Enabling Synergistic Knowledge Sharing and Reasoning in Large Language Models with Collaborative Multi-Agents

Ayushman Das[1,2],Shu-Ching Chen[1], Mei-Ling Shyu[2], Saad Sadiq[3]
[1]Data Science and Analytics Innovation Center
University of Missouri-Kansas City, Kansas City, MO, USA
[2]School of Science and Engineering
University of Missouri-Kansas City, Kansas City, MO, USA
[3]Microsoft
Seattle, WA, USA
ad5f2@mail.umkc.edu, s.chen@umkc.edu, shyum@umkc.edu, saad.sadiq@microsoft.com

***Abstract*—Despite the significant advancements in the field of Natural Language Processing (NLP), Large Language Models (LLMs) have shown limitations in performing complex tasks that require arithmetic, commonsense, and symbolic reasoning. Reasoning frameworks like ReAct, Chain-of-thought (CoT), Tree-of-thoughts (ToT), etc. have shown success but with limitations in solving long-form complex tasks. To address this, we propose a knowledge-sharing and collaborative multi-agent assisted framework on LLMs that leverages the capabilities of existing reasoning frameworks and the collaborative skills of multi-agent systems (MASs). The objectives of the proposed framework are to overcome the limitations of LLMs, enhance their reasoning capabilities, and improve their performance in complex tasks. It involves generating natural language rationales and in-context few-shot learning via prompting, and integrates the reasoning techniques with efficient knowledge-sharing and communication-driven agent networks. The potential benefits of the proposed framework include saving time and money, improved efficiency for computationally intensive reasoning, and the ability to incorporate multiple collaboration strategies for dynamically changing environments.**

***Index Terms*—large language model (LLM), multi-agent system (MAS), knowledge sharing, reasoning**

I. Introduction

The field of Natural Language Processing (NLP) has witnessed a significant transformation with the advent of large language models (LLMs). These models have demonstrated enhanced performance and sample efficiency, thereby revolutionizing the NLP landscape. However, despite their impressive capabilities, LLMs have shown limitations in performing complex tasks that require arithmetic, commonsense, and symbolic reasoning [22], [24], [26].

Recent research has explored methods to enhance the reasoning abilities of LLMs. One such approach involves generating natural language rationales that lead to the final answer, a technique that has proven beneficial for arithmetic reasoning [24]. Another promising avenue is the concept of in-context few-shot learning via prompting, which has shown success in simple question-answering tasks [4], [22].

Despite these advancements, the ability of LLMs to demonstrate reasoning is often seen as a limitation (as shown in Figure 1) [13]. To address this, researchers have proposed chain-of-thought prompting, where the LLM is prompted to generate a thinking steps that imitate the process of reasoning through a problem [22]. This approach has significantly improved model performance across a variety of multi-step reasoning tasks.

The use of agents for reasoning in LLMs presents a dynamic and interactive approach to problem-solving [9]. Each agent, essentially an instance of the language model, contributes to the reasoning process, working collaboratively with other agents to solve a given task. This multi-agent approach allows for the distribution of cognitive load, with each agent focusing on a specific aspect of the task, thereby reducing the complexity of the problem for each individual agent. The communication between agents is crucial in this setup. Agents exchange information and build upon each other's responses to generate a collective output. This interactive reasoning process allows for a more comprehensive and nuanced solution, as agents can build upon the reasoning of other agents.

In multi-agent systems, collaboration is an important form of interaction that enables groups of agents to arrive at a mutual agreement. The potential benefits of agent collaboration include saving time and money, efficiency for computationally intensive collaborations, and the ability to incorporate multiple collaboration frameworks for changing environments. Efficient multi-agent systems can also allow for the use of smaller LLMs like GPT-3.5, LlaMA [19], and LLaMA 2 [20] to produce competitive results with significantly larger models like GPT-4 [12].

This paper introduces a novel framework for task improvement in agent collaboration with large language models. Drawing inspiration from traditional multi-agent networks, we propose a knowledge-sharing and communication-driven multi-agent framework. This framework leverages the reasoning capabilities of existing reasoning frameworks and the collaborative skills of multi-agent systems to enhance task

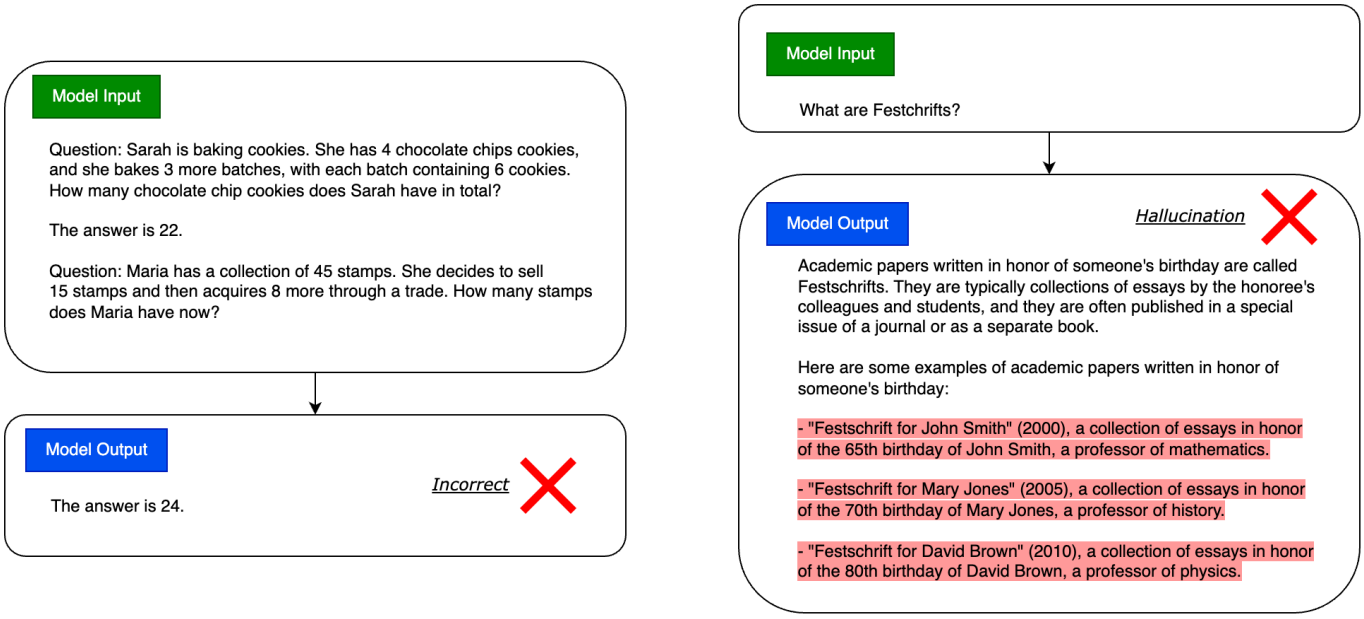

Fig. 1. *Reasoning issues with LLMs*

performance. Our framework builds on the current state-of-the-art practices like ReAct [24], Chain-of-thought (CoT) [22], Tree-of-thoughts (ToT) [23], etc. that the intuition for complex reasoning tasks typically admits multiple reasoning steps to reach a correct answer.

In this paper, we present a comprehensive exploration of reasoning and agent collaboration with large language models, proposing a multi-agent framework for task improvement. The proposed strategies and frameworks aim to overcome the limitations of LLMs and enhance their reasoning capabilities, thereby improving their performance in complex tasks.

The rest of this paper is organized as follows. A comprehensive exploration of the current state-of-the-art approaches in reasoning and agent communication is in Section II. Then, in Section III, we present our proposed framework and how it helps overcome some of the limitations. In Section IV, some open questions and associated challenges are discussed. In the end, Section V concludes this vision paper.

II. Current State-of-the-art

The integration of reasoning methods with agent-based communication has been shown to significantly enhance the performance of LLMs [22], [8]. Agent-based methods involve the exchange of messages between agents, facilitating a dynamic and interactive reasoning process (as shown in Figure 2). These methods significantly improve LLMs' abilities to comprehend and work on complex reasoning-driven tasks, ranging from arithmetic to programming. Here, we discuss the success of these approaches and also some of the challenges associated with scaling them out.

The chain-of-thought (CoT) [22] model operates by prompting the LLM to generate a series of short sentences that mimic the reasoning process a person might employ in solving a task. Instead of directly responding with an answer, the model generates a chain of thoughts that leads to the final answer. This approach not only improves the model's performance in multi-step reasoning tasks but also provides a clear rationale for each step of the reasoning process.

Improving on the CoT model, chain-of-thoughts with self-consistency [21] enables the LLM to generate multiple different chains of reasoning, replacing the naive greedy decoding used in CoT and finally picking the most consistent answer. This allows for the generation of several variations of the chain and then the evaluation of the chain based on their final answer.

On the other hand, the tree-of-thoughts (ToT) [23] model represents the reasoning process as a tree, where each thought is a coherent thought-driven step that serves as intermediate reasoning toward breaking down a complex problem. The model maintains and explores a set diverse of alternative reasoning flows instead of just picking one, and evaluates its current status to make more global decisions. This approach allows the model to self-evaluate the flow with different approaches and incrementally solve the problem by exploring multiple reasoning process.

The graph-of-thoughts (GoT) model [1] expands on ToT, representing the reasoning process as a Directed Acyclic Graph (DAG) allowing for improved reasoning by aggregating over multiple reasoning steps to form an improved and coherent thought. Reasoning sub-steps are scored by checking the difference in generation of the partial solution and keeping the best-scored thoughts. Each of these approaches has explored the notion that thinking through the steps of a complex problem allows for the increased reasoning capacity, and it can take a variety of forms to reach a consistent solution.

Another approach to improving LLMs' reasoning abilities is using ReAct [24]. It proposes the use of partial steps broken

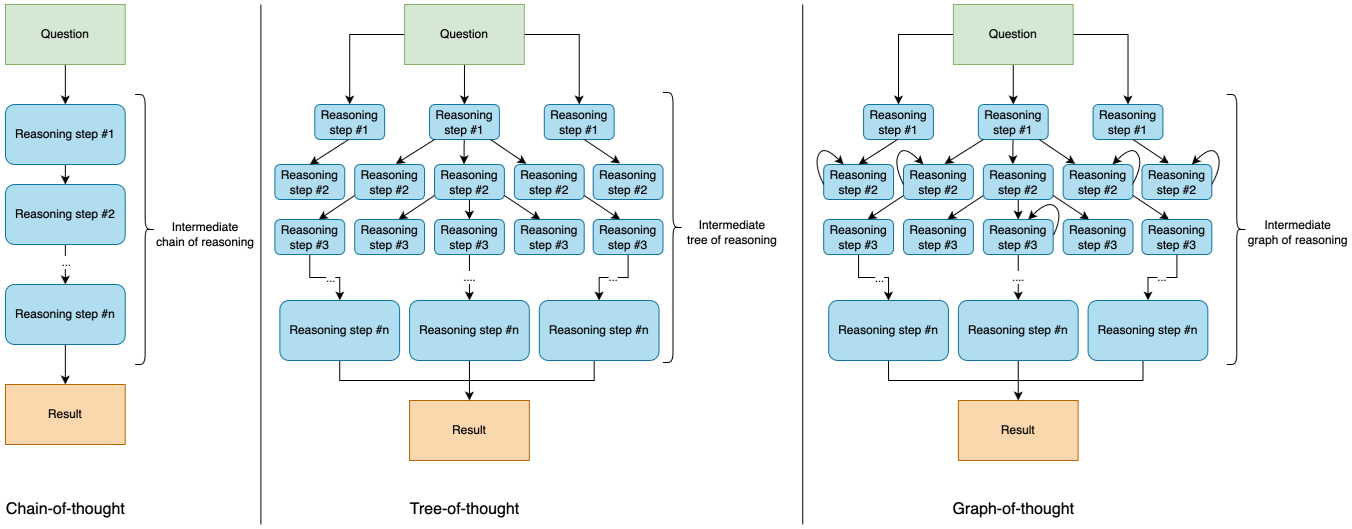

Fig. 2. Chain-of-thought (CoT) vs Tree-of-thoughts (ToT) vs Graph-of-thought (GoT)

down as thought, action, and observation steps that allow the LLM to understand and change its course of action upon being presented with additional context. It shows that an LLM can help orchestrate the result if asked to work through the steps in detail, focusing on a thought pattern being established.

All of these models have proven highly effective in improving the reasoning abilities of LLMs. By mimicking the human reasoning process, these models enhance the models' performance in complex tasks that require arithmetic, commonsense, and symbolic reasoning. However, it is still challenging for these models to scale out, particularly in the context of complex tasks.

Although complex reasoning tasks like basic math word problems have still been difficult for LLMs to solve correctly, LLMs have shown success in generating coherent and correct code because of the inclusion of code in their training data [14]. Approaches like Program-Aided Language (PAL) Models [7] read natural language problems and generate appropriate code backing each of the intermediate code steps to statements in the problem. This allows for reliable computation by an external compiler or interpreter. Offloading the heavy reasoning to an external system allows for integrating traditional software checks and balances like test-driven development.

Lastly, LLM+P [11] explores the idea of offloading reasoning, task planning, and optimization by relying on an external classical planner. The task is broken down and converted into a Planning Domain Definition Language (PDDL) then using a classical planner to find an optimal solution. This moves reasoning out of the LLM chain and uses the LLMs ability to orchestrate a solution and coding abilities to transfer necessary context out of the system.

Agent-driven systems built on LLMs extend reasoning from in-context learning to distributing the cognitive load on a collection of agents, where each agent is an instance of the LLM [5]. Each agent can focus on a specific aspect of the task, thereby reducing the complexity of the problem for each individual agent. This approach also allows for a more dynamic and interactive reasoning process, as agents can build upon the reasoning of other agents, leading to a more comprehensive and nuanced solution.

Applications like AutoGPT [18] and BabyAGI [25] demonstrate the potential of agents in collaborative task-solving and agent-based reasoning. These applications leverage the capabilities of LLMs and the collaborative skills of multi-agent systems to improve task performance. They highlight the potential of agent collaboration in revolutionizing the NLP landscape and for complex task reasoning and solving. The use of multiple agents allows AutoGPT and BabyAGI to distribute the cognitive load of complex tasks to reason on dynamic data sources, which shows promise but difficulty in execution on large and complex tasks. The communication between agents in AutoGPT and BabyAGI is based on the exchange of messages, which can lead to context overflow in long-form reasoning and hallucination. Despite their popularity, these applications have not seen much success in real-world usage because of the concerns about task steering and coherence.

Despite the significant advancements in LLMs, they still face considerable limitations, particularly concerning context length or context windows. LLMs are designed to process a fixed number of tokens, which restricts the amount of context they can consider when generating responses. This limitation becomes particularly evident in long-form reasoning tasks or when the model needs to maintain a coherent narrative over a large number of tokens. The context window of these models is often insufficient to capture all the necessary information, leading to a loss of context and coherence in the generated responses. This can result in the model producing irrelevant or incorrect information, a phenomenon known as hallucination. Furthermore, the fixed context length also limits the model's ability to handle tasks that require understanding and manip-

ulating large amounts of information.

To address this, we propose a knowledge-sharing, communication-driven collaborative multi-agent framework. We build on traditional multi-agent systems, which allow for efficient communication between agents and coordination. We propose a communication-routing framework that enables the routing of messages without the need to parse the entire message, thereby improving the efficiency of the communication process. This not only enhances the reasoning abilities of LLMs but also mitigates the limitations associated with context overflow and hallucination.

Hallucinations: Frameworks around LLM reasoning although effective have shown that they can lead to hallucinations. This occurs when the reasoning process generates outputs that are not based on reality, but rather on the internal logic of the system. For example, a system might generate a hallucination of a person that does not exist in the real world but rather is a product of the system's internal logic.

Ineffective multi-agent collaboration: Another downside of these reasoning methods is that they can lead to ineffective multi-agent collaboration. This occurs when multiple agents are trying to collaborate with each other, but their reasoning methods lead to various paths that are difficult to consider during coordination. This can lead to a breakdown in communication and an inability to reach a coherent and correct answer.

AutoGPT: AutoGPT is an application that leverages the capabilities of LLMs for collaborative task-solving and agent-based reasoning. AutoGPT employs a multi-agent approach to tackle complex tasks. In this setup, each agent is essentially an instance of the language model that works collaboratively with other agents to solve a given task. The agents communicate with each other, exchanging information and building upon each other's responses to generate a collective output.

BabyAGI: BabyAGI is another application that utilizes LLMs for collaborative task-solving and agent-based reasoning. Similar to AutoGPT, BabyAGI employs a multi-agent approach where each agent is an instance of the language model. These agents work together, exchange information, and build upon each other's responses to solve complex tasks. Both frameworks build on agent-based collaboration and multi-step reasoning for complex task-solving.

III. Framework

The proposed framework in this paper involves the integration of a multi-agent system with communicative knowledge-sharing into LLMs. We propose a communication framework, inspired by Knowledge Query Markup Language (KQML) and Knowledge Interchange Format (KIF) [6], to help orchestrate. KQML is a language for agent communication that operates by passing messages with a structured header, rather than relying on the message body. Similarly, KIF is a language format for exchanging information between multiple parties working on a collaborative goal. This approach significantly enhances the efficiency of multi-agent collaboration and communication.

For a complex task like solving a multi-step reasoning problem, we propose a master and a collection of worker agents for breaking down and organizing the execution [5], [22], [23]. The master agent, also known as the orchestrating agent, can be employed to break down the problem into smaller, more manageable tasks (as shown in Figure 3).

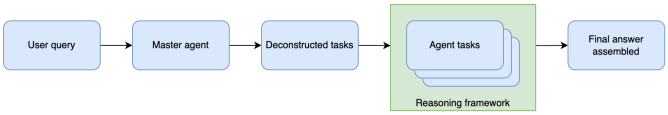

Fig. 3. Flow of the proposed framework with decomposed tasks

The master agent would first analyze the problem and identify the different steps required to solve it. It would then communicate with the worker agents available from an agent zoo (i.e., a collection of available agents) and uses KQML-based communication performatives to assign each worker agent a specific part of the problem to solve [10] (as shown in Figure 4). For instance, it might use the "tell" performative to instruct one agent to perform a certain calculation and the "ask" performative to request another agent to retrieve a specific piece of information.

The final response ($\mathcal{R}$) is the collection of what all the worker agents work on to help respond to the query, where c_i refers to the context for the i^{th} worker agent, a_i refers to the action for the i^{th} worker agent, and r_i refers to the response from the i^{th} worker agent.

$$\mathcal{R} = \{(c_0, a_0, r_0), ..., (c_t, a_t, r_t)\} = (c_i, a_i, r_i)|_{i=0}^{t}$$

The worker agents ($\mathcal{A}$) would then carry out their assigned tasks, using their specialized skills and knowledge to solve their part of the problem [3]. Once they have completed their tasks, they would communicate their results back to the master agent, again using knowledge-interchange performatives. For example, they might use the "reply" performative to report their results. The action to the worker agent is the request from the master agent ($\mathcal{U}$) based on the overall request/context ($\mathcal{C}$).

$$a_i = \mathcal{U}(\mathcal{C})$$

Context (c_i) to the worker agent is provided by the master agent and reasoned independently beyond. This allows for the integration with the reasoning-based frameworks like CoT, ToT, GoT, etc. The provided context to the request ($\mathcal{C}$) is improved by reasoning to provide task-specific steering.

$$c_i = \mathcal{A}(\mathcal{C}, a_i)$$

Response from the worker agent (r_i) is the reply from the worker agent ($\mathcal{A}$) based on the context provided (c_i) and the action requested (a_i).

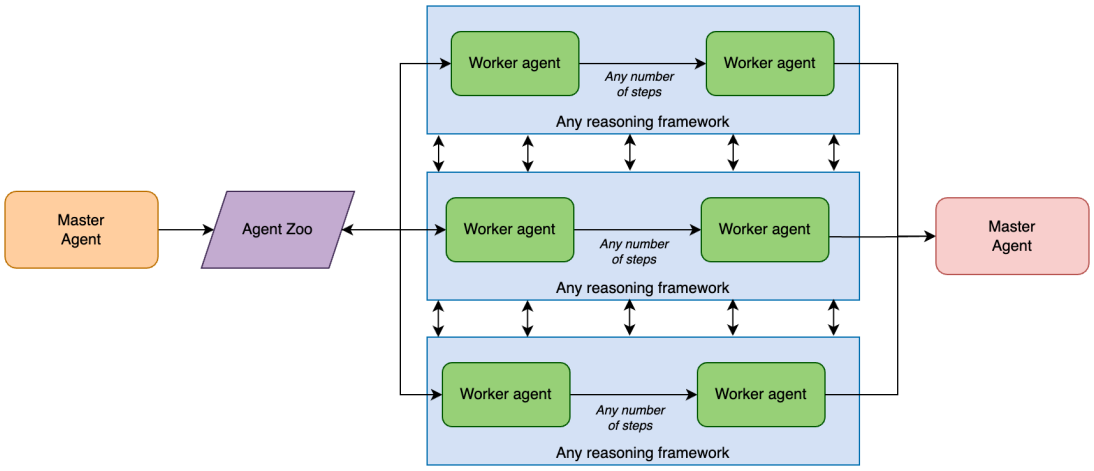

Fig. 4. Proposed framework

$$r_i = \mathcal{A}(c_i, a_i)$$

The master agent would then gather the responses from all the worker agents, using the information they provide to build a comprehensive understanding of the problem. It would then use this understanding to generate the final solution to the problem [2], [10], [16].

This approach allows for efficient coordination of tasks, as the master agent can easily understand the purpose of each message without having to parse the entire message body. Furthermore, by breaking down the problem into smaller tasks and assigning each task to a specialized worker agent, this approach allows for a more coherent and effective solution to complex problems.

The integration of the communication framework with reasoning frameworks like chain-of-thought, tree-of-thoughts, etc. can significantly enhance task coordination in complex tasks. When combined, these frameworks can facilitate efficient and reliable communication between agents, localizing context to tasks, and communicating to orchestrate further processing, thereby improving task coordination.

To highlight the effectiveness of our communication framework, we focus on these 5 areas:

A. Prevention of context overflow and hallucination

Focusing on the structured header minimizes the risk of context overflow, as the model does not need to process a large amount of context to understand the message. This not only improves the efficiency of communication but also ensures that all important information is captured and processed.

Furthermore, separating the steps and the final answer reduces the chances of task-specific context leaking, which can lead to hallucinations. Providing a clear indication of the message's purpose in the structured header ensures that the model focuses on the relevant information and does not generate irrelevant or incorrect information.

This standardization facilitates interoperability between different systems, allowing agents from different systems to communicate effectively with each other. This is particularly beneficial in the context of complex tasks, where multiple agents from different systems may need to collaborate to achieve a common goal.

B. Versatility and scalability

The versatility of the communication framework is one of its key strengths, particularly in the context of LLMs and agent-based communication. The proposed collaborative approach supports a wide range of communication types, including information sharing, query, and command. This versatility allows it to be used for a variety of tasks, making it a highly adaptable tool for agent communication [15]. The communication framework specifies a broad interpretation for agent communication and collaboration and can be implemented in any system regardless of the existing tools and technologies used.

In addition to its versatility, the communication framework also offers significant scalability advantages. It is designed to handle an increasing number of agents without a significant impact on performance. This scalability is crucial in the context of LLMs, where the number of agents can grow rapidly as the complexity of the tasks increases. The framework's ability to efficiently manage and coordinate the communication between a large number of agents makes it a highly scalable solution for agent-based communication.

C. Task coordination

In the context of task coordination, the proposed framework can facilitate efficient coordination between a master or coordinating agent and worker agents. The master agent can use the performative in the message to instruct the worker agents on the tasks they need to perform. The worker agents, in turn, can use the performative to report their progress or results back to the master agent. This allows for efficient coordination of tasks, as the master agent can easily understand the purpose of

each message without having to parse the entire message body. Although we highlight the use of the master and worker agents, task coordination can be achieved with any combination of agent groups. Our proposed framework also encompasses the use of synchronous and asynchronous communication capabilities that can be used dynamically to achieve higher task concurrency.

D. Integration with reasoning framework

The integration of communication with state-of-the-art reasoning frameworks like ReAct, CoT, ToT, GoT, LLM+P, etc. can significantly enhance the reasoning capabilities of LLMs. This is particularly beneficial in the context of complex tasks, where a naive, communicate-everything approach can lead to context overflow and hallucination. By communicating strategically, the framework allows agents to focus on the relevant information and ignore the irrelevant information, thereby enhancing the efficiency of communication and improving the performance of LLMs in complex tasks.

Reasoning in a task-specific context also will lead to higher contextual reasoning, preventing context leaking, and allowing agents to request additional reasoning or context as needed. Our approach is extendable and allows for future enhancements that can integrate with state-of-the-art reasoning frameworks.

IV. Open questions and challenges

Evaluating the performance of LLMs, particularly in the context of complex tasks and agent-based communication, presents several challenges.

One of the primary challenges is the evaluation of steerability. Steerability refers to the ability of a model to generate responses that are not only relevant and coherent but also align with the specific instructions or prompts given by the user. Evaluating steerability requires assessing the model's ability to maintain a balance between following the user's instructions and generating creative and diverse responses. This is a complex task that requires a nuanced understanding of the context and the specific requirements of the task [17].

Coherence is another aspect that is difficult to evaluate. Coherence refers to the model's ability to generate responses that are logically consistent and maintain a coherent narrative throughout the conversation. Evaluating coherence requires assessing the model's ability to maintain context over long conversations and its ability to avoid generating irrelevant or contradictory information, a phenomenon known as hallucination.

Evaluating the model's ability to solve complex tasks is also challenging. Complex tasks often require the model to understand and manipulate large amounts of information, maintain a coherent narrative over a large number of tokens, and generate responses that are not only relevant and coherent but also accurate and complete. Evaluating the model's performance in these tasks requires a comprehensive assessment of its reasoning capabilities, its ability to handle large amounts of information, and its ability to generate accurate and complete responses.

Furthermore, the integration of the proposed communication framework into LLMs for agent-based communication adds another layer of complexity to the evaluation process. Evaluating the effectiveness requires assessing the efficiency of communication between agents, the reduction in the chances of miscommunication and hallucination, and the improvement in task coordination.

The evaluation of LLMs, particularly in the context of complex tasks and agent-based communication, presents several challenges. These challenges highlight the need for more sophisticated evaluation methods that can accurately assess the performance of LLMs in these complex scenarios.

V. Conclusion

The integration of a collaborative multi-agent framework with large language models (LLMs) presents a promising avenue for enhancing task performance in agent collaboration. The proposed communication framework, inspired by KQML and KIF, offers a structured approach to communication, versatility, and the ability to prevent context overflow and hallucination significantly improving the efficiency and reliability of agent communication.

When combined with reasoning frameworks like chain-of-thought, tree-of-thoughts, or other reasoning strategies, the proposed communication framework can facilitate efficient and reliable communication between agents, thereby improving task coordination in complex tasks. Furthermore, the use of a multi-agent system, designed in a master-worker agent setup allows for efficient task distribution and coordination, leading to more effective solutions to complex problems.

Despite the challenges associated with scaling out reasoning frameworks, the potential benefits of integrating KQML and KIF with LLMs are significant. As demonstrated by applications like AutoGPT and BabyAGI, agent collaboration can revolutionize the NLP landscape, and a knowledge-sharing-backed communication framework plays a crucial role in this process. Future research should continue to explore and refine this approach, with the aim of fully exploiting the potential of agent collaboration with LLMs.

VI. Acknowledgement

This work was partially supported by the National Science Foundation (NSF), Grant Number CNS-2301552.

References

[1] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, et al. *Graph of Thoughts: Solving Elaborate Problems with Large Language Models*. 2023. arXiv: 2308.09687.

[2] Daniil A. Boiko, Robert MacKnight, and Gabe Gomes. *Emergent autonomous scientific research capabilities of large language models*. 2023. arXiv: 2304.05332.

[3] Andres M Bran, Sam Cox, Andrew D White, and Philippe Schwaller. *ChemCrow: Augmenting large-language models with chemistry tools*. 2023. arXiv: 2304.05376.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[5] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, et al. *AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors in Agents*. 2023. arXiv: 2308.10848.

[6] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. "KQML as an Agent Communication Language". In: *Proceedings of the Third International Conference on Information and Knowledge Management*. CIKM '94. Gaithersburg, Maryland, USA: Association for Computing Machinery, 1994, pp. 456–463. ISBN: 0897916743. DOI: 10.1145/191246.191322. URL: https://doi.org/10.1145/191246.191322.

[7] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, et al. "PAL: Program-aided Language Models". In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 10764–10799. URL: https://proceedings.mlr.press/v202/gao23f.html.

[8] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, et al. "Reasoning with language model is planning with world model". In: *arXiv preprint arXiv:2305.14992* (2023).

[9] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. *Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents*. 2022. URL: https://proceedings.mlr.press/v162/huang22a/huang22a.pdf.

[10] Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber, et al. *MRKL Systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning*. 2022. arXiv: 2205.00445.

[11] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, et al. "Llm+ p: Empowering large language models with optimal planning proficiency". In: *arXiv preprint arXiv:2304.11477* (2023).

[12] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, et al. "AgentBench: Evaluating LLMs as Agents". In: *arXiv preprint arXiv: 2308.03688* (2023).

[13] Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, et al. "Entity-Based Knowledge Conflicts in Question Answering". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 7052–7063. URL: https://aclanthology.org/2021.emnlp-main.565.

[14] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. *Language Models of Code are Few-Shot Commonsense Learners*. 2022. arXiv: 2210.07128.

[15] Aaron Parisi, Yao Zhao, and Noah Fiedel. *TALM: Tool Augmented Language Models*. 2022. arXiv: 2205.12255.

[16] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, et al. *Toolformer: Language Models Can Teach Themselves to Use Tools*. 2023. arXiv: 2302.04761.

[17] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, et al. *HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face*. 2023. arXiv: 2303.17580.

[18] Significant-Gravitas. *Auto-GPT*. 2023. URL: https://github.com/Significant-Gravitas/Auto-GPT (visited on 09/20/2023).

[19] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971.

[20] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288.

[21] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, et al. *Self-Consistency Improves Chain of Thought Reasoning in Language Models*. 2023. arXiv: 2203.11171.

[22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903.

[23] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, et al. *Tree of Thoughts: Deliberate Problem Solving with Large Language Models*. 2023. arXiv: 2305.10601.

[24] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, et al. "ReAct: Synergizing Reasoning and Acting in Language Models". In: *International Conference on Learning Representations (ICLR)*. 2023.

[25] yoheinakajima. *babyagi*. 2023. URL: https://github.com/yoheinakajima/babyagi (visited on 09/20/2023).

[26] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, et al. *Least-to-Most Prompting Enables Complex Reasoning in Large Language Models*. 2023. arXiv: 2205.10625.