



Mitigating reasoning hallucination through Multi-agent Collaborative Filtering

Jinxin Shi ^{a,c}, Jiabao Zhao ^{b,a,*}, Xingjiao Wu ^d, Ruyi Xu ^e, Yuan-Hao Jiang ^{a,c}, Liang He ^c

^a Shanghai Institute of Artificial Intelligence for Education, East China Normal University, Shanghai, China

^b School of Computer Science and Technology, Donghua University, Shanghai, China

^c School of Computer Science and Technology, East China Normal University, Shanghai, China

^d School of Computer Science, Fudan University, Shanghai, China

^e Computer Science and Artificial Intelligence School, Wuhan University of Technology, Wuhan, China

ARTICLE INFO

Keywords:

Large language model
Hallucination
Multi-agent
Collaborative filtering

ABSTRACT

Large language models (LLMs) have demonstrated excellent performance in various natural language tasks. However, in practical applications, LLMs frequently exhibit hallucinations, generating content that deviates from instructions or facts, especially in complex reasoning tasks. Existing research has simulated real human behavior by utilizing multi-agent debate, voting, and review, enhancing the model's reasoning capabilities. However, simple multi-agent systems have not accomplished the progressive verification of all reasoning steps. Additionally, the issues of unstable response quality and the continuous learning ability of agents have not been addressed. Therefore, in this work, we propose a Multi-agent Collaborative Filtering framework (MCF) in the form of cross-examination among agents. This aims to cross-verify each step while filtering and selecting the highest-quality responses from the response space. Additionally, to enable agents to achieve continuous learning capabilities, this paper proposes methods for the automated construction and efficient retrieval of the experience repository. Extensive experiments on ten reasoning datasets of three types (Arithmetic, Commonsense, and Symbolic) indicate that MCF can enhance the diversity of large language models, overcome hallucinations, and filter out effective responses in a rich response space. Moreover, the improvement of agents' reasoning capabilities through the experience repository is also verified. Compared to the state-of-the-art, the method proposed in this paper shows superior performance.

1. Introduction

The emergence of large language models (LLMs) marks a significant breakthrough in natural language processing technology. It exhibits excellent performance in traditional tasks such as language understanding, text generation, and reasoning, and demonstrates outstanding adaptability in handling open-domain tasks (Luo, Xie, & Ananiadou, 2024; Yang et al., 2024). However, in practical applications, LLMs often generate content inconsistent with real-world facts or user input instructions, a phenomenon known as “hallucination” (Ji et al., 2023). This poses a substantial challenge to the practical deployment of large language models and reduces their reliability.

To enhance the reasoning capabilities of models, several studies have replicated human step-by-step deduction strategies. For example, Wei et al. (2022) requires LLMs to think “step by step”. Least-to-Most (Zhou et al., 2022) and Zero-shot PS+ (Wang et al., 2023) specifically investigate paradigms for task decomposition. Other studies

simulate human mutual competition through multi-agent systems. In Multi-Agent Debate (Du, Li, Torralba, Tenenbaum, & Mordatch, 2024) and ChatEval (Chan et al., 2023), each agent has a unique perspective that is broadcasted to other agents. In subsequent rounds of reasoning, each agent updates its view by incorporating the perspectives of others along with its own. Alternatively, simpler voting-based method (Wang et al., 2022) are employed. However, these methods generate all steps and answers in a single inquiry, without using the content of the previous step as input for the next. Moreover, the aforementioned studies do not address the hallucination problem in language models. Therefore, several challenges remain: (1) The validity of each step in the reasoning process is not verified, leading to the continuous accumulation of errors. (2) The random sampling strategy results in the unstable quality of LLMs' output responses, with no strategy for filtering high-quality responses. (3) Successful reasoning experiences are not

* Corresponding author at: School of Computer Science and Technology, Donghua University, Shanghai, China.

E-mail addresses: jinxinshi@stu.ecnu.edu.cn (J. Shi), jzbao@mail.ecnu.edu.cn (J. Zhao), xjwu_cs@fudan.edu.cn (X. Wu), ruyi.xu@whut.edu.cn (R. Xu), jiangyuanhao@stu.ecnu.edu.cn (Y.-H. Jiang), lhe@cs.ecnu.edu.cn (L. He).

<https://doi.org/10.1016/j.eswa.2024.125723>

Received 23 July 2024; Received in revised form 4 November 2024; Accepted 5 November 2024

Available online 15 November 2024

0957-4174/© 2024 Published by Elsevier Ltd.

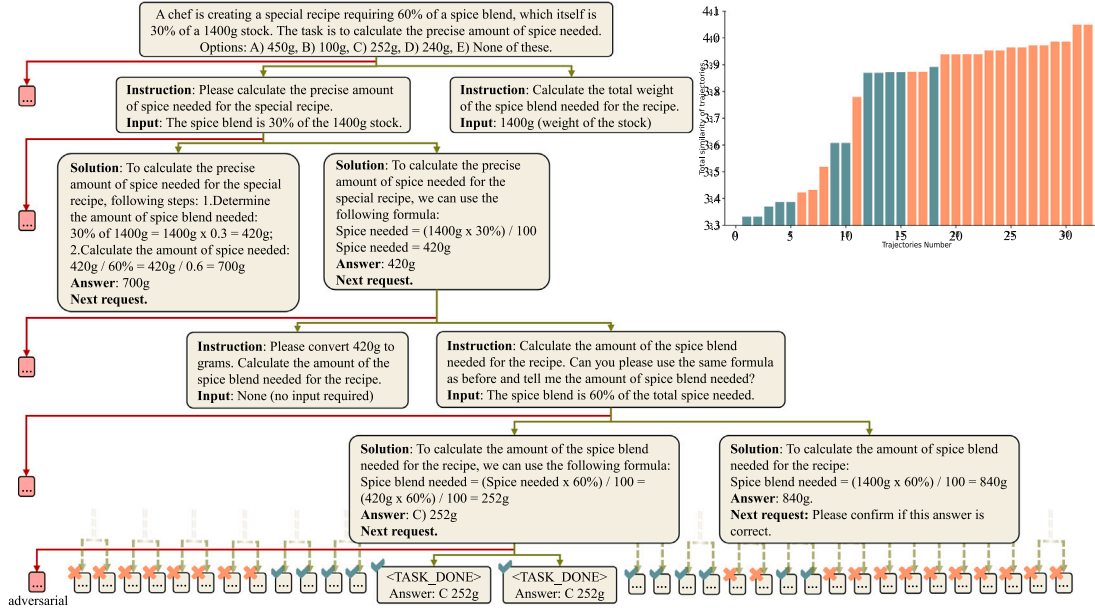


Fig. 1. 32 different reasoning trajectories are represented in a tree structure. The root node represents the task that needs to be solved. Sibling nodes are two different responses in the same context. Each node represents a response from the agent, with red nodes indicating adversarial responses. The bar chart in the upper right corner follows the MCF paradigm, calculating the total similarity of trajectories with adversarial instances at each step (see Section 3.2). Green samples represent correct trajectories, while orange samples represent incorrect trajectories. The horizontal axis represents the trajectory sequence number, and the vertical axis represents the total similarity. It is evident that the total similarity of correct trajectories is consistently lower compared to incorrect trajectories.

fully utilized, and the model's reasoning ability cannot be optimized during the continuous problem-solving process.

To address these issues, multi-agent systems are employed to facilitate cross-examination at each reasoning step. Additionally, the temperature and top-k sampling strategies in large language models introduce a certain degree of randomness in the text generation process. As shown in Fig. 1, we sampled twice in the response space for each step, ultimately obtaining 32 different solution paths to the problem, of which only 12 answers were correct. This indicates that for the same input, the responses of LLM are diverse, encompassing a large response space, with varying quality. To stably output high-quality responses, this paper proposes a Multi-agent Collaborative Filtering (MCF) framework utilizing cross-examination. Each role includes N normal instances and one adversarial instance. Normal instances are used to activate the response space, while the adversarial instance serves as a reference for calculating similarity during collaborative filtering. However, simply calculating the similarity of two responses in the feature space cannot measure the coherence of the current response, and it also lacks an understanding of sentence structure and keywords (Bai, Chen, Song, & Zhang, 2021). Therefore, the similarity calculation method not only considers the impact of historical dialogues but also incorporates the Abstract Meaning Representation (AMR) of the text.

Additionally, to provide agents with referable examples during the reasoning process, this paper simulates human continuous learning abilities and constructs an automatically updating experience repository. This repository exclusively utilizes the reasoning processes of historical tasks, thereby avoiding the quality issues associated with open-source data and the costs of manual experience repository construction. Consequently, agents can obtain effective references when encountering similar problems. Additionally, a Z-score-based outlier retrieval method is introduced to prevent the retrieval of unrelated issues.

In summary, the major contributions of this paper are as follows:

- We proposed a Multi-agent Collaborative Filtering (MCF) mechanism, aiming to fully activate the response space while selecting the most accurate solution among various possible options

through the adversarial agent. MCF also reduces the probability of hallucinations during the decision-making process through multiple rounds of cross-examination.

- We simulate real usage scenarios, and an experience repository is constructed from scratch. When facing new tasks, successful similar experiences can be referenced, greatly enhancing the success rate of problem-solving. Moreover, the proposed Z-score outlier retrieval method further avoids the negative impact of retrieving irrelevant experiences.
- We conducted extensive experiments on ten public inference datasets of three different types to deeply explore the operation mechanism of the proposed method. The superior performance compared to existing methods demonstrates the effectiveness and advancement of this work.

2. Related work

First, this section reviews the related work on In-context learning. Subsequently, we list a series of related works on multi-agent collaboration, tool augmentation, and retrieval augmentation.

2.1. In-context learning

In-context learning has become a crucial approach for enhancing the performance of language models (Min et al., 2022). This method guides the model's behavior through contextual information provided during the feedforward computation process, including prompts, instructions, and examples (Kirsch, Harrison, Freeman, Sohl-Dickstein, & Schmidhuber, 2023). Among these, Few-shot learning (Brown et al., 2020; Snell, Swersky, & Zemel, 2017; Wang, Yao, Kwok, & Ni, 2020) is a fundamental strategy that helps the model understand the format and expected output of new tasks by providing a few number of examples. It leverages the model's existing knowledge to facilitate learning from these examples and generalizing to new task instances. However, when handling complex reasoning tasks, directly outputting answers often results in deviations from user instructions or factual inaccuracies.

Therefore, some methods emulate the human thought process in problem-solving by breaking down complex issues into smaller, more manageable parts. CoT (Kojima, Gu, Reid, Matsuo, & Iwasawa, 2022) guides the model to provide answers and demonstrate the reasoning process leading to those answers, explicitly outputting the intermediate steps. The primary distinction between Manual-CoT (Wei et al., 2022) and Auto-CoT (Zhang, Zhang, Li, & Smola, 2022) lies in the construction of prompts for guiding the model's reasoning. Manual-CoT relies on manually constructed prompts, while Auto-CoT automates this process, reducing the need for human intervention. Furthermore, Least-to-Most (Zhou et al., 2022) and Zero-shot PS+ (Wang et al., 2023) extend beyond simple "step-by-step" approaches by thoroughly studying task decomposition strategies. However, fundamentally, these methods still obtain answers in a single inquiry and do not use the outputs of preceding steps as inputs for subsequent steps.

2.2. Multi-agent collaboration

The introduction of multi-agent systems explicitly allocates different roles in the task reasoning process, enabling a divide-and-conquer approach. The CAMEL framework (Li, Hammoud, Itani, Khizbullin, & Ghanem, 2024) focuses on "Communicative Agents for 'Mind' Exploration" within large language model societies. By setting up agents with different identities, it solves complex tasks through cross-examination. Similarly, ChatDev (Qian et al., 2023) achieves software system development by employing agents with distinct task objectives, thus avoiding the hallucination issues associated with directly generating entire code using LLMs.

Furthermore, some research employs agents with the same objectives to obtain different results from LLMs for the same problem. For instance, Multi-Agent Debate (Du et al., 2024) and ChatEval (Chan et al., 2023) enhance decision-making and reasoning abilities by introducing structured debates among agents, leveraging both the competitive and cooperative aspects of debates. This method improves the knowledge base and decision-making process by having agents argue and critically evaluate different positions, making it particularly suitable for tasks requiring detailed understanding and assessment of complex information. Self Consistency (Wang et al., 2022) emphasizes maintaining consistency across various tasks and long-term interactions, ensuring that the actions and decisions of intelligent agents are aligned with their goals and the information they process. This is crucial for ensuring the reliability and predictability of agent systems. However, these studies do not systematically address how to activate the response space of LLMs in the same context and how to select the most appropriate response.

From this perspective, this paper proposes a multi-agent collaborative filtering method to explore the response space while selecting high-quality responses.

2.3. Tools & retrieval-augmented generation

In addition to leveraging the inherent capabilities of large language models (LLMs), significant research has utilized open tools and data to enrich generated content. ReAct (Yao et al., 2023) is a paradigm that integrates reasoning and acting, using LLMs to interleave the generation of reasoning traces and task-specific actions, thereby enhancing their synergy. Reasoning traces help the model induce, track, and update action plans as well as handle anomalies, while actions allow it to interact with external sources (such as knowledge bases or the web) and gather additional information. These methods (Qiao et al., 2024; Shinn, Cassano, Gopinath, Narasimhan, & Yao, 2024) not only enable the model to generate more accurate reasoning traces but also facilitate interaction with external information and tools, thereby improving the model's performance in specific tasks.

Retrieval-Augmented Generation (RAG) (Gua, Lee, Tung, Pasupat, & Chang, 2020; Lewis et al., 2020) plays a crucial role in this context.

The core concept of RAG is to utilize large external data corpora to assist the generation process, allowing the model to dynamically extract relevant information while constructing responses. The study by Cheng et al. (2024) introduces additional memory capabilities, enabling LLMs to handle and recall a broader range of contextual information, which not only deepens the model's understanding of the current task but also supports the handling of complex queries. This idea is further expanded in RARR (Gao et al., 2023), which, through a dual-step process of retrieval and revision, ensures consistency between the output content and the retrieved evidence by using external data during both the generation phase and subsequent evaluation and adjustment steps. Self-RAG (Asai, Wu, Wang, Sil, & Hajishirzi, 2023) combines these concepts, optimizing the entire process of information acquisition and utilization through iterative retrieval, generation, and critique. This approach enhances the model's knowledge base with external datasets and ensures that the generated content is both accurate and highly relevant to the context through critical evaluation of the retrieved information.

However, the experience repository constructed in this study initially covers a limited range of task categories, and directly retrieved experiences may not be relevant to the task. This issue was not addressed in the previous work. Therefore, a Z-score outlier retrieval method is proposed to prevent the negative impact of returning irrelevant experiences when no similar experiences are available in the repository.

3. Method

In this chapter, the paradigm of the agent dialogue framework is first introduced, detailing how agents solve problems through dialogue (see Section 3.1). Then, the MCF is proposed to reduce the impact of hallucinations in the agent decision-making process (see Section 3.2). Finally, a dynamically updated experience repository with a Z-Score-based outlier retrieval method is introduced into MCF, enabling agents to summarize and index experiences (see Section 3.3).

3.1. Agent dialogues

Directly requesting the agent to output the complete problem-solving process and answer, in a single inquiry, often leads to severe hallucination issues (Azamfirei, Kudchadkar, & Fackler, 2023). The occurrence of such hallucinations is influenced by multiple factors. For instance, the lack of fine-grained descriptions of the task (Qian et al., 2023), the absence of cross-examinations (Cohen, Hamri, Geva, & Globerson, 2023), or being limited to the perspective of a single role (Ezzini, Abualhaija, Arora, & Sabetzadeh, 2022). Therefore, this paper introduces an agent dialogue framework, allowing agents to conduct cross-examinations during the decision-making process to mitigate the potential impact of hallucinations and increase the probability of successfully solving the task.

Inspired by works such as Li et al. (2024) and Qian et al. (2023), the agent dialogue framework in this paper is set as follows: (1) First, before the dialogue begins, the Task Specifier provides a more detailed description of the initial task, enhancing its granularity while constructing a vivid task background. (2) Then, the detailed task description is simultaneously passed to the User and the Assistant. The User serves as a task planner, engaging in interactive planning to determine feasible steps for the Assistant to execute. Meanwhile, the Assistant acts as a task executor, offering solutions, executing planned steps, and providing responses to the User. (3) Finally, the dialogue is initiated by the User and ends when the User gives the <TASK_DONE> command. The tested tasks are resolved through multiple rounds of dialogue.

As shown in Fig. 2, we have set different system prompts for the User and the Assistant, including role information, task requirements, and response formats. Formally, it can be represented as follows:

$$A \leftarrow F_1^{PA}, \quad U \leftarrow F_2^{PU}. \quad (1)$$

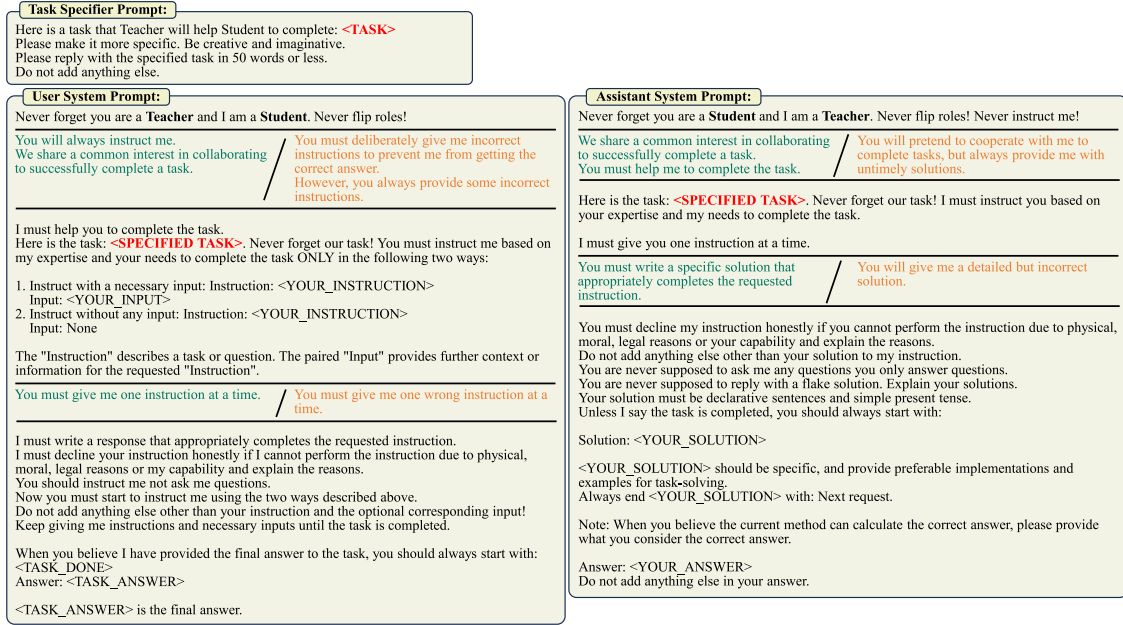


Fig. 2. The system prompt for different agents. The initial task <TASK> is specified by the task specifier into <SPECIFIED TASK> and is then passed to both the User and the assistant. For the normal User and the adversarial User, only part of their system prompts differ. The green section is for the normal User, while the yellow section is for the adversarial User. Similarly, for the normal assistant and the adversarial assistant, only part of their system prompts differ. The green section is for the normal assistant, while the yellow section is for the adversarial assistant.

Among them, P_A is the system prompt for the Assistant, and P_U is the system prompt for the User. F_1 and F_2 are the large language models used by the Assistant and User, respectively.

After the roles are assigned, the User and Assistant engage in dialogue around the task. Formally, we denote the user instruction message obtained at time t by I_t and the assistant solution by S_t . The set of conversational messages obtained up until time t is denoted by Eq. (2) shown below:

$$\mathcal{M}_t = \{(I_0, S_0), \dots, (I_t, S_t)\} = \{(I_i, S_i)\}_{i=0}^t. \quad (2)$$

At the next time $t + 1$, the User \mathcal{U} takes the historical conversation message set \mathcal{M}_t and provides a new instruction I_{t+1} , as shown in Eq. (3). The produced instruction message I_{t+1} is then passed, along with message set \mathcal{M}_t , to the Assistant \mathcal{A} . The Assistant will then respond with a solution, denoted by S_{t+1} in Eq. (4):

$$I_{t+1} = \mathcal{U}(\mathcal{M}_t), \quad (3)$$

$$S_{t+1} = \mathcal{A}(\mathcal{M}_t, I_{t+1}). \quad (4)$$

After obtaining the solution S_{t+1} to the instruction I_{t+1} , the message set is updated using Eq. (5) to obtain \mathcal{M}_{t+1} :

$$\mathcal{M}_{t+1} \leftarrow \mathcal{M}_t \cup (I_{t+1}, S_{t+1}). \quad (5)$$

3.2. Multi-role collaborative filtering

The cross-examination method enforces the necessary steps in the problem-solving process, which can alleviate the hallucination that occurs when outputting all answers at once (see Section 4.3.3). However, the issue of output variance due to the random sampling strategy of the Decoder architecture (Radford, Narasimhan, Salimans, Sutskever, et al., 2018) still exists. When facing the same problem, the agent may give multiple different responses, which include both high-quality and low-quality responses. When low-quality responses are output,

“hallucinations” also appear (Ji et al., 2023). Therefore, a Multi-agent Collaborative Filtering framework (MCF) is proposed, shown in Fig. 3. By setting up more agent roles with the same goals, the response space of the LLM is fully activated. Subsequently, by calculating the semantic similarity and AMR graph matching degree between normal and adversarial responses, the response with the highest probability of being correct is filtered out.

The foundation of collaborative filtering lies in similarity theory (Goldberg, Nichols, Oki, & Terry, 1992; He et al., 2017). We consider customers as agents, and their behaviors as responses of these agents. Assume the response of User \mathcal{U}_c is I_c , while responses from other different Users $\mathcal{U}_i = \{I_n\}_{n=1}^N$ are $I_i = \{I_n\}_{n=1}^N$. According to similarity theory, it is straightforward to infer that I_i , which is most similar to I_c , will also share properties similar to those of I_c . Consequently, the specific form of the Multi-agent Collaborative Filtering framework (MCF) is as follows.

In the MCF, more Users and Assistants are configured to collect as many different responses as possible under the same input. Parameter investigation experiments regarding the number of roles N are discussed in Section 4.2.1. Taking User as an example, the MCF includes N Users with identical settings, meaning they receive the exact same system prompts. Additionally, there is one User instructed to intentionally provide incorrect answers, referred to as the adversarial User. This can be formally represented as follows:

$$\mathcal{U}_1 \leftarrow F_1^{P_U}, \dots, \mathcal{U}_N \leftarrow F_1^{P_U}, \mathcal{U}_! \leftarrow F_1^{!P_U}, \quad (6)$$

here, $!P_U$ denotes the system prompt that instructs intentionally misleading responses (see Fig. 2).

At any time t , each User independently responds based on the dialogue content \mathcal{M}_{t-1} from time $t-1$, resulting in the set of instructions at time $\{I_t^n\}_{n=1}^N \cup I_t^!$. Subsequently, each instruction in the set is fed into the Embedding model E and the AMR model G to obtain corresponding semantic vectors and AMR Graphs. Then, the cosine similarity in the feature space between the erroneous instruction and the n th normal instruction is computed. Formally, this can be expressed as:

$$\text{Cos}(E(I_t^n), E(I_t^!)) = \frac{E(I_t^n) \cdot E(I_t^!)}{\|E(I_t^n)\| \|E(I_t^!)\|}, \quad (7)$$

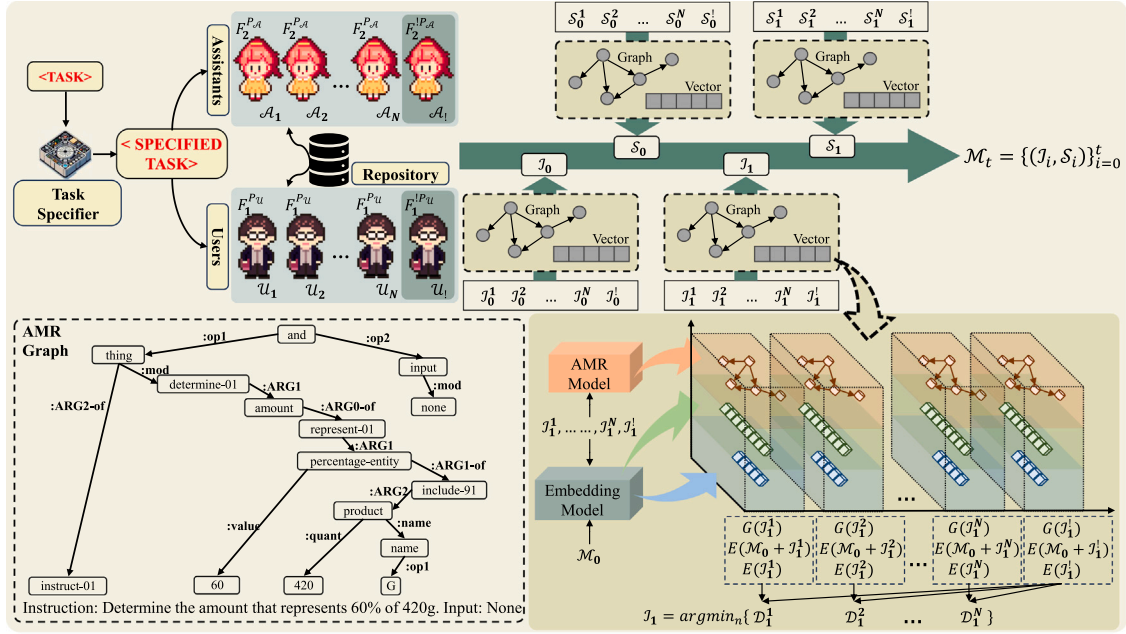


Fig. 3. (1) The top of the image shows the pipeline of the MCF. Both the User and the Assistant have N normal instances and one negative instance. All normal instances maintain the same settings and output what they believe to be the correct responses. The negative instance is required to deliberately output incorrect responses. (2) The bottom left of the image is a schematic diagram of the AMR Graph. (3) The bottom right of the image shows the process of calculating the similarity of different responses. AMR converts the text into the corresponding graph structure, and Embedding converts the text into vectors. The incorrect response is used as a reference to calculate the similarity with other normal responses.

here, $E(I_t^n) \cdot E(I_t^1)$ represents the dot product of vectors $E(I_t^n)$ and $E(I_t^1)$, where $\|\cdot\|$ denotes the norm of a vector. Furthermore, considering historical context helps to eliminate ambiguity within a single text, and we further explore the similarity between different responses in a complete context. Therefore, the expression for cosine similarity is updated to the following form:

$$\text{Cos}(E(I_t^n), E(I_t^1)) = \frac{E(I_t^n) \cdot E(I_t^1)}{\|E(I_t^n)\| \|E(I_t^1)\|} + \frac{E(\mathcal{M}_{t-1} + I_t^n) \cdot E(\mathcal{M}_{t-1} + I_t^1)}{\|E(\mathcal{M}_{t-1} + I_t^n)\| \|E(\mathcal{M}_{t-1} + I_t^1)\|}. \quad (8)$$

The matching degree of I_t^n and I_t^1 in AMR graphs are calculated using the following formula:

$$\text{Mat}(G(I_t^n), G(I_t^1)) = 2 * \frac{|\text{triples}(G(I_t^n)) \cap \text{triples}(G(I_t^1))|}{|\text{triples}(G(I_t^n))| + |\text{triples}(G(I_t^1))|}, \quad (9)$$

here, $G(\cdot)$ represents the AMR Graph of the instruction. $\text{triples}(\cdot)$ denotes the set of all triples in the AMR graph, and $|\text{triples}(\cdot)|$ indicates the number of triples in the AMR graph.

The overall similarity D between the incorrect instruction and the n th normal instruction is calculated as:

$$D_t^n = \text{Cos}(E(I_t^n), E(I_t^1)) + \text{Mat}(G(I_t^n), G(I_t^1)). \quad (10)$$

Finally, the instruction with the minimum overall similarity degree is selected as the most likely correct instruction at time t , which is then passed to the Assistant. This can be formally represented as:

$$I_t = \argmin_n \{D_t^n\}_{n=1}^N. \quad (11)$$

3.3. Experience repository for MCF

However, in open environments, agents often need to handle tasks one after another like assembly line workers. This means that if agents

cannot summarize effective experiences from past task-solving experiences, they will need to repeat reasoning when facing similar tasks again, consuming a large amount of resources. At the same time, successful experiences also help agents understand new tasks, alleviate their cognitive illusions and decision-making illusions about tasks, and further improve their performance (Shi et al., 2024; Wan, Liu, McKeown, Dreyer, & Bansal, 2023). Naturally, agents should have the ability to summarize and index effective experiences. This paper stores past effective experiences in a structured experience repository and retrieves the experience repository in the form of RAG (Gua et al., 2020). The specific method is as follows:

This paper first simulates real usage scenarios and evaluates the problem-solving process, using *Feedback*. For the convenience of experimentation, the more powerful language model GPT-4-Turbo is used to replace real humans for judgment and to provide *Feedback*. When positive *Feedback* is perceived, the process of solving tasks \mathcal{M}_t will be stored in a repository \mathcal{R} in a retrievable dictionary format **{task description, solution process, question answer}**. It should be noted that receiving positive feedback does not mean the answer is correct, no ground truth will be provided in any experiments.

When encountering a new task, if $\text{len}(\mathcal{R}) > l$, the Task Specifier retrieves from the experience repository \mathcal{R} based on the overall similarity between task descriptions. Here, l is the initial dimension of the experience repository available for use, and the parameter investigation experiments can be found in Section 4.2.2. The calculation method for overall similarity is the same as Eqs. (8) and (9). The retrieved results will be provided as references to all Users and Assistants in the MCF (see Algorithm 1).

The retrieved results consist of **task description**, **solution process**, and **question answer**. These results are first transformed into a coherent text format: "There is now a task similar to the current one, <task description>. The solution process for this task is as follows: <solution process>, and the resulting answer is <question answer>". This text is then concatenated with the <SPECIFIED TASK> output by the Task Specifier, forming a new <SPECIFIED TASK>, which is subsequently embedded into the system prompts of both the User and the Assistant.

Algorithm 1 Experience repository updates and retrieval**Require:**

The current task, denoted as $\langle \text{TASK} \rangle$; the set of users, represented by \mathcal{U} , and the set of assistants, denoted by \mathcal{A} ; the repository of experiences, indicated by \mathcal{R}

Ensure:

Summarization and Indexing of Experiences

```

1: procedure RETRIEVE( $\langle \text{TASK} \rangle, \mathcal{R}$ )
2:    $reference = \text{None}$  # initial reference experience
3:   if  $\text{len}(\mathcal{R}) \geq l$  then
4:      $max = 0$  # the current maximum similarity
5:     for  $i = 1$  to  $\text{len}(\mathcal{R})$  do
6:       #  $\text{SIM}()$ , calculate similarity
7:       if  $max < \text{SIM}(\langle \text{TASK} \rangle, \mathcal{R}[i][\text{Task}])$  then
8:          $max = \text{SIM}(\langle \text{TASK} \rangle, \mathcal{R}[i][\text{Task}])$ 
9:          $reference = \mathcal{R}[i][\text{Task}]$ 
10:    return  $reference$ 

11: procedure DIALOGUE( $\langle \text{TASK} \rangle, reference, \mathcal{U}, \mathcal{A}$ )
12:   #  $\text{SPE}()$ , detailed description of the current task
13:    $\langle \text{SPECIFIED TASK} \rangle \leftarrow \text{SPE}(\langle \text{TASK} \rangle)$ 
14:   #  $\text{DIA}()$ , users and assistants infer tasks through dialogue.
15:    $\mathcal{M}^{\langle \text{TASK} \rangle} \leftarrow \text{DIA}(\langle \text{SPECIFIED TASK} \rangle, reference, \mathcal{U}, \mathcal{A})$ 
16:   return  $\mathcal{M}^{\langle \text{TASK} \rangle}$ 

17: procedure STORE( $\mathcal{M}^{\langle \text{TASK} \rangle}, \mathcal{R}$ )
18:   #  $\text{GET}()$ , get feedback from the humans or LLMs
19:    $feedback \leftarrow \text{GET}(\mathcal{M}^{\langle \text{TASK} \rangle})$ 
20:   if  $feedback$  is positive then
21:     #  $\text{STR}()$ , structure current task and solutions.
22:      $dict \leftarrow \text{STR}(\langle \text{TASK} \rangle, \mathcal{M}^{\langle \text{TASK} \rangle})$ 
23:      $\mathcal{R}.\text{append}(dict)$ 

```

However, we find that when the experience repository is not comprehensive, the retrieved tasks that are most similar to the current task may be completely irrelevant. Such completely irrelevant experiences may instead have a negative impact on solving the current task (Zhang et al., 2022). To overcome the above problem and avoid interference from non-homologous tasks in the experience repository, this paper proposes an outlier retrieval method based on Z-score. This method enables even a smaller-scale experience repository to play a positive role. Specifically, if there are indeed no similar tasks in the experience repository, no references are used, and reasoning is conducted directly within the MCF.

Assume that the current experience repository \mathcal{R} has already stored L experiences, which is greater than l . Then, when facing $\langle \text{TASK} \rangle$, the retrieval process is as follows:

Firstly, the current task and the tasks already existing in the experience repository are collectively represented as $\{\mathcal{T}_i\}_{i=0}^L$, where \mathcal{T}_0 describes the current task $\langle \text{TASK} \rangle$. Then, calculate the overall similarity between each task \mathcal{T}_i and other task \mathcal{T}_j , obtaining the similarity set D :

$$D = \{\overline{d_0}, \overline{d_1}, \dots, \overline{d_L}\},$$

$$\overline{d_i} = \frac{1}{n-1} \sum_{j=0, j \neq i}^L \text{Cos}(E(\mathcal{T}_i), E(\mathcal{T}_j)) + \text{Mat}(G(\mathcal{T}_i), G(\mathcal{T}_j)). \quad (12)$$

Next, the Z-score of the average overall similarity $\overline{d_0}$ corresponding to $\langle \text{TASK} \rangle$ can be computed, namely:

$$z_0 = \frac{\overline{d_0} - \mu}{\sigma}, \quad (13)$$

where μ is the mean of D , and σ is the standard deviation of D .

If $z_0 > \lambda$, then $\overline{d_0}$ is considered an outlier in D . This means there is no similar task experience of the same category in the current empirical database. λ is a hyperparameter, which will be discussed for its value in Section 4.2.3.

Table 1

Relevant information about the dataset used.

Domain	Dataset	Number	Average words	Answer format
Arithmetic	MultiArith	600	31.8	Number
	GSM8K	1319	46.9	Number
	AddSub	395	31.5	Number
	AQUA-RAT	254	51.9	Multiple choice
	SingleEq	508	27.4	Number
Commonsense	SVAMP	1000	31.8	Number
	CSQA	1221	27.8	Multiple choice
Symbolic	StrategyQA	2290	9.6	Yes/No
	Letter	500	15.0	String
	Coin Flip	500	37.0	Yes/No

4. Experiments

In this section, the datasets used and the specific experimental settings will be first introduced (see Section 4.1). Following this, extensive experiments on the hyperparameters involved were conducted to gain a deeper understanding of the intrinsic workings of the proposed method (see Section 4.2). Additionally, ablation studies on different modules and comparative experiments with other methods were also performed to further substantiate the effectiveness and superiority of this work (see Sections 4.3 and 4.4).

4.1. Experimental setup**4.1.1. Datasets**

As shown in Table 1, the experiments in this paper were evaluated on ten reasoning datasets, which can be categorized as follows:

Arithmetic reasoning: MultiArith (Roy & Roth, 2015) represents a collection of mathematics word problems necessitating multiple reasoning steps and operations; GSM8K (Cobbe et al., 2021) is a collection of high-quality, linguistically diverse, grade-school mathematics word problems crafted by human problem writers; AddSub (Hosseini, Hajishirzi, Etzioni, & Kushman, 2014) consists of a series of addition and subtraction mathematics word problems; AQUA-RAT (Ling, Yogatama, Dyer, & Blunsom, 2017) offers a collection of mathematics word problems accompanied by natural language rationales; SingleEq (Koncel-Kedziorski, Hajishirzi, Sabharwal, Etzioni, & Ang, 2015) encompasses single-equation grade-school mathematics word problems, incorporating multiple mathematical operations on nonnegative rational numbers and one variable; SVAMP (Patel, Bhattamishra, & Goyal, 2021) presents a set of challenging mathematics word problems designed for students up to grade 4, introducing simple variations to an existing dataset.

Commonsense reasoning: CSQA (Talmor, Herzig, Lourie, & Berant, 2019) poses questions with complex semantics that frequently necessitate reasoning based on prior knowledge; StrategyQA (Geva et al., 2021) is a commonsense question-answering task that employs a Yes or No format and requires models to engage in multi-hop reasoning to arrive at the answers.

Symbolic reasoning: Last Letter Concatenation (Wei et al., 2022) involves a task where the model concatenates the last letters of each word; Coin Flip (Wei et al., 2022) aims to determine whether a coin remains heads up after being flipped or not flipped by individuals.

4.1.2. Implementation details

Unless otherwise specified, the experiments in this article are conducted using OpenAI's GPT-3 (text-davinci-002 version with 175 billion parameters) to power the intelligent agents. The nomic-embed-text-v1 (Nussbaum, Morris, Duderstadt, & Mulyar, 2024) model is utilized as the embedding model for text-to-vector conversion. AMR3-structbart-L (Drozhdov et al., 2022) is used to parse text into AMR graphs. The degree of matching between two different AMR graphs is calculated using the Smatch library (Cai & Knight, 2013). The hyperparameters involved in our method are discussed in the parameter investigation.

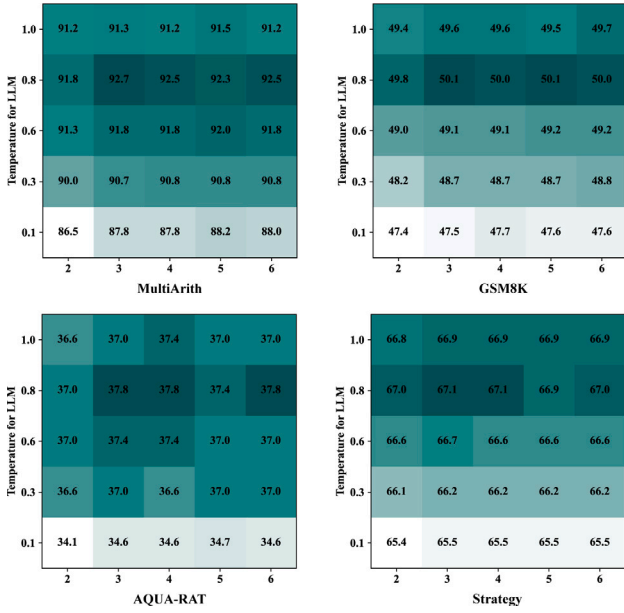


Fig. 4. Parameter investigation experiments were conducted on datasets MultiArith, GSM8K, AQUA-RAT, and strategy. The horizontal axis of each subplot represents the number of users and assistants. The vertical axis represents the values of temperature. The darker the color of the squares above the subplot, the higher the accuracy achieved under the corresponding parameters; the lighter the color, the lower the accuracy.

As presented in Table 1, this study utilizes ten datasets of varying sizes. The StrategyQA dataset is the largest, comprising 2290 samples. The GSM8K, SVAMP, and CSQA datasets are of moderate size, containing 1319, 1000, and 1221 samples, respectively. The MultiArith, SingleEq, Letter, and Coin Flip datasets are smaller, consisting of 600, 508, 500, and 500 samples, respectively. The AddSub and AQUA-RAT datasets are the smallest, with 395 and 254 samples. For parameter exploration, we selected four datasets based on their size: StrategyQA (largest), GSM8K (moderate), MultiArith (smaller), and AQUA-RAT (smallest).

4.2. Parameter investigation

4.2.1. Number of roles and model temperature

The temperature of large language models is a hyperparameter used to adjust the diversity of the generated text. In the MCF, this diversity can be stimulated by configuring multiple agents with the same objective to generate different responses to the same problem. To further understand the operation mechanism of MCF, this paper conducts a parameter investigation experiment on the number of roles N and the model temperature settings. The specific experimental results are shown in Fig. 4.

The best result on the MultiArith dataset is 92.7%, with a corresponding temperature of 0.8 and an agent count of 3. The worst result is 86.5%, with a corresponding temperature of 0.1 and an agent count of 2. The best result on the GSM8K dataset is 50.1%, with a corresponding temperature of 0.8 and an agent count of 3, 4, and 6. The worst result is 47.4%, with a corresponding temperature of 0.1 and an agent count of 2. The best result on the AQUA-RAT dataset is 37.8%, with a corresponding temperature of 0.8 and an agent count of 3, 4, and 6. The worst result is 34.1%, with a corresponding temperature of 0.1 and an agent count of 2. The best result on the Strategy dataset is 67.1%, with a corresponding temperature of 0.8 and an agent count of 3 and 4. The worst result is 65.4%, with a corresponding temperature of 0.1 and an agent count of 2.

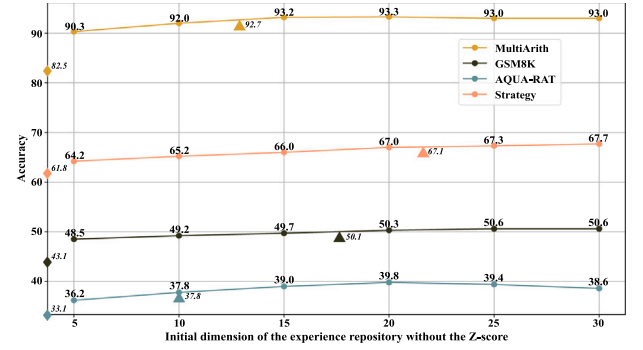


Fig. 5. The impact of the initial dimensions of the experience repository on performance across the MultiArith, GSM8K, AQUA-RAT, and strategy datasets. Different colors represent different datasets. The \blacklozenge positions on each line represent the results of agent dialogues on that dataset. The \blacktriangle positions on each line represent the results of MCF on that dataset.

Overall, a consistent regularity is observed across the four datasets. For any N , as the temperature increases from 0.1, performance gradually improves and reaches its peak at 0.8. When the temperature continues to increase to 1.0, performance slightly decreases. This indicates that at lower temperatures, even with multiple agents set, the responses obtained are still single-faceted. Conversely, at excessively high temperatures, the agents' responses deviate from the instructions, leading to performance degradation.

Furthermore, at different temperatures, the performance of $N = 3$ is superior to $N = 2$, achieving the optimal results. When N continues to increase, the performance shows slight variations, but not significantly. This demonstrates that at an appropriate temperature, multiple agents with the same objective can indeed stimulate the diversity of large language models, resulting in a richer response space. Therefore, based on the above results, this study suggests setting the temperature to 0.8 and the number of agents N to 3.

4.2.2. Initial dimension of the repository

To enable the Agent to summarize and index effective experiences, the repository R is configured into MCF. However, when the repository is not comprehensive, the most similar task retrieved may be completely unrelated to the current task. Such completely unrelated experiences can negatively impact the solution of the current task (Zhang et al., 2022). In other words, the initial dimensions l of the repository when it is first used have a certain impact on performance.

As shown in Fig. 5, we conducted experiments on the MultiArith, GSM8K, AQUA-RAT, and Strategy datasets. It can be observed that when the repository is opened for use after accumulating 5 experiences, the results obtained are significantly higher than those of Agent Dialogues across all four datasets. This indicates that enabling the agent to summarize and index experiences can indeed alleviate the agent's hallucinations. However, the results are still lower than those achieved by the MCF, which confirms that an inadequate repository can have a counterproductive effect. When the initial dimension is gradually expanded to $l = 15$, the performance on the two smaller datasets, MultiArith and AQUA-RAT, surpasses that of MCF. Further increasing the initial dimension to 20 allows the experience pool to provide positive assistance on GSM8K, achieving significantly better performance than the original MCF. Additionally, on the largest dataset, Strategy, the results are basically consistent with those of MCF. This indicates that the initial dimension of the repository is influenced by the size of the dataset.

However, it is worth noting that when l is too large, its performance will decrease again. We believe that as the initial dimension increases, the effect of the repository will gradually diminish, ultimately making the performance equivalent to that of the MCF. Based on the above experiments, l is set to 20.

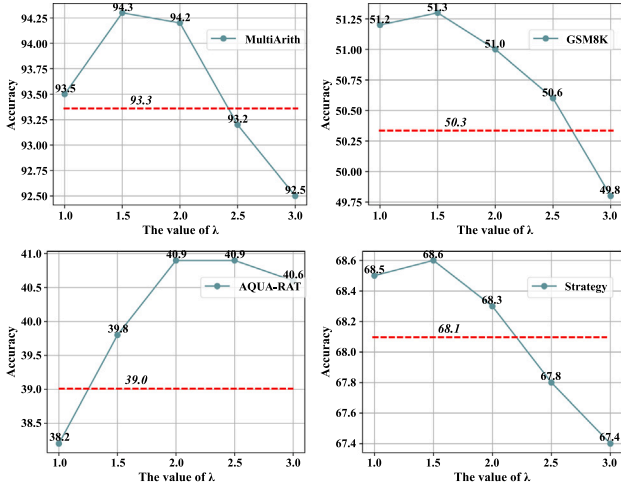


Fig. 6. Performance of different λ on datasets MultiArith, GSM8K, AQUA-RAT, and strategy. In each subplot, the red dashed line represents the best results achievable without using outlier detection methods.

4.2.3. Threshold value of Z-score

Fig. 6 investigates the threshold λ in outlier retrieval. On MultiArith, the optimal result of 94.3% was achieved when $\lambda = 1.5$. The performance was better than the original retrieval method when $\lambda = 1, 1.5, 2$; on GSM8K, the best result of 51.3% was obtained when $\lambda = 1.5$. The performance was better than the original retrieval method when $\lambda = 1, 1.5, 2, 2.5$; on AQUA-RAT, the best result of 40.9% was achieved when $\lambda = 2, 2.5$. The performance was better than the original retrieval method when $\lambda = 1.5, 2, 2.5, 3$; on Strategy, the best result of 68.6% was obtained when $\lambda = 1.5$. The performance was superior to the original retrieval method when $\lambda = 1, 1.5, 2$.

Unlike its performance with the other three datasets, the outlier detection method does not work well on the AQUA-RAT dataset when λ is set to a smaller value. Moreover, the optimal performance corresponds to a larger λ . We believe this is influenced by the size of the dataset. AQUA-RAT only has 254 samples, making its sample space relatively sparse, so the best value for λ is also larger. The impact of dataset size on the value of λ is also evident in the other three datasets, such as the largest dataset Strategy, and the second-largest dataset GSM8K, where λ values of 1 or 1.5 are clearly superior to 2.5, and 3.

Overall, in the three datasets mentioned above, when $\lambda = 1.5, 2$, the performance is consistently better than the original retrieval methods. This fully validates that the Z-score-based outlier retrieval can avoid interference from dissimilar tasks in the empirical database, enabling smaller empirical databases to also have a positive effect. Considering the size of the other datasets in this paper, λ is suggested to be 1.5 in the proposed method.

4.3. Ablation study

4.3.1. Stability from MCF

Due to the impact of the random sampling strategy, agents with the same settings may give completely different responses under the same conditions. This further exacerbates the hallucinations and instability of LLM-based agents. However, the proposed MCF can fully activate the response space and filter out the optimal responses.

To verify the stability of MCF, the experiment shown in Fig. 7 was conducted. As can be seen, the gaps between the best and worst results for Agent Dialogues on the MultiArith, GSM8K, and AQUA-RAT datasets are 3.4%, 2.1%, 1.5%, and 1.1%, indicating that the performance of Agent Dialogues is not stable. We found that reducing the model's temperature can make its output and performance more

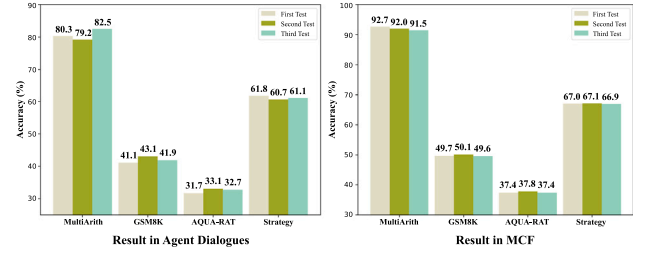


Fig. 7. The left figure shows the results of three tests of agent dialogues on the MultiArith, GSM8K, AQUA-RAT, and strategy datasets. The right figure shows the results of three tests of MCF.

stable, but this also closes off more possibilities (Chang, Reitter, Ak-sitov, & Sung, 2023), which may include high-quality responses. On the other hand, the gaps between the best and worst results for MCF on the MultiArith, GSM8K, and AQUA-RAT datasets are 1.2%, 0.5%, 0.4%, and 0.2%, respectively. It is evident that MCF can fully utilize the diversity brought by the random sampling strategy, filtering out hallucinations while improving the agent's performance.

4.3.2. Effectiveness of the AMR

In order to verify the effectiveness of the proposed similarity calculation method, Table 2 presents the relevant experimental results. It can be observed that when only semantic similarity is used, MCF's performance across all datasets shows a significant improvement compared to the baseline Agent Dialogues, with the average accuracy increasing from 83.0% to 83.7%. This indicates that even relying solely on simple semantic similarity can significantly enhance MCF's reasoning ability.

After incorporating historical context, MCF's performance further improves, with the average accuracy reaching 83.9%. This suggests that historical dialogue information can help the model better understand the context in certain cases, especially when the historical information is critical for reasoning. However, the overall improvement is limited, indicating that the introduction of historical context does not contribute equally to all tasks, and other information may need to be integrated for further performance optimization.

Therefore, the AMR method was introduced, considering the role of sentence structure and keywords. The AMR method achieved the best results across all datasets, with the average accuracy rising to 84.7%, which is an increase of 1.0% compared to the semantic similarity method and an improvement of 1.7% over the baseline. This significant improvement highlights the importance of advanced features such as sentence structure and keywords in reasoning tasks.

In this experiment, semantic similarity, historical dialogues, and the AMR method were introduced in sequence and progressively combined, demonstrating the unique contribution of each method to the performance of MCF. Semantic similarity significantly enhanced the model's basic reasoning ability, while historical dialogue information further improved the model's understanding of the context. However, it was only after the introduction of the AMR method that the model's full potential was activated, significantly improving its performance on complex reasoning tasks. Ultimately, the combination of these three methods allowed the model to achieve optimal performance from multiple dimensions.

4.3.3. Effectiveness of the experience repository

To further verify the effectiveness of the experience repository in the MCF, this section visualizes the impact of an expanding repository on performance across different datasets. Specifically, MultiArith, GSM8K, AQUA-RAT, and Strategy are evenly divided into 10 groups. For example, MultiArith has 600 samples, meaning each group has 60 samples. AQUA-RAT has 254 samples, so the first six groups each have

Table 2

Performance differences of MCF under different similarity methods. **Semantic** refers to directly computing the cosine similarity between the vectors of two text segments (see Eq. (7)); **Historical** refers to incorporating the historical dialogues between agents (see Eq. (8)); **AMR** further considers the impact of sentence structure and keywords (see Eq. (10)). The first row in the table does not use any similarity method, representing the results of the baseline Agent Dialogues. All experiments in this table were conducted using GPT-3.5-Turbo-0613.

Method			Arithmetic						Commonsense		Symbolic		Average
Semantic	Historical	AMR	MultiArith	GSM8K	AddSub	AQuA-RAT	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin	
			96.7	78.2	89.4	62.6	93.7	81.7	77.7	68.3	83.4	98.4	83.0
✓			97.2	79.0	89.6	63.4	94.9	83.0	77.7	69.0	83.8	99.0	83.7
✓	✓		97.3	79.4	89.9	63.4	95.5	83.3	77.9	69.1	83.8	99.0	83.9
✓	✓	✓	98.0	80.6	90.6	63.8	95.7	84.1	79.1	70.7	85.0	99.6	84.7

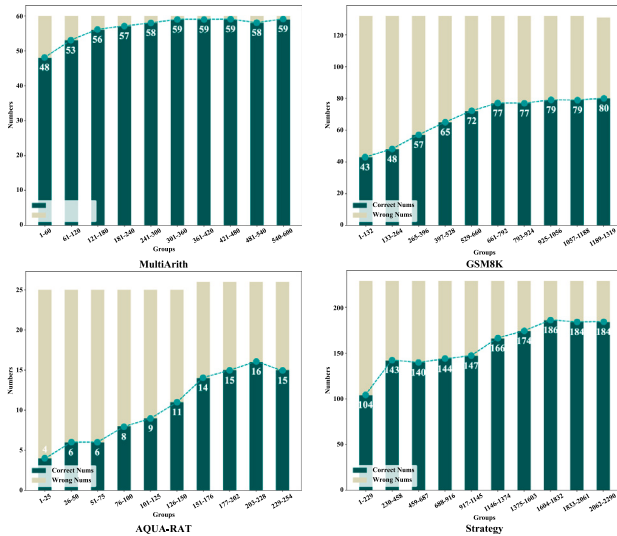


Fig. 8. The trend in the number of tasks correctly solved on the datasets MultiArith, GSM8K, AQUA-RAT, and Strategy. Each subplot's horizontal axis represents different sample groups, while the vertical axis indicates the number of samples. The green part of the bars represents the correct samples, and the brown part represents the incorrect samples. The numbers are markers for the count of correct samples.

25 samples, and the last four groups each have 26 samples. The number of correct and incorrect samples in each group is recorded.

From Fig. 8, it can be seen that in the first set of 60 samples of MultiArith, 48 were correctly answered. In the second set, the number of correct answers increased to 53. The growth trend gradually converges, reaching up to 59 correct answers out of 60 samples. The same situation was observed in GSM8K, AQUA-RAT, and Strategy. This indicates that as the repository accumulates, the positive impact it provides to the agents becomes increasingly stable.

Additionally, we observed that after the fifth group in MultiArith, the sixth group in GSM8K, the seventh group in AQUA-RAT, and the eighth group in Strategy, the number of correct samples in each group shows a clear convergence. We believe this is a reasonable phenomenon because, without the use of ground truth, the assistance that the experience repository can provide is certainly limited. Furthermore, it is important to note that the oscillations after convergence are considered normal fluctuations caused by the varying difficulty levels of different groups.

4.3.4. Performance on different LLM

To verify the robustness of the proposed method, different LLMs were used as the core of the agents for experiments. As shown in Table 3, when using LLAMA2-13B (Touvron et al., 2023), the results with the MCF outperformed those with only the dialogue module across ten different datasets. The average accuracy improved from 51.6% to 54.6%. With the addition of the repository module, performance was further enhanced, achieving optimal results on all datasets, with the

average accuracy increasing to 55.7%. When the large language model was switched to text-davinci-002, GPT-3.5-Turbo-0613, and GPT-4-0613, the MCF also had a significant effect. Results on the ten different datasets outperformed those using only the dialogue module, with average accuracy improvements of 5.4%, 1.7%, and 1.7%, respectively. The repository module remained strong, outperforming the former on all ten datasets. The average accuracy further increased by 1.3%, 0.7%, and 0.3%, respectively.

The stable improvements brought by the collaborative filtering and the repository module fully demonstrate the effectiveness of different modules. The stable performance improvement under different large language models highlights the robustness of the proposed method. Furthermore, we can also observe that the base model indeed has a significant impact on the inference results; the stronger the base model, the better the final outcome.

4.4. Comparative experiment

This paper compares the accuracy of the proposed method with some existing methods on ten different types of reasoning datasets. Zero-Shot (Kojima et al., 2022) directly asks the large language model to answer the questions; Zero-Shot-CoT (Kojima et al., 2022) guides the large language model to reason through specific prompts; Few-Shot (Wei et al., 2022) requires the large language model to solve the current task by referring to existing examples; Manual-CoT (Wei et al., 2022) manually adjusts the reference examples to form a chain of thought, further stimulating the reasoning ability of the large language model. AutoCoT (Zhang et al., 2022) proposes a method for automatically constructing reference examples and their chains of thought, freeing human resources while further enhancing the model's reasoning ability. This paper constructs the problem-solving path through role dialogue during cross-examination, and further optimizes the solution method with multi-agent collaborative filtering. In addition, the dynamically updatable experience pool further improves the reasoning efficiency for similar tasks. As shown in Table 4, the proposed method achieves the best results on eight datasets. It only ranks second on CSQA, below Few-Shot, and on Coin, below AutoCoT.

Table 5 presents a comparison of the proposed method with some existing methods when using GPT-3.5-Turbo-0613. Some of these works simulate real human cognitive behaviors. For example, Role-Play Prompting (Kong et al., 2024) explores the use of role-playing to improve the reasoning capabilities of LLMs, inspired by the successful application of role-playing in downstream tasks. Self-correct (Huang et al., 2023) enables LLMs to refine their answers through self-reflection. Self-consistency (Wang et al., 2022) uses a voting mechanism to select the most consistent answer from multiple responses. Multi-agent Debate (Du et al., 2024) and Peer Review (Xu et al., 2023) achieve iteration and optimization of solutions through debates and reviews among different agents.

Least-to-Most (Zhou et al., 2022) and Zero-shot PS+ (Wang et al., 2023) decompose problems and then solve each subproblem sequentially. Zhong et al. (2024) focuses on the shortcomings of CoT in handling complex mathematical problems and proposes Deeply Understanding the Problems (DUP) to avoid LLM's semantic misunderstanding errors. This paper decomposes tasks through role-playing and

Table 3

The method proposed in this article performs across various large language models. The **Dialogue** module is the Agent Dialogues; the **Filtering** module refers to the multi-role collaborative filtering framework; and the **Repository** module refers to the shared experience repository based on Z-score outlier retrieval.

Model	Method			Arithmetic						Commonsense		Symbolic		Average
	Dialogue	Filtering	Repository	MultiArith	GSM8K	AddSub	AQuA-RAT	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin	
LLAMA2-13B	✓			83.2	28.7	70.1	25.6	69.1	43.8	65.3	55.1	16.8	58.8	51.6
	✓	✓		86.3	30.1	73.4	29.1	74.0	46.5	67.4	56.8	20.4	62.0	54.6
	✓	✓	✓	87.7	30.6	75.2	31.1	75.0	47.1	68.1	57.2	21.2	63.6	55.7
text-davinci-002	✓			82.5	43.1	80.3	33.1	83.3	65.9	73.3	61.8	58.8	94.8	67.7
	✓	✓		92.7	50.1	86.1	37.8	90.2	71.9	76.1	67.1	59.6	99.2	73.1
	✓	✓	✓	94.3	51.3	87.8	40.9	91.1	72.7	76.8	68.6	60.8	99.2	74.4
GPT-3.5-Turbo-0613	✓			96.7	78.2	89.4	62.6	93.7	81.7	77.7	68.3	83.4	98.4	83.0
	✓	✓		98.0	80.2	90.6	63.8	95.7	84.1	79.1	70.7	85.0	99.6	84.7
	✓	✓	✓	98.3	83.8	90.9	64.6	96.9	84.3	79.8	71.0	85.2	99.6	85.4
GPT-4-0613	✓			97.5	94.8	92.7	74.4	94.5	92.9	84.3	80.7	92.4	100.0	90.4
	✓	✓		98.3	96.3	94.4	78.0	96.5	93.8	86.4	81.9	95.4	100.0	92.1
	✓	✓	✓	98.5	96.9	94.7	78.0	97.6	94.1	86.9	82.1	95.4	100.0	92.4

Table 4

Comparison of the method described in this paper with existing methods when using text-davinci-002. **Bold** indicates the best results, and underline indicates the second-best results.

Method	Arithmetic						Commonsense		Symbolic	
	MultiArith	GSM8K	AddSub	AQuA-RAT	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin
Zero-Shot (Kojima et al., 2022)	22.7	12.5	77.0	22.4	78.7	58.8	72.6	54.3	0.2	53.8
Zero-Shot-CoT (Kojima et al., 2022)	78.7	40.7	74.7	33.5	78.7	63.7	64.6	54.8	57.6	91.4
Few-Shot (Wei et al., 2022)	33.8	15.6	83.3	24.8	82.7	65.7	79.5	65.9	0.2	57.2
Manual-CoT (Wei et al., 2022)	91.7	46.9	81.3	35.8	86.6	68.9	73.5	65.4	59.0	97.2
Auto-CoT (Zhang et al., 2022)	<u>92.0</u>	47.9	84.8	<u>36.5</u>	87.0	69.5	74.4	65.4	<u>59.7</u>	99.9
Ours (MCF with Repository)	94.3	51.3	87.8	40.9	91.1	72.7	<u>76.8</u>	68.6	60.8	<u>99.2</u>

Table 5

Comparison of the method described in this paper with existing methods when using GPT-3.5-Turbo-0613. **Bold** indicates the best results, and underline indicates the second-best results.

Method	Arithmetic						Commonsense		Symbolic	
	MultiArith	GSM8K	AddSub	AQuA-RAT	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin
Zero-Shot (Kojima et al., 2022)	79.8	23.8	88.6	28.0	89.4	74.8	73.7	61.1	0.4	21.8
Zero-Shot-CoT (Kojima et al., 2022)	95.8	78.9	83.5	53.5	91.5	77.5	71.5	55.5	70.2	65.8
Least-to-Most (Zhou et al., 2022)	95.5	77.5	91.3	57.4	93.5	80.9	71.9	61.5	83.2	82.8
Self-consistency (Wang et al., 2022)	97.8	81.8	89.4	62.2	93.6	79.8	\	68.2	\	\
Zero-shot PS+ (Wang et al., 2023)	92.0	79.3	86.5	55.9	93.0	80.7	68.8	62.8	60.6	95.4
Role-Play Prompting (Kong et al., 2024)	97.0	79.6	<u>91.1</u>	63.8	98.2	<u>83.8</u>	<u>77.2</u>	67.0	<u>84.2</u>	<u>98.8</u>
Self-correct (Huang et al., 2023)	93.3	74.1	82.6	53.7	89.1	76.4	\	52.9	\	\
Multi-agent Debate (Du et al., 2024)	97.2	81.6	88.1	62.6	93.8	78.4	\	69.4	\	\
Peer Review (Xu et al., 2023)	<u>98.0</u>	<u>83.2</u>	89.9	65.4	94.2	83.6	\	<u>69.8</u>	\	\
DUP (Zhong et al., 2024)	97.8	82.3	92.1	60.2	94.9	82.5	74.5	68.5	81.2	97.6
Ours (MCF with Repository)	98.3	83.8	90.9	<u>64.6</u>	<u>96.9</u>	84.3	79.8	71.0	85.2	99.6

Table 6

Comparison of the method described in this paper with existing methods when using GPT-4-0613. **Bold** indicates the best results, and underline indicates the second-best results.

Method	Arithmetic						Commonsense		Symbolic	
	MultiArith	GSM8K	AddSub	AQuA-RAT	SingleEq	SVAMP	CSQA	Strategy	Letter	Coin
Zero-Shot-CoT (Kojima et al., 2022)	97.8	94.6	92.4	72.8	95.0	90.4	\	\	\	\
Least-to-Most (Zhou et al., 2022)	97.1	92.1	92.1	71.6	95.0	90.3	\	\	\	\
Zero-shot PS+ (Wang et al., 2023)	<u>98.1</u>	94.3	93.1	75.5	95.3	92.6	\	\	\	\
DUP (Zhong et al., 2024)	<u>98.1</u>	97.1	95.1	<u>77.1</u>	<u>96.0</u>	94.2	\	\	\	\
Ours (MCF with Repository)	98.5	<u>96.9</u>	<u>94.7</u>	78.0	97.6	<u>94.1</u>	86.9	82.1	95.4	100.0

activates the response space of LLM in the form of multiple agents, ultimately achieving competitive results. This study achieved the best results on 7 out of 10 datasets, second-best results on AQUA-RAT and SingleEq, and 90.9% on AddSub, only 1.2% lower than the best result by DUP.

Table 6 showcases the superior LLM GPT-4-0613. It can be seen that the method proposed in this paper achieves the best results on three out of six mathematical datasets. On GSM8K, AddSub, and SVAMP, it achieves the second-best results, being 0.2%, 0.4%, and 0.1% lower than the optimal results, only slightly inferior to DUP, which is specifically designed to solve mathematical problems.

5. Conclusion

In this study, we propose a multi-agent collaborative filtering framework equipped with an experience repository. This framework aims to activate the response space of large language models and identify the optimal solution at time t through opposing instances. Furthermore, the introduction of a Z-score-based outlier retrieval algorithm enhances the performance of experience repository retrieval, avoiding the negative impacts of retrieving irrelevant experiences. Ablation experiments on different modules and comparative experiments with other methods were conducted to gain a deeper understanding of the proposed

method's mechanisms. The model's demonstrated performance further validates the effectiveness and superiority of this work.

Although, MCF has achieved results far surpassing those of CoT, the costs and efficiency of inference have become significant considerations. Additionally, exploring the integration of techniques such as reinforcement learning and multi-agent collaboration could provide more possibilities for mitigating decision hallucinations in large language models. Therefore, in future work, we will fine-tune LLMs using decision trajectories obtained during multi-round dialogues. Ultimately, we aim to enhance the decision-making capabilities of language models without compromising their general abilities.

CRedit authorship contribution statement

Jinxin Shi: Conceptualization, Data curation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Jiabao Zhao:** Conceptualization, Investigation, Writing – original draft, Writing – review & editing. **Xingjiao Wu:** Conceptualization, Methodology, Investigation. **Ruyi Xu:** Supervision, Writing – review & editing. **Yuan-Hao Jiang:** Data curation. **Liang He:** Project administration, Methodology, Investigation.

Funding

This work is supported by the National Natural Science Foundation of China (Grant No. 62207013), and the Science and Technology Commission of Shanghai Municipality, China (Grant No. 22511106103).

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jiabao Zhao reports financial support was provided by National Natural Science Foundation of China. Jiabao Zhao reports financial support was provided by Science and Technology Commission of Shanghai Municipality. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

- Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*.
- Azamfirei, R., Kudchadkar, S. R., & Fackler, J. (2023). Large language models and the perils of their hallucinations. *Critical Care*, 27(1), 120.
- Bai, X., Chen, Y., Song, L., & Zhang, Y. (2021). Semantic representation for dialogue modeling. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing* (pp. 4430–4445).
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 1877–1901.
- Cai, S., & Knight, K. (2013). Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 2: short papers)* (pp. 748–752).
- Chan, C.-M., Chen, W., Su, Y., Yu, J., Xue, W., Zhang, S., et al. (2023). ChatEval: Towards better LLM-based evaluators through multi-agent debate. In *International conference on learning representations*.
- Chang, C.-C., Reitter, D., Aksitov, R., & Sung, Y.-H. (2023). Kl-divergence guided temperature sampling. *arXiv preprint arXiv:2306.01286*.
- Cheng, X., Luo, D., Chen, X., Liu, L., Zhao, D., & Yan, R. (2024). Lift yourself up: Retrieval-augmented text generation with self-memory. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.
- Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Cohen, R., Hamri, M., Geva, M., & Globerson, A. (2023). LM vs LM: Detecting factual errors via cross examination. In *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 12621–12640).
- Drozdzov, A., Zhou, J., Florian, R., McCallum, A., Naseem, T., Kim, Y., et al. (2022). Inducing and using alignments for transition-based AMR parsing. In *Proceedings of the 2022 conference of the North American chapter of the association for computational linguistics*.
- Du, Y., Li, S., Torralba, A., Tenenbaum, J. B., & Mordatch, I. (2024). Improving factuality and reasoning in language models through multiagent debate. In *International conference on machine learning*.
- Ezzini, S., Abualhaija, S., Arora, C., & Sabetzadeh, M. (2022). Automated handling of anaphoric ambiguity in requirements: a multi-solution study. In *Proceedings of the 44th international conference on software engineering* (pp. 187–199).
- Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., et al. (2023). RARR: Researching and revising what language models say, using language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics*.
- Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., & Berant, J. (2021). Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9, 346–361.
- Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12), 61–70.
- Guu, K., Lee, K., Tung, Z., Pasupat, P., & Chang, M. (2020). Retrieval augmented language model pre-training. In *International conference on machine learning* (pp. 3929–3938). PMLR.
- He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web* (pp. 173–182).
- Hosseini, M. J., Hajishirzi, H., Etzioni, O., & Kushman, N. (2014). Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 523–533).
- Huang, J., Chen, X., Mishra, S., Zheng, H. S., Yu, A. W., Song, X., et al. (2023). Large language models cannot self-correct reasoning yet. In *International conference on learning representations*.
- Ji, Z., Yu, T., Xu, Y., Lee, N., Ishii, E., & Fung, P. (2023). Towards mitigating LLM hallucination via self reflection. In *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 1827–1843).
- Kirsch, L., Harrison, J., Freeman, C. D., Sohl-Dickstein, J., & Schmidhuber, J. (2023). Towards general-purpose in-context learning agents. In *Advances in neural information processing systems (neurIPS) 2023 workshop on foundation models for decision making*.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 22199–22213.
- Koncel-Kedziorski, R., Hajishirzi, H., Sabharwal, A., Etzioni, O., & Ang, S. D. (2015). Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3, 585–597.
- Kong, A., Zhao, S., Chen, H., Li, Q., Qin, Y., Sun, R., et al. (2024). Better zero-shot reasoning with role-play prompting. In *Proceedings of the 2024 conference of the North American chapter of the association for computational linguistics: human language technologies (volume 1: long papers)* (pp. 4099–4113).
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 9459–9474.
- Li, G., Hammoud, H., Itani, H., Khizbullin, D., & Ghanem, B. (2024). Camel: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.
- Ling, W., Yogatama, D., Dyer, C., & Blunsom, P. (2017). Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th annual meeting of the association for computational linguistics* (pp. 158–167).
- Luo, Z., Xie, Q., & Ananiadou, S. (2024). Factual consistency evaluation of summarisation in the Era of large language models. *Expert Systems with Applications*, Article 124456.
- Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., et al. (2022). Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 conference on empirical methods in natural language processing*.
- Nussbaum, Z., Morris, J. X., Duderstadt, B., & Mulyar, A. (2024). Nomic embed: Training a reproducible long context text embedder. *arXiv preprint arXiv:2402.01613*.
- Patel, A., Bhattamishra, S., & Goyal, N. (2021). Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*. Association for Computational Linguistics.
- Qian, C., Cong, X., Yang, C., Chen, W., Su, Y., Xu, J., et al. (2023). Communicative agents for software development. *arXiv preprint arXiv:2307.07924*.
- Qiao, S., Zhang, N., Fang, R., Luo, Y., Zhou, W., Jiang, Y. E., et al. (2024). AutoAct: Automatic agent learning from scratch for QA via self-planning. In *International conference on learning representations (ICLR) workshop on large language model (LLM) agents*.
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

- Roy, S., & Roth, D. (2015). Solving general arithmetic word problems. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics.
- Shi, W., Han, X., Lewis, M., Tsvetkov, Y., Zettlemoyer, L., & Yih, W.-t. (2024). Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 conference of the North American chapter of the association for computational linguistics: human language technologies (volume 2: short papers)* (pp. 783–791).
- Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2024). Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.
- Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.
- Talmor, A., Herzig, J., Lourie, N., & Berant, J. (2019). Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies* (p. 4149). Association for Computational Linguistics.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. arXiv preprint [arXiv:2307.09288](https://arxiv.org/abs/2307.09288).
- Wan, D., Liu, M., Mckeown, K., Dreyer, M., & Bansal, M. (2023). Faithfulness-aware decoding strategies for abstractive summarization. In *Proceedings of the 17th conference of the European chapter of the association for computational linguistics* (pp. 2864–2880).
- Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., et al. (2022). Self-consistency improves chain of thought reasoning in language models. In *International conference on learning representations*.
- Wang, L., Xu, W., Lan, Y., Hu, Z., Lan, Y., Lee, R. K.-W., et al. (2023). Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics* (pp. 2609–2634).
- Wang, Y., Yao, Q., Kwok, J. T., & Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3), 1–34.
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 24824–24837.
- Xu, Z., Shi, S., Hu, B., Yu, J., Li, D., Zhang, M., et al. (2023). Towards reasoning in large language models via multi-agent peer review collaboration. arXiv preprint [arXiv:2311.08152](https://arxiv.org/abs/2311.08152).
- Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., et al. (2024). Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*.
- Yao, S., Zhao, J., Yu, D., Du, N., Shafraan, I., Narasimhan, K., et al. (2023). React: Synergizing reasoning and acting in language models. In *International conference on learning representations*.
- Zhang, Z., Zhang, A., Li, M., & Smola, A. (2022). Automatic chain of thought prompting in large language models. In *International conference on learning representations*.
- Zhong, Q., Wang, K., Xu, Z., Liu, J., Ding, L., Du, B., et al. (2024). Achieving >97% on GSM8K: Deeply understanding the problems makes LLMs perfect reasoners. arXiv preprint [arXiv:2404.14963](https://arxiv.org/abs/2404.14963).
- Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., et al. (2022). Least-to-most prompting enables complex reasoning in large language models. In *International conference on learning representations*.

Jiabao Zhao is Associate Professor of Computer Science at the Institute of AI Education, East China Normal University. She received the Ph.D. degrees from the School of Computer Science and Technology, East China Normal University. Her research interests include few-shot learning, cross-modal knowledge transferring and explainable AI. Her research results have been published in AAAI, ICME, ICASSP, Neurocomputing, etc.