Ensembling Language Models to Navigate Complex Problem Solving in the Face of Conflicting Information

Dennis Perepech, Muyuan He, Aayushi Goenka, Chetan Sah, Sanah Sidhu
University of Southern California
{perepech, muyuanhe, goenkaa, csah, sanahsid}@usc.edu

Abstract

Large Language Models (LLMs) exhibit impressive reasoning capabilities, yet they often fall short of human standards for application specific use cases. Moreover, foundational LLMs are typically not available for fine tuning. As such, previous research has investigated various prompting techniques to elicit well-reasoned and correct responses from LLMs; however, these techniques become less effective as problems require further reasoning steps and introduce inconsistent or contradictory information in addition to rules on how to address such conflicts. We demonstrate how an ensemble LLM approach can improve performance on such problems by having one LLM first decompose a given problem into subproblems that require fewer reasoning steps and then having a second solver LLM iteratively solve the subproblems. The built up context by the solver LLM is then used to answer the original question. Additionally, we demonstrate how a smaller LLM fine tuned for the task of question decomposition can lead to performance in this task at or above a foundational LLM not fine tuned for question decomposition.

1 Introduction

The evolution of large language models (LLMs) has marked a significant milestone in the field of artificial intelligence, showcasing remarkable abilities in a wide range of text and NLP tasks (Devlin et al., 2019). Larger language models, like GPT-4, and the introduction of new prompting techniques, like chain-of-thought (CoT), have opened new avenues in solving complex reasoning problems (OpenAI et al., 2024). However, many advances in LLMs have arisen from increasing network sizes and tunable parameters. As a result, training and maintaining these LLMs become difficult in terms of computational efficiency and adaptability.

Prompting techniques like Chain-of-Thought prompting (Wei et al., 2023), Progressive-Hint prompting (Zheng et al., 2023), and Least-to-Most prompting (Zhou et al., 2023) have developed to improve the performance of certain tasks in response to the opaque nature of foundational LLMs, like ChatGPT and Gemini. However, accuracy on complex tasks remains low (Sahoo et al., 2024). Thus, in isolation these methods are often insufficient, especially for defensible reasoning tasks with contradictory information. Such problems mimic real world scenarios where information is inconsistent, such as when two sources of information contain different information on the same topic. One approach to resolve these conflicts is to establish a preference of information sources, so when information for two sources is contradictory, the information from the preferred source prevails. Therefore, solving such problems requires not only performing logical inference on atomic propositions but discerning which information should be valued. This kind of problem is further made more difficult by providing only some of the required knowledge as input and requiring any remaining information come from an internal understanding of the world.

Solving complex reasoning tasks can be considered in two parts – identifying subproblems and solving such subproblems to reach a final solution. Oftentimes, the same LLM is used for both tasks, but recently, work has demonstrated how these two capabilities could be separated into separate modules and delegated to different LMs, leading to a more flexible and efficient approach.

We investigate whether problem decomposition through the use of multiple LLMs can be applied specifically to natural language reasoning problems with conflicting information.

2 Related Work

The research area of problem decomposition focuses on modular problem-solving techniques that improve the adaptability and precision of LLMs.

Dua et al. (2022) demonstrated a prompting technique for improving LLM reasoning by decomposing a complex problem into subproblems and solving each subproblem by iteratively querying an

LLM. Their approach, Successive Prompting, enabled them to decouple LLMs' capabilities in identifying and solving subproblems, allowing them to judiciously inject data to help the model answer a particular question it could not previously answer. Ye et al. (2023) applied a similar approach for reasoning with tabular data. The authors decomposed tabular data and a given query into intermediate parts using a LLM. Then, the authors leveraged the decomposed context and query along with several in-context prompting examples with an LLM to arrive at a final answer to the query.

Building on the ideas of problem decomposition, Khot et al. (2023) proposed Decomposed Prompting, an approach to break down complex reasoning tasks into sub-tasks (via prompting) that could be delegated to a shared library of LLMs dedicated to those sub-tasks. The authors highlighted how such an approach can not only increase performance on multi-hop open-domain QA datasets but be more flexible and adaptable to different problems.

Extending these concepts further, Yoran et al. (2023) introduced Multi-Chain Reasoning (MCR), which prompts LLMs to meta reason over multiple chain-of-thought instances, rather than aggregating their answers, enabling high quality response on multi-hop QA datasets. Unlike past work which sampled LLM responses to arrive at a final solution, MCR served as a means to collect and reason on multiple pieces of evidence.

The most similar approach to ours is DaSLaM (Juneja et al., 2024). DaSLaM is a framework that separates the decomposition and solving stages of problem decomposition into two distinct modules. It leverages a 13B parameter LLaMa (Touvron et al., 2023) instruction tuned language model for problem decomposition and a larger language model for solution generation. This approach not only enhances the performance on complex reasoning tasks but also offers a system that can work with solvers of varying capacities. Additionally, the use of a smaller fine tuned LLM for question decomposition enables problem specific solution generation while utilizing and acknowledging the opaque nature of foundational LLMs. In contrast to our approach, DaSLaM was applied to mathematical problem solving datasets, like JEEBench (Arora et al., 2023), AQuA (Ling et al., 2017), and MATH (Hendrycks et al., 2021), as opposed to natural language reasoning problems.

3 Problem Description

We seek to investigate whether the technique of problem decomposition and iterative solving extends to natural language reasoning problems. More specifically, we investigate whether the DaSLaM model could improve the accuracy of linguistic reasoning problems with contradictory information. We compare our approach against zero shot and chain-of-thought approaches.

We utilize the BoardgameQA dataset, which consists of problems presented as fictitious board game scenarios (Kazemi et al., 2023). It aims to measure the defensible reasoning capacity of language models when faced with inconsistent or contradictory information. The BoargameQA dataset has multiple versions with varying levels of depth and contradictions. We have chosen to work with the main depth-2 test dataset from BoardgameQA.

An individual sample from the dataset, contains a set of facts, rules, preferences of rules, and a question. The task requires identifying if the premise of the question follows from the game state ("proved"), does not follow from the game state ("disproved"), or is not able to be determined given the information provided ("unknown").

A key point to note from the BoardgameQA research is that large-scale language models, even when fine-tuned on complex non-mathematical challenges, perform poorly on the BoardgameQA dataset (Kazemi et al., 2023). The authors demonstrated a monotonic decline in model performance as the reasoning depth increases, suggesting that models face greater challenges with tasks involving contradictory information.

4 Approach

Our approach is designed to augment the reasoning capabilities of LLMs by structurally decomposing complex reasoning tasks into manageable components. We investigated three different decomposers: gpt-3.5-turbo, gemini-1.0-pro (Team et al., 2024), and the fine tuned LLaMa 13B parameter model obtained from the authors of DaSLaM. We evaluated ChatGPT and Gemini as question solvers. We utilized the BoardgameQA dataset, which was obtained from the zip download in the original paper. Minimal preprocessing was applied to filter out information not relevant to our analysis, such as the logical proofs and natural language explanations.

We generated the base answers for Gemini and ChatGPT using the example field from the dataset

	Percent ’Proved’	Percent ’Disproved’	Percent ’Unknown’	Accuracy
Ground Truth	33.4	33.3	33.3	-
ChatGPT Zero Shot	39.2	47.4	13.4	45.0
ChatGPT CoT	11.5	38.8	49.7	37.7
ChatGPT+DaSLaM Decomposer	42.8	39.9	17.3	**53.0**
Gemini Zero Shot	35.7	44.3	20.0	45.2
Gemini CoT	33.1	44.8	22.1	42.9
Gemini+Gemini Decomposer	24.0	44.1	31.9	**50.3**
Gemini+ChatGPT Decomposer	15.9	30.7	53.4	44.3
Gemini+DaSLaM Decomposer	33.5	39.0	27.5	49.4

Table 1: Accuracy on BoardgameQA-main-depth-2 test set for several solver LMs and problem decomposers.

	‘Proved’ Label Statistics			‘Disproved’ Label Statistics			‘Unknown’ Label Statistics		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
ChatGPT Zero Shot	0.46	0.54	0.49	0.46	0.66	0.54	0.39	0.16	0.22
ChatGPT CoT	0.48	0.16	0.24	0.39	0.45	0.42	0.35	**0.52**	**0.41**
ChatGPT+DaSLaM Decomposer	0.51	**0.65**	**0.57**	**0.57**	**0.68**	**0.62**	**0.51**	0.26	0.35
Gemini Zero Shot	0.47	0.50	0.49	0.44	0.59	0.51	0.44	0.26	0.33
Gemini CoT	0.44	0.43	0.43	0.44	0.60	0.51	0.39	0.26	0.31
Gemini+Gemini Decomposer	**0.60**	0.43	0.50	0.49	**0.65**	**0.56**	0.44	0.42	0.43
Gemini+ChatGPT Decomposer	0.57	0.27	0.37	0.5	0.46	0.48	0.37	**0.60**	**0.46**
Gemini+DaSLaM Decomposer	0.51	**0.51**	**0.51**	**0.51**	0.60	0.55	**0.45**	0.37	0.4

Table 2: Precision, recall and F1 scores for tested approaches on BoardgameQA-main-depth-2 test set.

in a zeroshot fashion. We then utilized the base answers from ChatGPT and Gemini in addition to the original question to generate subquestions using the LLaMa 13B model for each respective solver as done by the original DaSLaM paper. We utilized 2xT4 GPUs available on Kaggle to run the LLaMa model on our data.

To solve the problems using the subquestions, we first passed the problem context (game facts, rules, and rule preferences) to the problem solver LLM. We then iteratively prompted the LLM to solve each subquestion. Finally, the original question was prompted to the LLM, which the LLM solved using the built up context.

Similarly, we tested ChatGPT and Gemini serving as question decomposers by prompting them to “Generate subquestions that would help lead to the answer of this question…”. The subquestions were then used in a manner similar to the questions generated by the LLaMa model.

The efficacy of each decomposer-solver configuration was thoroughly assessed, not only against baseline models where the LLMs were tasked with directly answering complex questions but also across different combinations of decomposers and solvers. Additionally, we employed Chain of Thought (CoT) as another comparison to evaluate the effectiveness of our decomposition and solving approach. We evaluated the correctness of a response by the presence of the label keywords (“proved”, “disproved”, “unknown”) and manually evaluated any ambiguous cases (e.g. “Therefore, x is proved, making the original statement disproved.”) We analyzed the accuracy, precision, recall, and F1-score for each label.

Below, we note any other remarks with regards to specifically using Gemini and ChatGPT.

Gemini The use of Gemini was facilitated by Google Colab, which, while free, provided sufficient computational resources for the tasks. Specific safety settings were also configured for Gemini to ensure that all content was processed consistently, with thresholds set to prevent the blocking of content that could be essential for forming complete responses to the questions posed.

ChatGPT We accessed ChatGPT via the OpenAI API. For all prompts, the ChatGPT system instructions were “You are a helpful bot that can answer reasoning questions based off of board game situations.” The total inference cost was approximately $25.

5 Experimental Results

In examining the accuracy of various approaches, the integration of ChatGPT with the LLaMa decomposer emerges as a front-runner, achieving the highest model accuracy at 0.530. Using the LLaMa decomposer with ChatGPT improved results by

8%, while it improved the Gemini results by 4.2%.

For both ChatGPT and Gemini, CoT approaches decreased the accuracy of responses compared to the zero shot setting. This could be attributed to the additional context forming the chain of thought confounding the information of the prompted question, leading to less accurate responses.

The fine tuned LLaMa question decomposer increased the accuracy for both ChatGPT and Gemini, but more so for ChatGPT. This could suggest a latent synergy between the LLaMa model and ChatGPT as the LLaMa model was fine tuned on GPT-3 responses. Alternatively, it could have been due to the poor responses given by Gemini.

In our study, we found that Gemini provided, concise, often direct answers (such as just 'disproved') without delving into the underlying reasoning. In contrast, ChatGPT's answers were more elaborate, detailing the reasoning process that lead to its conclusions. When these responses were used with the LLaMa model, the additional contextual information and explicit reasoning chains present in ChatGPT's answers provided a richer context that assisted in generating better subquestions. The improved subquestions then lead to improved context being generated when the solver LM iteratively solved the subquestions, allowing for more accurate final responses.

The label distribution for ChatGPT and Gemini in the zero shot cases demonstrate how the LLMs are predisposed to saying disproved. Additionally, both models reported the 'unknown' label less than 20% of the time when it accounted for 33% of the ground truth. This confirms the findings from the authors of BoardgameQA who cited that reasoning with unknown labels is particularly challenging for few shot LMs (Kazemi et al., 2023). No approach achieved an F1 score greater than 0.5 for the unknown label, asuggesting that this approach still has LLMs struggle with identifying unknown information (Yin et al., 2023).

6 Conclusions

Our investigation into the use of large language models for solving complex natural language reasoning problems has highlighted several key findings. Firstly, the integration of a decomposer LLM with foundational LLMs enhances the ability to handle reasoning tasks involving conflicting information. By decomposing complex problems into simpler subproblems, our approach not only improves the accuracy of responses but also offers a more efficient and flexible problem-solving mechanism compared to traditional single-model approaches.

The experimental results from BoardgameQA datasets demonstrate the efficacy of our approach: the combination of ChatGPT with the LLaMa decomposer outperformed other configurations. This suggests that the decomposer LLM aids in navigating the complexities of the dataset's reasoning challenges, especially when paired with a robust solver like ChatGPT. This modular approach not only addresses the limitations of single-model systems in handling detailed, complex reasoning but also allows for adaptability to various types of reasoning tasks beyond those presented in our current study. Still, solving natural language reasoning questions with contradictory information remains a challenging problem.

In conclusion, our research affirms the value of leveraging multiple LLMs in a decomposed, iterative manner to tackle complex reasoning problems. It opens up new avenues for further refining these techniques and exploring their application across more diverse and challenging datasets.

7 Future Work

We outline several directions of future research.

Fine Tuning Natural Language Question Decomposer In our analysis, we tested the LLaMa 13B LM question decomposer which was fine tuned for problem decomposition on mathematical reasoning problems. Although it was successful in increasing the accuracy on our chosen dataset, we suspect a question decomposer fine tuned specifically for natural language reasoning problems would perform better. Furthermore, decomposers of various sizes could be tested to determine the ideal size of a decomposer LM, factoring both computational complexity and accuracy.

Improving the Subproblem Generation The `DaSLaM` question decomposer was fine tuned to generate questions that would lead to correct answers using reinforcement learning; however, the quantitative quality of subproblems remains a question. Future research can explore additional ways to evaluate the quality subproblems in helping reach the solution to queried questions in application-specific and general settings. Moreover, future research can explore the representation of LM subanswers and their impact on the final answer.

8 Division of Labor

Aayushi: My work focused on coding the script for using the GPT-3.5 API to generate sub-questions from a subset of the reasoning questions within the BoardGameQA dataset, aiming to dissect complex questions into manageable sub-problems. It also included experimenting with different prompts to yield optimal sub-questions such that they led up to answering the final question without throwing out random sub-problems and structuring the format of the generated sub-questions. I also helped with prompting for the Gemini API. Moreover, I implemented and determined the cost analysis for using external resources.

Sanah: During the first half of the project, I was tasked with picking the right version of the BoardgameQA dataset. This involved determining which version had the best mix of contradictions and difficulty level, ensuring it was suitable for our needs. I also conducted trials with Gemini to assess its potential as a decomposer in our framework. In the second half, I was responsible for writing the script to interact with ChatGPT by using the GPT-3.5 API to iteratively prompt the LLM to solve the subproblems generated by the decomposer. The script initiates a chat session with ChatGPT and sends the context of the problem, collects an initial response. It then cycles through each subproblem, sends them to ChatGPT and gathers the responses. In the last step, the final question is sent to the LLM to obtain the conclusive response.

Muyuan: In the initial stages of our project, my efforts were concentrated on the development and execution of scripts intended for the fine-tuning of both the Gemma 7B and initially the LLaMA 7B models. Despite the promising direction, this phase encountered significant setbacks due to memory constraints, rendering fine-tuning efforts unfeasible. Consequently, the project's focus was strategically redirected towards leveraging LLaMA models specifically for response generation. Subsequent to this phase, in collaboration with Chetan Sah, I engaged in the extension of our methodology to incorporate the LLaMA 13B model. In the latter part of the research, I assumed full responsibility for the integration and utilization of the Gemini model. This role encompassed the comprehensive setup of the experimental framework, the execution of all methods utilizing Gemini as a solver, and the meticulous analysis of performance metrics resultant from these implementations. Moreover, I collaborated with Chetan in generating results for ChatGPT3.5 using the CoT method, further contributing to our analytical capabilities and enriching our dataset with diverse solution strategies.

Chetan: In the project, I was tasked with implementing the LLaMA 13B model to address a complex logical problem, receiving support from Muyuan in this effort. Additionally, I managed the application of chain-of-thought prompting with ChatGPT-3.5, collaborating with Muyuan to enhance our model's reasoning capabilities. My responsibilities also included creating visual graphs that displayed our findings and provided valuable insights based on the results. These contributions were essential for documenting the project's progress and supporting our research objectives.

Dennis: In the first half of the project, I attempted fine tuning a Gemma 2B model for question decomposing, but we abandoned that approach after the model gave very poor answers. During the next half of the project, I ported Muyuan's code to load the LLaMa 13B model and LoRA adapters from `DaSLaM` into Kaggle where we had access to 2xT4 GPUs and a weekly guaranteed 30 hours of compute. I ran the LLaMa model for $\sim$ 60 hours to generate the subquestions for ChatGPT and Gemini. I also cleaned the output of the LLaMa model to omit any formatting issues and make it a standard format. I also wrote a script that was extended by Chetan and Sanah to connect to the OpenAI API to query the questions and receive the LLM's responses. I ran Sanah's and Chetan's implementations to gather results for ChatGPT. I also wrote a script to process the results of ChatGPT to determine the label the model predicted based on keywords. I also wrote and executed a script to manually evaluate any ambiguous cases and adjusted those records. Our code and data is available at here.

References

Daman Arora, Himanshu Gaurav Singh, and Mausam. 2023. Have llms advanced enough? a challenging problem solving benchmark for large language models.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions.

Dan Hendrycks, Collin Burns, Saurav Kadavath, et al. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Gurusha Juneja, Subhabrata Dutta, Soumen Chakrabarti, et al. 2024. Small language models fine-tuned to coordinate larger language models improve complex reasoning.

Mehran Kazemi, Quan Yuan, Deepti Bhatia, et al. 2023. Boardgameqa: A dataset for natural language reasoning with contradictory information.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, et al. 2023. Decomposed prompting: A modular approach for solving complex tasks.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, et al. 2024. Gpt-4 technical report.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, et al. 2024. A systematic survey of prompt engineering in large language models: Techniques and applications.

Gemini Team, Rohan Anil, Sebastian Borgeaud, et al. 2024. Gemini: A family of highly capable multimodal models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. 2023. Llama: Open and efficient foundation language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Yunhu Ye, Binyuan Hui, Min Yang, et al. 2023. Large language models are versatile decomposers: Decompose evidence and questions for table-based reasoning.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, et al. 2023. Do large language models know what they don't know?

Ori Yoran, Tomer Wolfson, Ben Bogin, et al. 2023. Answering questions by meta-reasoning over multiple chains of thought.

Chuanyang Zheng, Zhengying Liu, Enze Xie, et al. 2023. Progressive-hint prompting improves reasoning in large language models.

Denny Zhou, Nathanael Schärli, Le Hou, et al. 2023. Least-to-most prompting enables complex reasoning in large language models.