Enhancing LLM Agent Effectiveness via Reflective Multi-Agent System

Aissa Hadj Mohamed[1], **Frances A. Santos**[1], **Julio Cesar dos Reis**[1]

[1]Universidade Estadual de Campinas (UNICAMP)

Abstract. *In the last couple of years, we have observed the rapid development of agent systems, which are incorporating Large Language Models (LLMs) as their core components to perform tasks such as content generation, task planning, and conversational actions. Reflection memory, a key component of agent systems, enables LLM agents to improve their results. This study presents a novel reflective multi-agent system designed to enhance the effectiveness of LLM agents. The solution utilizes N independent agents to generate diverse responses to user prompts (questions), which are then aggregated and analyzed by a decision-making agent to produce a final answer. The reflection mechanism is triggered by user feedback, enabling self-critique agents and accumulating diverse error patterns in their respective memories. Our experimental evaluation demonstrates that our approach outperforms individual agents on the ARC Challenge dataset. Our results reveal 56.85% for our solution, compared to an average of 54.83% for single agents with reflection memory, using the small, distilled model DeepSeek with 1.5 billion parameters. This research highlights the effectiveness of a reflective multi-agent system in enhancing the overall results of LLM agents when performing tasks.*

1. Introduction

Large Language Models (LLMs) are reshaping the way people interact with online services, thanks to LLM-powered applications, commonly known as virtual assistants, such as ChatGPT [OpenAI 2024], Gemini [Team 2023], etc. LLMs can be used for a variety of tasks, including content generation (*e.g.*, text writing, music composition, and code generation), conversational capabilities (*e.g.*, personalized recommendations, question answering, and translation) and more [[Guo et al. 2024], [Zhang et al. 2024]].

LLM-based applications still present limitations that hinder their use in more complex scenarios. One key issue is hallucination [Wang et al. 2024], where an LLM generates inaccurate or entirely false information when queried about topics it has not learned during pretraining. LLMs are stateless, meaning they do not retain any memory of past interactions. Additionally, LLMs encounter challenges such as continuous decision-making, a lack of long-term memory, and limited context windows in dynamic environments [Liang et al. 2024]. A significant amount of research is focused on enhancing LLMs' ability to improve themselves. Some researchers are working on using carefully crafted prompts to help models learn how to improve, although this approach typically only works for one-off tasks. Others are tweaking how models get feedback, which allows them to get better at 'thinking things' through [Liang et al. 2024].

An LLM-based intelligent agent is a software agent that leverages LLMs as the main component of its reasoning, decision-making, and communication capabilities. The

agent uses the LLM to understand natural language inputs, produce natural language outputs, and execute tasks that require complex language-based reasoning. In this context, the LLM is an advanced linguistic processor that enables the agent to engage in human-like dialogue, provide contextualized answers, and adapt to new information. Beyond purely conversational interactions, this agent can use other modules (such as knowledge bases, external APIs, or sensors) to perform broader functions, including planning, information retrieval, and user-specific personalization. LLM agents can be equipped with memories to store information perceived from the environment and leverage the recorded memories to facilitate future actions. Research shows that memory can help the agent accumulate experiences, self-evolve, and behave more consistently, reasonably, and effectively [[Wang et al. 2024], [Kagaya et al. 2024]].

The memory module, responsible for storing, processing, and retrieving task-relevant information, enables the agent to retain historical experiences and support decision-making. Rather than retraining the LLM to acquire new information, the memory module allows the agent to accumulate knowledge dynamically, increasing adaptability while reducing computational costs [Shinn et al. 2023].

Among the various types of memory, memory reflection enables the LLM agent to learn from past experiences, thereby improving its future actions. In [Shinn et al. 2023], the authors introduced the concept of "Reflexion" as a technique that leverages 'verbal reinforcement' to help LLM-based agents learn from past mistakes. This approach is an alternative to traditional trial-and-error learning, as conventional reinforcement learning methods often require large amounts of training data and expensive model fine-tuning.

Within the Reflexion framework, an LLM agent is prompted to self-evaluate its past actions and generate outputs critically. Given a user prompt, the agent's initial response, and an environment feedback (*e.g.*, user corrections or external validation), the agent analyzes its previous answer to identify flaws in reasoning, factual inaccuracies, or logical inconsistencies. The agent then creates strategies, refines its approach when faced with similar queries to improve its future outputs. This iterative reflection process allows the agent to continuously enhance its responses' quality, accuracy, and reliability. However, integrating the memory reflection into LLM agents presents several open research challenges, as briefly summarized in the following:

- **Computational costs**: computing resources are needed when the agent is tasked to reflect on its past answers. The generated reflection must be stored in a database and retrieved when the agent processes a new user prompt [Zhang et al. 2024].
- **Memory management and organization**: the reflection memory has limited capacity. The agent needs to decide which memories to store, update, and erase to avoid memory overload and ensure efficient retrieval [[Lu et al. 2023], [Zhang et al. 2024]].
- **Subjectivity and self-assessment bias**: the quality of the self-critique depends on the LLM agent's ability to accurately assess its performed actions, which can be subjective or biased [[Gallegos et al. 2024], [Li et al. 2024a]]. The agent may favor information that confirms its existing beliefs, leading to biased reflections and hindering its ability to learn from mistakes.
- **Difficulty adapting to new circumstances**: The agent may overfit past experiences, making it difficult to generalize to new situations.

This study proposes a reflective multi-agent system designed to enhance the quality of reflection memory, thereby increasing agent effectiveness. When receiving a user prompt, our solution generates diverse answers from N agents. The answers are then concatenated with the user prompt and submitted to another agent named the decision-maker (DM) for providing a final output to the user. The user gives personalized feedback to each agent. All the agents self-critique their initial answers based on the feedback and build their reflective memories. Our multi-agent system approach leverages their reflections to generate improved answers when they receive a new prompt similar to those in their memories. The intuition is that by increasing the likelihood of initial negative experiences, we increase the probability that the multi-agent system will improve its future responses when receiving a new prompt (question) similar to the ones that generated the reflections.

We conduct an experimental evaluation considering the ARC dataset on a multi-agent system composed of three agents and a DM. Our findings demonstrate improved results compared to a single-agent approach, with and without the memory component. Our reflective multi-agent system outperforms individual agents, achieving an overall effectiveness of 56.85% compared to an average of 54.83% for single agents with reflection memory. The DM agent resolves 47.35% successfully when the agent's answers conflict.

The remaining of the article is structured as follows: Section 2 presents existing solutions from the literature. Section 3 presents our designed and implemented solution. Section 4 reports the experimental evaluation. Section 5 discusses our findings. Section 6 wraps up the conclusion remarks and points out future investigations.

2. Related Work

[Renze and Guven 2024] demonstrated the effectiveness of the reflection memory component in improving the LLM agents' results. In their experiments, agents using various types of self-reflection (eight types) outperformed the baseline agent without any self-reflection. The authors concluded that LLM agents that can self-reflect on their own mistakes based on error signals from the environment can learn to avoid similar mistakes in the future. This helps prevent agents from getting stuck in unproductive loops because they continue repeating the same error indefinitely.

According to [Zhao et al. 2024], state-of-the-art language models like GPT-4 and Claude are primarily accessible through API calls, with their parametric weights remaining proprietary and unavailable to the public. This scenario underlines the growing need for new methodologies that allow learning from agent experiences without requiring parametric updates [Zhao et al. 2024]. The authors [Zhao et al. 2024] proposed the Experiential Learning (ExpeL) agent to autonomously gather experiences and extract knowledge from a collection of training tasks. At inference, the agent recalls its extracted insights and past experiences to make informed decisions. Their empirical results highlighted the robust learning efficacy of the ExpeL agent, indicating a consistent enhancement in its performance as it accumulates experiences.

The authors [Wu et al. 2024] observed that LLMs struggle to effectively self-correct or engage in iterative reasoning when faced with complex tasks. As a solution, they proposed the Editable Large Language Model (E-LLM) to interact with generated text dynamically. E-LLM appends new tokens and inserts or deletes tokens within the

text, offering flexibility that traditional LLMs lack. This capability enables E-LLM to perform on-the-fly error correction, avoiding the constraints imposed by prior outputs. This approach offers various key advantages: it allows the model to self-correct during the generation process, reducing logical inconsistencies, it dynamically adjusts reasoning time based on task complexity, enhancing efficiency, and it eliminates the need for manually crafted reasoning chains, while still providing superior reasoning capabilities.

The authors [Liang et al. 2024] proposed a framework called Self-evolving Agents with Reflective and Memory-augmented Abilities (SAGE). SAGE comprises three agents: the User, the Assistant, and the Checker. SAGE integrates iterative feedback, reflective mechanisms, and a memory optimization mechanism based on the Ebbinghaus forgetting curve to enhance the agents' capabilities in handling multitasking and long-term information. The agents, through self-evolution, can adaptively adjust strategies, optimize information storage and transmission, and effectively reduce cognitive load. Experimental results showed that SAGE significantly improved model performance, achieving a 2.26X improvement on closed-source models and an improvement ranging from 57.7% to 100% on open-source models, with particularly notable effects on smaller models. SAGE adopts a self-reflection mechanism, demonstrating that LLMs can improve and produce high-quality work across different tasks without extra training.

In [Lippmann et al. 2024], the authors noticed that LLM agents with memory reflection tend to perform poorly when they successfully complete their tasks initially or when agents use a relatively small LLM. This is because previous self-reflection approaches focus only on unsuccessful decisions made by agents. They overlook the importance of reinforcing successful behaviors similarly. In their investigation, [Lippmann et al. 2024] proposed Sweet&Sour to leverage both positive (sweet) and negative (sour) experiences into the reflection process. This enables the agent to learn effectively from successful actions by reinforcing strategies that lead to positive outcomes while learning from failures. When the current policy achieves rewards, its approach queries the agent to extrapolate from it, encouraging the agent to verbalize what made its current policy successful and what can be generalized. In addition, Sweet&Sour uses a managed memory approach to store and retrieve relevant reflections. This is implemented using a dual-buffer structure, where experiences are stored in short-term and long-term memory based on their outcome (success or failure) and recency.

One limitation of the proposed solutions [[Liang et al. 2024], [Lippmann et al. 2024], [Renze and Guven 2024], [Wu et al. 2024]] is that they focus on a single agent with a reflection memory that improves its responses by iterating on its initial answer. During the training phase, the agent may or may not make errors when generating a response, but the correctness of that response influences the quality of its self-critique. If the agent initially provides a correct answer, the impact of reflection on future user prompts is limited.

To overcome that limitation, the authors [Li et al. 2024b] designed a solution where an LLM model generates various candidate responses based on a prompt. Then, given a set of response candidates and the prompt, the same LLM model outputs a final answer. Given an input X to the LLM model, this approach contains three steps: (1) Exploration: Given an input X, prompt LLM M to generate K candidate responses r_j, (2) Reflection: For each response r_j, prompt M with the concatenated input [X; r_j] to

generate a self-critique c_j, and (3) Revision: concatenate the K response-reflection pairs into a new input and prompt M to generate an improved output.

The solution proposed by [Li et al. 2024b] shares similarities with our reflective multi-agent system, diverging primarily in the prompt construction for the LLM agent identified as our solution's decision-maker (DM). In [Li et al. 2024b], the LLM agent receives a prompt comprising N paired elements, precisely (response candidate, agent reflection). Conversely, our proposed solution constructs the prompt for the same agent by concatenating N response candidates and the DM reflection. While seemingly incremental, this modification significantly reduces input token requirements. Consequently, our proposed system mitigates the constraints imposed by limited input token windows [[Tworkowski et al. 2023], [Zhang et al. 2024]], thus enhancing its practical applicability in real-world deployments where token efficiency is required.

Park *et al.* [Park et al. 2023] proposed a similar approach to ours by adopting a multi-agent system with reflection memory. Their generative agents simulate believable human behavior (such as waking up, cooking, and working). The agents store a complete record of their experiences using natural language, synthesize those memories over time into higher-level reflections, and retrieve them dynamically to plan behavior. In an evaluation, these generative agents produce believable individual and emergent social behaviors: for example, starting with only a single user-specified notion that one agent wants to throw a Valentine's Day party, the agents autonomously spread invitations to the party over the next two days, make new acquaintances, ask each other out on dates to the party, and coordinate to show up for the party together at the right time. Park *et al.* [Park et al. 2023] demonstrated through ablation studies that the components of their agent architecture, such as observation, planning, and reflection, each contribute critically to the believability of agent behavior.

In summary, our reflective multi-agent system, compared to existing solutions: (a) mitigates the constraints imposed by limited input token windows, as opposed to [Li et al. 2024b], (b) increases the likelihood of initial errors for future gains in effectiveness, as opposed to [[Liang et al. 2024], [Lippmann et al. 2024], [Renze and Guven 2024], [Wu et al. 2024]], and (c) comprises agents focused on the same task, as opposed to [Park et al. 2023]. The specific agents in our solution can be derived from different personalities and LLM models (some agents could use *DeepSeek* while others, for instance, use Gemini). This approach can yield more variance in agents' responses for richer negative and positive experiences.

3. Reflective Multi-Agent System

Agent reflection memory can be defined as a function of the initial user prompt, the agent's generated response, and the reflection prompt provided to the agent. This study aims to improve the quality of these reflection memories. We define quality as the extent to which a reflection memory increases the agent's effectiveness on future prompts that are semantically similar to the one that originally triggered the reflection. The reflection memory effectiveness is measured by its ability to help the agent generalize and refine its responses in similar contexts. Therefore, we define:

$$Q_{RM} = f(p_u, a, f_{env}, p_r) \tag{1}$$

where the reflection memory quality Q_{RM} is a function of (i) the user prompt p_u, (ii) the agent generated answer a, (iii) the environment feedback f_{env} to a, and (iv) the reflection prompt p_r. The primary goal is to maximize Q_{RM} to the greatest extent possible. Our approach to achieving this involves increasing the variance in the responses generated by the agents. Figure 1 presents our multi-agent system and its mechanism to output a response to a user input.

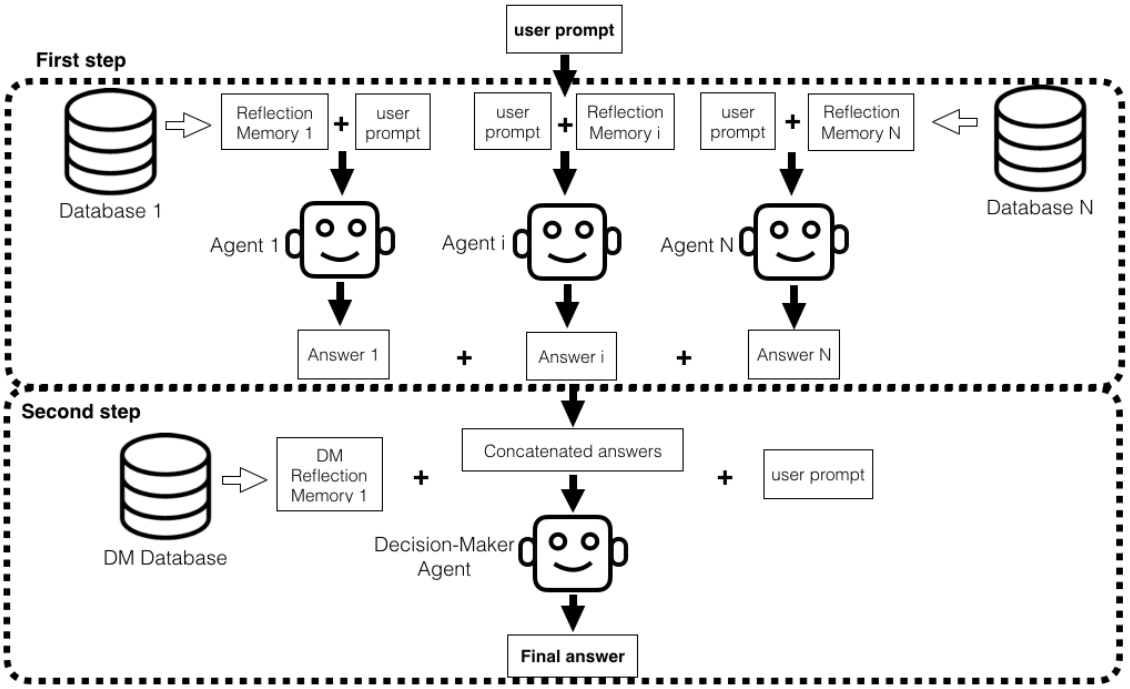

Figure 1. The multi-agent system comprises N agents answering a user prompt using their respective reflection memories. The decision-maker agent receives N concatenated answers, its reflection memory, and the user prompt, and provides the final answer to the user.

The proposed multi-agent system comprises N independent agents designed to generate diverse and refined responses to user prompts. Upon receiving a user prompt, the N agents concurrently produce N distinct outputs to maximize response diversity. We can set the LLM temperature to a high value (close to 1.0) to obtain diverse answers. These outputs, along with the original user prompt, are aggregated and submitted to the decision-maker agent (DM agent). Using its reasoning capabilities, the DM agent generates a final response, which is then presented to the user. User feedback, such as a binary correctness assessment for multiple-choice questions, is distributed to each N agent and DM agent for individual self-reflection.

The emphasis on generating diverse outputs among the N agents facilitates heterogeneous self-critique processes. Consequently, each agent's reflection memory accumulates distinct error patterns. This diversification of error experiences mitigates the likelihood of future error recurrence from the N + DM agents. By increasing the possibility of committing various types of errors initially, our solution decreases the probability that it commits errors in the future.

After receiving feedback, each agent self-reflects on its initial answer. Depending

on the task, the reflection prompt contains instructions questioning the agent whether it understood the user prompt correctly, whether its reasoning was flawed, and whether it considered every fact contained in the prompt.

4. Experimental Evaluation

This section evaluates our proposed multi-agent system and demonstrates its effectiveness in enhancing the performance of LLM agents.

4.1. Dataset and LLMs

We selected the ARC (AI2 Reasoning Challenge) dataset [1], which consists of $1,027$ multiple-choice questions requiring factual recall, scientific knowledge application, and logical inference. The "Challenge" subset contains questions that are particularly difficult for current AI models because they require abstract reasoning, understanding causal relationships, and synthesizing multiple pieces of information. To facilitate the experimental results analysis, we note that LLM models, with up to 7 billion parameters, achieve accuracy scores ranging from 40% to 50% on the ARC Challenge dataset [2].

We selected two LLM models for the experiments. The first one is DeepSeek-R1-Distill-Qwen-1.5b [3]. Our choice was motivated by the large number of requests made to the LLMs during both the training and inference phases, making lightweight models particularly advantageous in this context. We selected GPT 3.5 [4] as the second model to evaluate our multi-agent system approach using large models.

4.2. Procedures

Given the ARC sample size over $1,000$ questions, we ask the agents to provide a clear option selection as follows: "***Respond ONLY in this format: 'Answer: X', where X is the letter (A, B, C, or D) corresponding to your chosen answer. Otherwise, your response will automatically be evaluated as wrong.*". We set the number of agents to $N = 3$ given our computational constraints. This results in a total of four agents, including the DM agent. We efficiently extract the responses the agents select and automate the feedback process. If the agent does not provide an answer in the expected format, it is penalized, meaning that its answer is automatically counted as wrong. We use the accuracy score as the evaluation metric.

We ask the agents to answer the ARC questions in two rounds (also called epochs). After the first round (epoch 1), we ask the agents to reflect on their answers, whether they are correct or not. Then, in epoch 2, the agents solve the ARC questions again, with the agent prompts containing their initial answers, the feedback (whether the answer is correct or not), and their reflections. Table 1 presents the details about our experimental settings. Tables [2, 3, 6] present the accuracy scores of each agent (1 to 3) separately, and the scores of our proposed multi-agent system represented by the effectiveness of the DM agent.

The experimental results are available on the GitHub repository: https://github.com/aissahm/Multi-Agent-System.

[1] ARC Challenge dataset: https://huggingface.co/datasets/allenai/ai2_arc

[2] LLMs results on ARC: https://paperswithcode.com/sota/common-sense-reasoning-on-arc-challenge

[3] DeepSeek model: https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B

[4] ChatGPT: Conversational AI Model, https://openai.com/chatgpt

Table 1. Experimental Settings

Aspect	Details
Dataset	ARC Challenge dataset (1,027 multiple-choice science questions)
Evaluation Metric	Accuracy score (correct selection of A, B, C, or D)
Answer Format Enforcement	Responses must follow strict format: `Answer: X`; otherwise, considered incorrect (agent is penalized)
Agents per Run	3 agents + 1 decision-making (DM) agent (total of 4 agents)
Rounds (Epochs)	2 rounds: • Epoch 1: Agents answer the ARC questions • Epoch 2: Agents reflect on their answers and retry with initial answer + reflection + feedback
LLM Performance Range Benchmark	Reported LLM (with up to 7 billion parameters) accuracy on ARC Challenge ranges from 40% to 50%

4.3. Experimental Results

Figure 2 presents an increased overall effectiveness for the three single agents and the DM agent. The number of answers wrong or in non-correct format decreased from epoch 1 to 2, while the number of correct answers increased. The improvements from the three agents demonstrate the effectiveness of the reflection memory component.

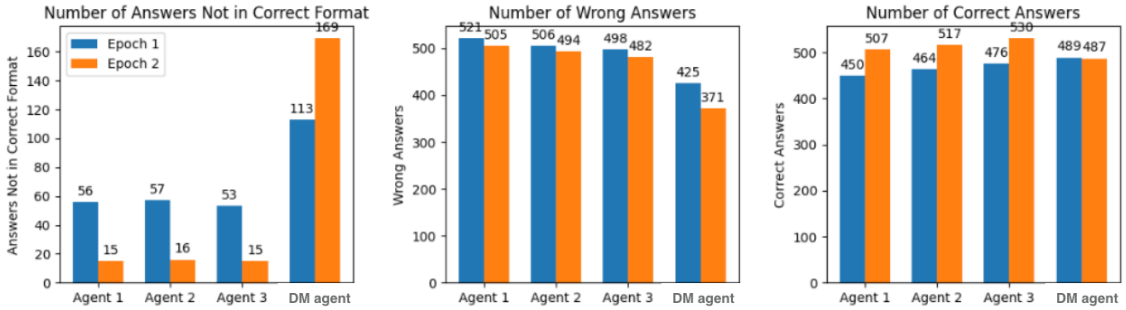

Figure 2. Comparisons of results among the three agents and the DM in epochs 1 and 2.

Table 2 presents the overall accuracy scores for each agent and the multi-agent system. The three single agents (without memory) have an average effectiveness score of 45.11%. Despite its relatively small size, DeepSeek-R1-Distill-Qwen-1.5b performs on par with larger models containing over 7 billion parameters. At epoch 1, our multi-agent system outperformed the single agents, achieving an accuracy score of 46.74% compared to 43.02%, 44.35%, and 45.50% for agents 1, 2, and 3, respectively. In epoch 2, the results of the multi-agent system slightly decreased compared to epoch 1. Its performance (46.55%) is lower than the 3 agents' (48.47%, 49.42%, and 50.66%). We noticed an increase in occurrences at epoch 2, where the DM agent fabricates information, such as asserting the presence of more than three agents. As a result, a good portion of its answers to ARC questions are counted as 'wrong', and its effectiveness decreased at epoch 2.

Table 3 presents results only considering the questions (628) where all the agents provided a valid answer (not penalized). If an agent does not provide an answer in the

Table 2. Overall Accuracy Scores (%) with $1,027$ ARC questions

Approach	Epoch 1	Epoch 2	Basis point improvement
Agent 1	43.81	49.36	+5.550
Agent 2	45.18	50.34	+5.160
Agent 3	46.34	**51.60**	+5.260
3-Agent average	45.11	50.43	+5.320
multi-agent system	**47.61**	47.41	-0.2000

expected format, we discard the question when computing the agents' accuracy scores. The agent benefits from reflection memory. For instance, agent 2 improved its accuracy score from 51.09% at epoch 1 to 54.37% at epoch 2. In addition, without memory (at epoch 1), our solution performed better than the three agents responding separately, with a score of 56.71% versus 54.84% for agent 2. At epoch 2 (with the reflection memory) our multi-agent system results kept stable at 55.78%. The three agents benefited the most from their reflection memories as they reduced the gap with the multi-agent system compared to epoch 1. However, the multi-agent system performed better than the single agents, with a highest score of 54.84% versus 55.78% for our solution.

Table 3. Overall Accuracy Scores (%) on 628 ARC questions. We discard the questions when no answer is provided in the expected format (no penalty is applied to the agents).

Approach	Epoch 1	Epoch 2	Basis point improvement
Agent 1	50.00	53.18	+3.180
Agent 2	52.07	55.41	+3.340
Agent 3	53.02	55.90	+2.880
3-Agent average	51.70	54.83	+3.130
multi-agent system	**57.80**	**56.85**	-0.9500

Table 4 presents the outcome of the DM agent when it receives conflicting answers from the three agents. In those cases, the agents selected at least two options for a given question. The DM accuracy scores range from 49.23% to 47.35% in epochs one and two. The superior scores compared to a success rate of 25% in a random choice strategy and the consistency with the overall performances (Table 3). This shows that the DM effectively uses its reasoning abilities to formulate its answers.

Table 4. Score of DM agent when there is a conflict between 3 agents (628 ARC questions)

	Epoch 1	Epoch 2
Number of conflicting questions	192	188
% Correct Answers	49.23%	47.35%

Table 5 presents some inadequacies in our results. Sometimes, the LLM agents respond incorrectly to questions in epoch 2 that they answered correctly in epoch 1. One possible explanation is that the agents are lost in the information provided in the prompts, suggesting that the DeepSeek-R1-Distill-Qwen-1.5B model struggles when pre-

sented with excessive details in a prompt. If we correct those inadequacies, the accuracy scores in Table 2 would be about 7 basis points higher in epoch 2.

Table 5. Number of Correct Answers at Epoch 1 then Wrong at Epoch 2

Approach	Number of questions
Agent 1	71
Agent 2	71
Agent 3	74
multi-agent system	75

Table 6 presents experimental evaluations using the GPT 3.5 model on ARC Challenge Test questions. Despite setting the temperature to 1 for the LLM model responses, the single agents' performances at epoch 1 are all above 96% and consistent. The three agents provided the same options, albeit with slightly different wording in their outputs. This finding suggests that our proposed solution does not benefit from this context. Our solution leverages diversity in agents' responses to generate diversity in memory reflections.

Table 6. Scores for the reflective multi-agent system using GPT 3.5.

	Epoch 1	Epoch 2
Agent 1	96.86%	99.30%
Agent 2	96.95%	99.13%
Agent 3	96.95%	98.95%
multi-agent system	96.86%	97.73%

5. Discussion

Our experimental results suggested that the multi-agent system approach with the DeepSeek-R1-Distill-Qwen-1.5B model yields the following: (1) multiple agents, each with their reflection-memory component; and (2) a DM agent without reflection to provide a final answer to the initial prompts.

The experimental evaluation of the proposed system yielded three primary findings. First, the LLM agents enhanced their outputs by using reflection memory. Specifically, a significant improvement in accuracy was observed between the first and second epochs, indicating effective learning from prior errors. This is consistent with previous results from [Renze and Guven 2024].

Second, before memory utilization, the multi-agent system exhibited superior effectiveness at the initial epoch compared to individual agents responding to the ARC questions. This result, from the decision-maker agent (DM) compared to the three agents, demonstrates that it can successfully resolve conflicts regarding the agents' responses.

We found that whereas the reflective multi-agent system outperformed individual reflective agents (cf. Table 3), the DM agent exhibited a higher propensity for hallucinations compared to the agents (Table 2). This observation suggests a potential trade-off between the aggregation of diverse responses and the increased risk of hallucinatory outputs within the DM agent.

Our approach relied on environments or users providing feedback to the agent's outputs. We plan to explore alternative models to provide feedback for reflective learning. We do not implement a memory management mechanism, which can lead to potential redundancy when multiple agents make the same mistake. Future work will explore strategies to mitigate this redundancy and assess whether redundant agents can be removed to reduce latency.

6. Conclusion

This investigation proposed a reflective multi-agent system to enhance the effectiveness of LLM agents by improving the quality of their reflection memories. Quality was defined as the extent to which a reflection memory helps agents to answer better on semantically similar prompts. Our multi-agent approach increased response diversity, resulting in a broader range of errors. Through self-reflection, agents generated memories based on positive and negative experiences, allowing them to refine future responses. By increasing response variance, we improved the variance in self-reflections, ultimately reducing the likelihood of repeated mistakes. This addressed a key limitation in single-agent reflection approaches, where the number of initial errors is lower. Our evaluation used two LLMs on the ARC dataset due to computational constraints. Future studies aim to validate our approach across multiple LLMs and extend it to other domains, such as coding tasks, to assess its robustness further.

Acknowledgments

This study was sponsored by Petróleo Brasileiro S.A. (PETROBRAS) within the project "*Application of Large Language Models (LLMs) for online monitoring of industrial processes*" conducted in partnership with the University of Campinas.

References

[Gallegos et al. 2024] Gallegos, I. O., Rossi, R. A., Barrow, J., Tanjim, M. M., Kim, S., Dernoncourt, F., Yu, T., Zhang, R., and Ahmed, N. K. (2024). Bias and fairness in large language models: A survey. *arXiv preprint arXiv:2309.00770*.

[Guo et al. 2024] Guo, J., Wang, M., Yin, H., Song, B., Chi, Y., Yu, F. R., and Yuen, C. (2024). Large language models and artificial intelligence generated content technologies meet communication networks. *arXiv preprint arXiv:2411.06193*.

[Kagaya et al. 2024] Kagaya, T., Yuan, T. J., Lou, Y., Karlekar, J., Pranata, S., Kinose, A., Oguri, K., Wick, F., and You, Y. (2024). Rap: Retrieval-augmented planning with contextual memory for multimodal llm agents. *arXiv preprint arXiv:2402.03610*.

[Li et al. 2024a] Li, Y., Du, M., Song, R., Wang, X., and Wang, Y. (2024a). A survey on fairness in large language models. *arXiv preprint arXiv:2308.10149*.

[Li et al. 2024b] Li, Y., Yang, C., and Ettinger, A. (2024b). When hindsight is not 20/20: Testing limits on reflective thinking in large language models. *arXiv preprint arXiv:2404.09129*.

[Liang et al. 2024] Liang, X., Tao, M., Xia, Y., Shi, T., Wang, J., and Yang, J. (2024). Self-evolving agents with reflective and memory-augmented abilities. *arXiv preprint arXiv:2409.00872*.

[Lippmann et al. 2024] Lippmann, P., Spaan, M. T. J., and Yang, J. (2024). Positive experience reflection for agents in interactive text environments. *arXiv preprint arXiv:2411.02223*.

[Lu et al. 2023] Lu, J., An, S., Lin, M., Pergola, G., He, Y., Yin, D., Sun, X., and Wu, Y. (2023). Memochat: Tuning llms to use memos for consistent long-range open-domain conversation. *arXiv preprint arXiv:2308.08239*.

[OpenAI 2024] OpenAI (2024). Chatgpt: Conversational ai model. [Online]. Available from: `https://openai.com/chatgpt`.

[Park et al. 2023] Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior.

[Renze and Guven 2024] Renze, M. and Guven, E. (2024). Self-reflection in llm agents: Effects on problem-solving performance. *arXiv preprint arXiv:2405.06682*.

[Shinn et al. 2023] Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

[Team 2023] Team, G. (2023). Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

[Tworkowski et al. 2023] Tworkowski, S., Staniszewski, K., Pacek, M., Wu, Y., Michalewski, H., and Miłoś, P. (2023). Focused transformer: Contrastive training for context scaling. *arXiv preprint arXiv:2307.03170*.

[Wang et al. 2024] Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., and Wen, J. (2024). A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6).

[Wu et al. 2024] Wu, Y., Xu, G., and Dongchen, Z. (2024). proposalself-reflection like humans, editable-LLM (e-LLM) is all you need. [Online]. Available from: `https://openreview.net/forum?id=rk6PJlu562`. under review.

[Zhang et al. 2024] Zhang, Z., Bo, X., Ma, C., Li, R., Chen, X., Dai, Q., Zhu, J., Dong, Z., and Wen, J.-R. (2024). A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*.

[Zhao et al. 2024] Zhao, A., Huang, D., Xu, Q., Lin, M., Liu, Y.-J., and Huang, G. (2024). Expel: Llm agents are experiential learners. *arXiv preprint arXiv:2308.10144*.