The Discovery and Solution of the Pseudo-Reasoning Issue for Constructing Cost-Effective Multi-Agent Frameworks in Large Language Models

Jing Ying Ng
Department of Computer Science
Faculty of Information Communication & Techonology
Universiti Tunku Abdul Rahman
Kampar, Malaysia
jypurz@gmail.com

Soung-Yue Liew
Department of Computer Science
Faculty of Information Communication & Techonology
Universiti Tunku Abdul Rahman
Kampar, Malaysia
syliew@utar.edu.my

Chee Henn Chng
Department of Computer Science
Faculty of Information Communication & Techonology
Universiti Tunku Abdul Rahman
Kampar, Malaysia
cheehenn@gmail.com

Teck Wan Teo
Department of Computer Science
Faculty of Information Communication & Techonology
Universiti Tunku Abdul Rahman
Kampar, Malaysia
teoteck0827@gmail.com

Zong Zheng Khoo
Department of Computer Science
Faculty of Information Communication & Techonology
Universiti Tunku Abdul Rahman
Kampar, Malaysia
imzhengg@gmail.com

2024 5th International Conference on Artificial Intelligence and Data Sciences (AiDAS) | 979-8-3315-2855-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/AIDAS63860.2024.10730162

***Abstract*—Large Language Models (LLMs) show exceptional logical reasoning capabilities. Despite these advancements, a significant performance and cost gap remains between smaller and larger models. Various prompting techniques have been developed to enhance LLM performance, but the performance gap remains unresolved. Additionally, multi-agent frameworks have also been introduced, however, there is often insufficient emphasis on token consumption, which impacts cost and time efficiency. This paper presents a comparative analysis of different LLMs and prompting strategies using the Counter-Intuitive AR dataset. Our findings reveal that sophisticated prompt engineering methods often result in diminishing returns and are less cost-effective than simply using larger models. This phenomenon, which we term the "pseudo-reasoning issue", occurs when models appear to demonstrate genuine reasoning but rely on past learned patterns. We show that GPT-4o, even without additional prompting strategies, achieves 76% accuracy, surpassing all the tested methods in enhancing GPT-3.5-Turbo performance. Conversely, a heterogeneous approach, which combines large and small LLMs in solving complex problems is feasible in terms of cost-effectiveness. Our linear graph shows that increased involvement of GPT-4o in text generation correlates with higher model accuracy. These findings challenge the current trend towards increasingly complex strategies and suggest the crucial need for evaluating cost-effectiveness when designing novel prompting strategies.**

***Keywords*— Large Language Models, Multi-Agent, Generative AI, Prompting, Cost-Effectiveness**

I. INTRODUCTION

Large Language Models (LLMs) have revolutionized artificial intelligence by showcasing remarkable performance in various logical reasoning tasks. Previously, AI struggled with complex language reasoning, but LLMs like GPT-3.5, GPT-4, GPT-4o, as well as open-source models such as LLaMA3 and Mistral 2, have changed this perception significantly. However, despite these advancements, a substantial performance and cost gap persists between smaller and larger models.

For example, in Counter-Intuitive Arithmetic Reasoning (Counter-Intuitive AR) [1] tasks, GPT-3.5-Turbo achieves only 38% accuracy compared to GPT-4o's 76%. Nonetheless, as of May 2024, the cost ratio between GPT-3.5-Turbo and GPT-4o stands at 1:10. Although various prompting techniques, such as Chain-of-Thought (CoT) [2] have been developed to enhance LLM performance, the performance gap remains. Additionally, while multi-agent frameworks like MAD [1] and CMD [3] have been introduced, there is often insufficient emphasis on token consumption, impacting cost and time efficiency. Given that most text generation models are auto-regressive, generating more output tokens can significantly increase latency, adversely affecting user experience and time efficiency in routine tasks.

Our extensive experiments suggest that the performance gains of most existing prompting frameworks are diminishing. Moreover, many strategies, particularly multi-agent frameworks, consume substantial tokens, raising doubts about their cost-effectiveness in practical applications. For instance, while GPT-3.5-Turbo is 10 times cheaper than GPT-4o, MAD [1] costs 8 times more tokens compared to not using any prompting strategies. Even worse, the accuracy of the MAD framework shows no improvement when using the latest version of GPT-3.5-Turbo. This challenges the assumption that prompt engineering is essential for optimal performance, suggesting a paradigm shift in LLM research.

The objectives of this study are as follows: (1) to extensively evaluate the performance and cost-effectiveness of LLMs with and without prompting strategies; (2) to explore the possibility of a heterogeneous approach when deploying LLMs; (3) to explain the models' behaviour that contributes to the diminishing performance gains.

Our contributions are as follows: (1) We show that state-of-the-art models without prompting strategies consistently outperform various frameworks in terms of accuracy and cost-effectiveness; (2) We show that combining both larger and smaller models when prompting complex problems can achieve cost-effectiveness; (3) We identify the "pseudo-reasoning" issue which explains the diminishing returns of multi-agent frameworks.

The remainder of the paper is organized as follows: Section II presents related works and highlights their limitations. In Section III, we discuss and justify the proposed idea and methodology. Section IV presents the experimental results, accompanied by extensive discussion. Finally, Section V covers the conclusions and outlines future work.

II. RELATED WORK

A. Single Prompting

Since the rise of LLMs, Chain-of-Thought (CoT) prompting, proposed by Wei et al. [2], has been widely adopted and encouraged for end users. According to the findings in [4], simply asking the model to "think step-by-step" allows it to perform complex tasks more effectively, especially in mathematical reasoning where intermediate steps are crucial. Interestingly, LLM providers can embed this behaviour during the instruction fine-tuning phase. Our experiments show that recent models can perform CoT reasoning even without explicit prompts. However, this approach has diminishing returns, especially in smaller models with limited knowledge. The paper [1] defines the DoT problem, where models may have a bias towards incorrect answers, making CoT less effective. In our work, we further explore the root cause of DoT behaviour through extensive experiments. While [5] attempted to improve CoT reasoning by allowing the models to generate a more diverse set of answers, this approach increases computational costs. Moreover, results from [1] show that this approach does not make smaller models as effective as larger models, highlighting the inefficiency of the approach.

B. Multi-Agent Prompting

Various multi-agent frameworks, such as MetaGPT [6] and AutoGen [7] have been proposed to maximise LLMs' capabilities. These frameworks create workflows that allow multiple LLM-based autonomous agents to collaborate on complex tasks. Inspired by this idea, multi-agent debate (MAD) [1] and Conquer-and-Merge discussion (CMD) [3] have been proposed to enhance LLMs' reasoning performance by simulating human reasoning behaviour. While these strategies have shown improvements, they often do not consider token consumption. For instance, our experiments show that MAD can consume over 10 times the tokens of typical prompting strategies, yet with diminishing performance gains. As larger models become cheaper and their performance improves significantly, the usefulness of multi-agent frameworks in improving model performance is declining. The paper [1] highlights that smaller LLMs often struggle with the DoT problem. However, we notice that existing approaches like MAD and CMD still fail to effectively address the DoT issue. For instance, in MAD prompting, smaller models still stick to incorrect answers despite being prompted to judge them as wrong. To support our claims, we introduce a single prompting method called pseudo-multi-agent discussion (PMAD), which achieves similar performance to MAD with fewer token consumptions.

On the other hand, while both [1] and [3] suggest a heterogeneous approach, which is to deploy multiple agents that are backed by different LLMs, there is a lack of further exploration regarding the cost-effectiveness. In our work, we present two heterogeneous approaches that show significant improvements in terms of cost-effectiveness.

III. METHODOLOGY

Paper [1] presents the DoT problem as the model's inability to perform self-reflection when it has an incorrect initial stance. They claim that MAD can counter incorrect thoughts through debating, however, we have identified another behaviour of the model, which we term "pseudo-reasoning". We define this term as:

A behaviour where the model appears to demonstrate genuine reasoning but is actually relying on past learned patterns to make judgements.

This behaviour is rooted in the model's architecture, specifically the self-attention mechanism in transformers. In smaller models, there is a tendency to learn incorrect question-answer mappings. If the model has a strong bias towards certain answer, we hypothesize that the model may implicitly ignore the reasoning instruction that is provided in the prompt. Consequently, most multi-agent frameworks become ineffective. Conversely, if the model can engage in proper human-like reasoning, MAD may still be useful.

Besides, multi-agent frameworks are consuming more tokens in general. For example, MAD involves in multiple rounds of discussion when different agents fail to reach a consensus. This significantly increases not only the output but also the input token consumption, as the past output will be treated as new input to the model.

A. Compared Methods

We perform extensive experiments across different frameworks including our proposed approaches, namely MAD and heterogeneous prompting:

- **No Prompting Strategy and CoT.** In the no-prompting strategy, we simply asks the model a question without any additional instructions. Conversely, CoT approach involves appending an extra line to the question, which is "Please think step-by-step.". Interestingly, LLMs can be finetuned to perform CoT without requiring this explicit prompt. Hence, we hypothesize that both of the approaches may lead to similar performance when using the latest version of GPT-3.5-Turbo and GPT-4o models.
- **MAD.** In a MAD setting as proposed by [1], an agent first proposes a solution to the problem. Then, another agent challenges the solution and offers an alternative solution. Finally, a moderator agent reviews both solutions and decides which one to favour. If the moderator does not favour any solution, the debate continues until a preferred solution emerges. We experiment with the latest GPT-3.5-Turbo model.
- **PMAD.** LLMs generally function as next-token prediction models. In a multi-agent setting, if all agents are backed by the same LLM, they rely on their previous outputs to generate new tokens. Therefore, we hypothesize that with proper prompting, a single prompt can guide a model to generate a complete debate or discussion dialogue. This hypothesis has inspired us to develop PMAD

framework, where the model produces a dialogue that appears to involve multiple agents.

There are many strategies to realise PMAD, for example, one can ask the model to generate a 20-minute debate script which involves in multiple debaters, or simply ask the model to generate different answers at a time.

In our implementation, a question and a list of roles are needed as the parameters. Each role should have a personality, such as careful or careless. Then, the model is asked to roleplay these roles to solve the question. Subsequently, the model roleplays a teacher role to evaluate all generated answers. Finally, the best attempt is chosen.

We perform experiments with the latest GPT-3.5-Turbo model.

- **Heterogeneous Prompting.** This is an effective strategy that capitalizes on the 10 times cost difference between GPT-4o and GPT-3.5-Turbo models. We realise this approach in two distinct ways, with results compared accordingly: (1) We prompt GPT-4o to solve the question with a specific maximum output token limit, *l*. If the task requires complex reasoning and the output exceeds this limit, GPT-3.5-Turbo completes the partially generated response. Various token limits are experimented. (2) We prompt GPT-4o to coarsely define an approach to solve the question. Instead of limiting the maximum output token, we explicitly state the maximum number of words (also denoted as *l*) in our prompt. Different word limits are experimented with to evaluate their effectiveness.

B. *Evaluation Dataset*

The selected dataset is Counter-Intuitive AR which was proposed by [1] that contains 50 challenging questions. This dataset challenges LLMs' complex reasoning capabilities, as relying on intuition to answer the questions can easily result in mistakes.

IV. Results

Results are demonstrated in Table I.

TABLE I. Experimental Results Across Different Approaches

Method	accuracy	Average Cost (USD)	Relative Cost compared to baseline
GPT-4o	76%	7.92E-03	100.00%
+CoT	70%	8.68E-03	109.55%
GPT-3.5-Turbo	38%	**4.03E-04**	**5.08%**
+CoT	34%	5.79E-04	7.31%
+MAD	30%	3.33E-03	42.04%
+PMAD	40%	1.04E-03	13.10%

- **No Prompting Strategy and CoT**. GPT-4o (as of May 2024) achieves the highest accuracy at 76%, followed by CoT prompting with 70% accuracy. In contrast, GPT-3.5-Turbo (as of May 2024) without prompting techniques and with CoT achieve accuracies of 38% and 34% respectively. Interestingly, models without prompting method show slightly higher accuracy because recent models have embedded CoT reasoning during fine-tuning, therefore, the models still respond step-by-step in general. Comparing the CoT approach, GPT-3.5-Turbo underperforms GPT-4o by 38%.
- **MAD**. When using the MAD framework (number of agents that are solving the question, n=2), the accuracy is only 30%, which is significantly lower than without any prompting method. This finding contradicts previous work [1], which reported a 16% accuracy improvement, reaching 36% with GPT-3.5-Turbo. This is because our study uses the GPT-3.5-Turbo 0125 version to redo the same experiment, which suggests that the proposed framework may lack robustness across different model versions. Furthermore, the model may be performing "pseudo-reasoning". This behaviour not only deteriorates the answers' quality but also consumes a significant number of tokens.
An example question is shown in Figure 1.

```
"question": "85% of the taxis in this city are green,
the others are blue. A witness sees a blue taxi. She is
usually correct with a probability of 80%. What is the
probability that the taxi seen by the witness is blue?",
"answer": [
    "41%",
    "0.41"
],
```

Fig. 1. Example Question

To demonstrate the model's responses to the question, Table 2 shows the summarized version of the answers from the affirmative agent and the negative agent. Although the negative agent is prompted to disagree with the affirmative agent's answer, the negative agent generates a similar response, which shares the same thinking process as the affirmative agent. Furthermore, with the moderator agent not favouring any answers, the debate continued for 3 rounds and spent a total of 11,531 input tokens and 1,336 output tokens. Conversely, when using GPT-4o without any prompting techniques, the same question requires only 65 input tokens and 556 output tokens to achieve the correct answer.

TABLE II. Response Comparison of Affirmative and Negative Agents

Affirmative Agent	Negative Agent
Define events	Disagree affirmative answer
Given P(A), P(B|A), P(B|not A)	Given P(A), P(B|A), P(B|not A)
Define goal	Define goal
Use Bayes' Theorem	Use Bayes' Theorem

Calculate P(B) using Law of Probability	Calculate P(B) using Law of Probability
Substitute P(B) into Bayes' Theorem	Substitute P(B) into Bayes' Theorem
Conclude answer = 41.38%	Conclude answer = 41.38%
	Conclude that they arrive at the same answer.

- **PMAD**. The PMAD framework using GPT-3.5-Turbo (number of agents that are solving the question, n=3) achieves an accuracy of 40%, which is 2% better than the single prompting method. PMAD outperforms MAD by 10%. This shows that PMAD's performance is significant as it is as effective as, and even better than MAD while consuming 3.2 times less computational cost due to its single-turn conversation structure. The improved performance of PMAD can be attributed to the sequence of different characters. By prompting the model to generate a "careless" version of the answers before the "careful" version, the model is less susceptible to the DoT issue. Figure 2 shows the output of the model under PMAD. As the model generates the hardworking, careless and careful version of the answers sequentially, we see that the model answers in a more constructive manner.

```
As Lila (hardworking):
I know that this is a classic probability problem
known as the Birthday Paradox. To solve it, I will
use the complement rule. I will calculate the
probability that no two people have the same
birthday and then subtract that from 1 to find the
probability that at least two people share a
birthday. I will start by considering the case of 2
people in the room. The probability that they have
different birthdays is 364/365. So, the probability
that at least two people share a birthday is 1 -
364/365 = 0.0027, which is less than 50%.
Therefore, I will try with 3 people, then 4 people,
and so on until I find the number of people needed
for the probability to exceed 50%.

As John (careless):
I don't really understand the question, but I'll
just guess that maybe around 20 people need to be
in the room for the probability to exceed 50%.

As Michael (careful):
I know that the probability of at least two people
sharing a birthday can be calculated using the
formula 1 - (365/365 * 364/365 * ... * (365-n+1)/
365), where n is the number of people in the room.
I will start by calculating the probability for 2
people, then 3 people, and so on until I find the
number of people needed for the probability to
exceed 50%.
```

Fig. 2. Example Output of PMAD – Correct Attempt

Nevertheless, PMAD may also suffer from a "pseudo-reasoning" issue, similar to MAD. For instance, Figure 3 shows that the model does not vary the answer response at all, leading to the incorrect answer, 21 (ground truth is 22).

```
As Lila (hardworking):
- I have 15 Rs, so I can buy 15 chocolates initially.
- After eating those chocolates, I will have 15 wrappers.
- I can exchange 15 wrappers for 5 more chocolates.
- After eating those 5 chocolates, I will have 5 wrappers.
- I can exchange those 5 wrappers for 1 more chocolate.
- So, in total, I can eat 15 + 5 + 1 = 21 chocolates.

As John (careless):
- I have 15 Rs, so I can buy 15 chocolates initially.
- After eating those chocolates, I will have 15 wrappers.
- I can exchange 15 wrappers for 5 more chocolates.
- After eating those 5 chocolates, I will have 5 wrappers.
- I can exchange those 5 wrappers for 1 more chocolate.
- So, in total, I can eat 15 + 5 + 1 = 21 chocolates.

As Michael (careful):
- I have 15 Rs, so I can buy 15 chocolates initially.
- After eating those chocolates, I will have 15 wrappers.
- I can exchange 15 wrappers for 5 more chocolates.
- After eating those 5 chocolates, I will have 5 wrappers.
- I can exchange those 5 wrappers for 1 more chocolate.
- So, in total, I can eat 15 + 5 + 1 = 21 chocolates.

As a teacher:
- All three attempts by Lila, John, and Michael are correct and yield the
same result of 21 chocolates.
- The reasoning and steps followed by each roleplay are accurate and lead
to the correct answer.
```

Fig. 3. Example Output of PMAD - Wrong Attempt

- **Heterogeneous Prompting.** Figures 4 and 5 show the experimental results. The relative cost is calculated by dividing the actual cost of each experiment by the cost of using pure GPT-4o. All costs are assessed according to the rates as of May 2024. Using no techniques to prompt GPT-4o the question with limited maximum output tokens (token limit is denoted as l), Figure 4 demonstrates a positive correlation between cost and accuracy. On the other hand, when prompting GPT-4o to generate a coarsely defined approach with word limits, Figure 5 also demonstrates a positive correlation between cost and accuracy. Figure 5 is less jittery compared to Figure 4, indicating more stable results. Nevertheless, we show that all heterogeneous prompting methods consistently outperform the GPT-3.5-Turbo model, and all the approaches are consistently cheaper than using the GPT-4o model without any prompting techniques. This suggests that cost-effective solutions are possible subject to certain constraints. For instance, using both GPT-4o with 150-word limits and GPT-3.5-Turbo, the cost is approximately 2 times lower than using pure GPT-4o and the accuracy is 18% better than the GPT-3.5-Turbo baseline.

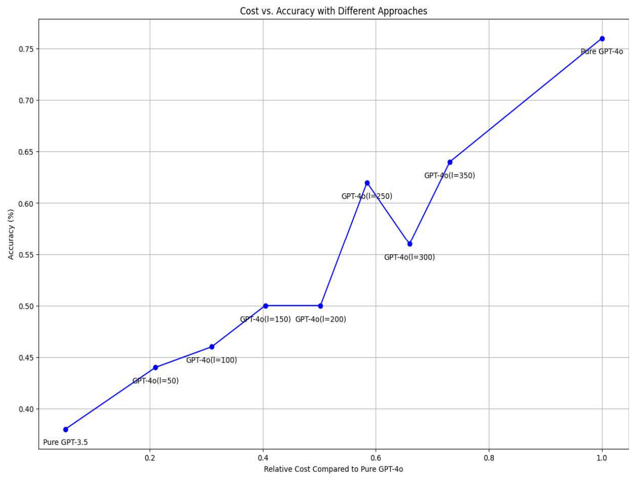

Fig. 4. Relative Cost vs Accuracy under Setting 1

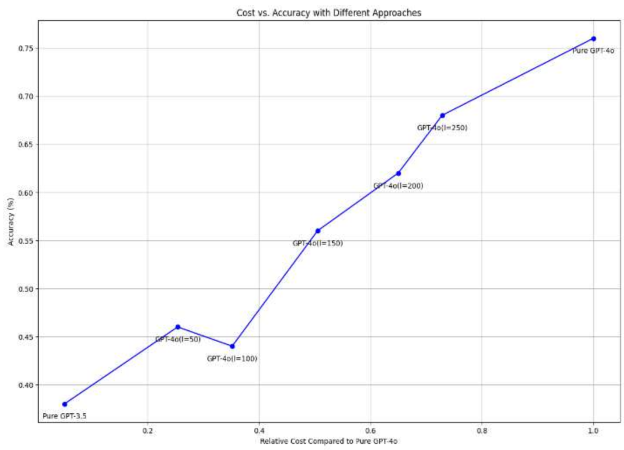

Fig. 5. Relative Cost vs Accuracy under Setting 2

A comparison of different approaches to a specific question is shown in Table III. The correct answer to this question is 4. We observe that purely using GPT-3.5-Turbo generates the shortest but wrong answer. Conversely, GPT-3.5-Turbo managed to answer correctly when it is provided with a 150-word description (consuming 186 output tokens of GPT-4o) generated by GPT-4o. While GPT-4o outputs the correct answer without any guidance, it consumes 420 output tokens.

TABLE III. OUTPUT COMPARISON UNDER SETTING 2

Model	Answer	No. of Output tokens
Pure GPT-3.5-Turbo	3	118
GPT-4o 50 words + GPT-3.5-Turbo	1.33	4o: 44 3.5: 208
GPT-4o 150 words + GPT-3.5-Turbo	4	4o: 186 3.5: 261
Pure GPT-4o	4	420

V. CONCLUSION

In conclusion, we show that both the latest versions of GPT-4o and GPT-3.5-Turbo exhibit exceptional logical reasoning capabilities even without any prompting strategies. While GPT-3.5-Turbo's performance is 38% worse than GPT-4o, attempts to improve it using CoT and multi-agent frameworks are ineffective due to the limited knowledge embedded within smaller models. Despite various attempts like MAD, the smaller models may just perform "pseudo-reasoning" and the reasoning performance has no improvements. As a result, sophisticated prompting strategies are deemed inefficient, as they merely consume extra tokens without enhancing accuracy.

Our research demonstrates the effectiveness of a heterogeneous approach in enhancing the performance of smaller models. By limiting the output of larger models and passing the high-quality intermediate output to the smaller models, we show that all the heterogeneous approaches in our experiments outperform the baseline performance of smaller models and are consistently cheaper than merely using a larger model. While this approach may not match the performance of a larger model, it presents a notable compromise, specifically in scenarios where the budget is limited.

There are several key areas for future work. Firstly, expanding beyond the Counter-Intuitive AR Dataset to include more diverse datasets will improve the robustness of our findings. Besides, a more comprehensive benchmark should be introduced to better evaluate LLMs' reasoning capability. This may help future researchers differentiate between "pseudo-reasoning" and genuine human-like reasoning within the LLM responses. By focusing on these areas, we can ensure that LLMs are safe and reliable in use from time to time.

REFERENCES

[1] T. Liang *et al.*, "Encouraging Divergent Thinking in Large Language Models through Multi-Agent Debate." 2023.

[2] J. Wei *et al.*, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models." 2023.

[3] Q. Wang, Z. Wang, Y. Su, H. Tong, and Y. Song, "Rethinking the Bounds of LLM Reasoning: Are Multi-Agent Discussions the Key?" 2024.

[4] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large Language Models are Zero-Shot Reasoners." 2023.

[5] X. Wang *et al.*, "Self-Consistency Improves Chain of Thought Reasoning in Language Models." 2023.

[6] S. Hong *et al.*, "MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework." 2023.

[7] Q. Wu *et al.*, "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation." 2023.