reasoning with neural models to improve performance on complex multi-step reasoning tasks.
- **Enhanced Benchmarks**: Developing benchmarks that encompass a wider variety of tasks, including spatial reasoning, inductive reasoning, and diagrammatic problem-solving.
- **Reducing Overfitting**: Creating more diverse and challenging datasets to minimize reliance on web-based training data and enhance models' generalization abilities.
- **Common-Sense and Multimodal Reasoning**: Improving models' ability to handle common-sense reasoning, relational contexts, and multimodal reasoning involving visual or diagrammatic data.
- **Refining Techniques**: Further developing fine-tuning, reinforcement learning, and self-correction mechanisms to address logical consistency issues in complex reasoning.

These advancements are crucial for pushing LLMs' reasoning abilities and applying them effectively in real-world scenarios like scientific discovery, legal reasoning, and automated problem-solving.

5 Conclusion

In this work, we have explored the challenges and advancements in enhancing the logical reasoning capabilities of Large Language Models (LLMs). While LLMs have demonstrated impressive progress in various tasks, significant gaps remain in their ability to handle complex reasoning, particularly in areas like multi-step logic, common-sense reasoning, and multimodal problem-solving. The integration of symbolic reasoning, improved benchmarks, and refined training techniques are crucial for overcoming these limitations. As LLMs continue to evolve, addressing these issues will be essential for their successful application in real-world domains such as science, law, and automated decision-making systems.

References

[1] Jiazhan Feng, Ruochen Xu, Junheng Hao, Hiteshi Sharma, Yelong Shen, Dongyan Zhao, and Weizhu Chen. 2023. Language Models can be Logical Solvers. *arXiv preprint arXiv:2311.06158v1* (2023). https://arxiv.org/abs/2311.06158v1

[2] Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. ROSCOE: A Suite of Metrics for Scoring Step-by-Step Reasoning. *arXiv preprint arXiv:2212.07919v2* (2023). https://arxiv.org/abs/2212.07919v2

[3] Fatemeh Haji, Mazal Bethany, Maryam Tabar, Jason Chiang, Anthony Rios, and Peyman Najafirad. 2024. Improving LLM Reasoning with Multi-Agent Tree-of-Thought Validator Agent. *arXiv preprint arXiv:2409.11527* 2409, 11527v1 (2024). Retrieved September 17, 2024 from http://arxiv.org/abs/2409.11527

[4] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large Language Models Cannot Self-Correct Reasoning Yet. *arXiv preprint arXiv:2310.01798v2* (2024). https://arxiv.org/abs/2310.01798v2

[5] Piyush Jha, Parag Jana, Abhishek Arora, and Vijay Ganesh. 2024. Reinforcement Learning via Symbolic Feedback. *arXiv preprint arXiv:2405.16661* 2405, 16661v1 (2024). Retrieved May 29, 2024 from http://arxiv.org/abs/2405.16661

[6] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. MATHVISTA: Evaluating Mathematical Reasoning of Foundation Models in Visual Contexts. *arXiv preprint arXiv:2310.02255v3* (2024). https://arxiv.org/abs/2310.02255v3

[7] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. LOGIC-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. *arXiv preprint arXiv:2305.12295v2* (2023). https://arxiv.org/abs/2305.12295v2

[8] Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. LogicBench: Towards Systematic Evaluation of Logical Reasoning Ability of Large Language Models. *arXiv preprint arXiv:2404.15522v2* (2024). https://arxiv.org/abs/2404.15522v2

[9] Archiki Prasad, Swarnadeep Saha, Xiang Zhou, and Mohit Bansal. 2023. RECEVAL: Evaluating Reasoning Chains via Correctness and Informativeness. *arXiv preprint arXiv:2304.10703v2* (2023). https://arxiv.org/abs/2304.10703v2

[10] Sumedh Rasal. 2024. LLM Harmony: Multi-Agent Communication for Problem Solving. *arXiv preprint arXiv:2401.01312* 2401, 01312v1 (2024). Retrieved January 2, 2024 from http://arxiv.org/abs/2401.01312

[11] Abulhair Saparov and He He. 2023. Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought. In *International Conference on*

Learning Representations (ICLR). Retrieved October 4, 2023 from https://arxiv.org/abs/2210.01240

[12] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2021. ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language. *arXiv preprint arXiv:2012.13048* (2021). Retrieved September 17, 2024 from https://arxiv.org/abs/2012.13048

[13] Yingjie Wang, Minjun Zhu, Qingyi Tao, Xiang Zhuang, et al. 2024. Reasoning via Planning (RAP). *arXiv preprint arXiv:2405.16661* 2405, 16661v1 (2024). Retrieved May 29, 2024 from http://arxiv.org/abs/2405.16661

[14] Sean Williams and James Huckle. 2024. Easy Problems That LLMs Get Wrong. *arXiv preprint arXiv:2405.19616* 2405, 19616v2 (2024). Retrieved June 1, 2024 from http://arxiv.org/abs/2405.19616

[15] Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. 2024. LogicVista: Multimodal LLM Logical Reasoning Benchmark in Visual Contexts. *arXiv preprint arXiv:2407.04973v1* (2024). https://arxiv.org/abs/2407.04973v1

[16] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2022. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. *arXiv preprint arXiv:2205.10625* 2205, 10625v3 (2022). Retrieved August 1, 2023 from http://arxiv.org/abs/2205.10625

[17] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *arXiv preprint arXiv:2305.10625* 2305, 10625v3 (2023). Retrieved August 1, 2023 from http://arxiv.org/abs/2305.10625

[18] Yu Zhang, Hui-Ling Zhan, Zehua Pei, Yingzhao Lian, Lihao Yin, Mingxuan Yuan, and Bei Yu. 2024. DiLA: Enhancing LLM Tool Learning with Differential Logic Layer. *arXiv preprint arXiv:2402.11903v3* (2024). https://arxiv.org/abs/2402.11903v3