QuBE: Question-based Belief Enhancement for Agentic LLM Reasoning

Minsoo Kim[1] **Jongyoon Kim**[1] **Jihyuk Kim**[2] **Seung-won Hwang**[*3]
[1]IPAI, Seoul National University [2]Yonsei University [3]Seoul National University
{minsoo9574, john.jongyoon.kim, seungwonh}@snu.ac.kr
jihyukkim@yonsei.ac.kr

Abstract

Despite advancements in Large Language Models (LLMs), many complex tasks are not easily solved in a single inference step, requiring the use of agentic LLMs in interactive environments. However, agentic LLMs suffer from a phenomenon known as *reasoning derailment*, due to the indiscriminate incorporation of observations from partially observable environments. We introduce **QuBE**, a method that enhances agents' focus on task-relevant contexts, by constructing a belief state via question answering. We validate QuBE through experiments in two agentic LLM scenarios with partial observability: 1) a canonical interactive decision-making scenario using text-based game engines, and 2) an interactive retrieval-augmented generation (RAG) scenario using search engines. In the ALFWorld text-based game, QuBE outperforms established baselines by substantial margins, and in the search engine scenario, it achieves marked improvements on the BEIR zero-shot retrieval benchmark. The results demonstrate that QuBE significantly mitigates reasoning derailment, refining the decision-making process of LLM agents in partially observed environments.[1]

1 Introduction

Recent advancements in the general capabilities of large language models (LLMs) has led to an explosion of interest in their deployment as *agents*, capable of autonomously performing complex, long-horizon tasks through interactive environments (Sumers et al., 2023; Muthusamy et al., 2023; Gur et al., 2024; Wang et al., 2024b). LLM-based agents leverage the LLM's capability to interleave environment interaction (*actions*) with reflective reasoning (*rationales*) (Yao et al., 2023; Huang et al., 2022; Shinn et al., 2023), enabling sophisticated action policies for complex tasks.

Despite these advances, LLM-based agents still face a critical issue known as *reasoning derailment* (Yao et al., 2023), a phenomenon in which agents struggle to recover from an initial erroneous rationale, resulting in a further cycle of incorrect reasoning and unsuccessful interactions. In this work, we identify the *partially observable* nature of many interactive environments (Hausknecht and Stone, 2015; Liu et al., 2022; Pleines et al., 2023), as a key factor contributing to reasoning derailment. In partially observable environments, observations available to the agent provide only an incomplete view of the underlying, true state of the environment which influences the agent's task. As such, the agent must effectively resolve the uncertainty and incompleteness in the observed information, to optimize its reasoning and actions.

However, existing LLM agent methods such as ReAct (Yao et al., 2023) and Reflexion (Shinn et al., 2023) are ill-equipped to address this challenge, as they operate directly on raw observed states, stored in a non-selective input context. Rather, such naive utilization of observations exacerbates the accumulation of noisy, incomplete, and task-irrelevant observed states, making reasoning derailment more likely. We illustrate this phenomenon in Fig.1-(a). In this ALFWorld example, the LLM agent erroneously generates the action "go to desk 1" despite already being there, which acts as a trigger for subsequent reasoning derailment due to a failure to distinguish between its current location (desk 2) and the target object's location (desk 1).

To mitigate reasoning derailment in LLM agents, we propose **Qu**estion-based **B**elief **E**nhancement (QuBE), to dynamically construct a focused, task-relevant *belief state*, which estimates the unobserved true state of the environment. Specifically, we formulate belief state construction as a question-answering process, implemented using an LLM with access to environment tools. By enabling the LLM to direct queries to both the agent's context

*Corresponding author.
[1]https://github.com/mnskim/QuBE

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 21403–21423
November 12-16, 2024 ©2024 Association for Computational Linguistics

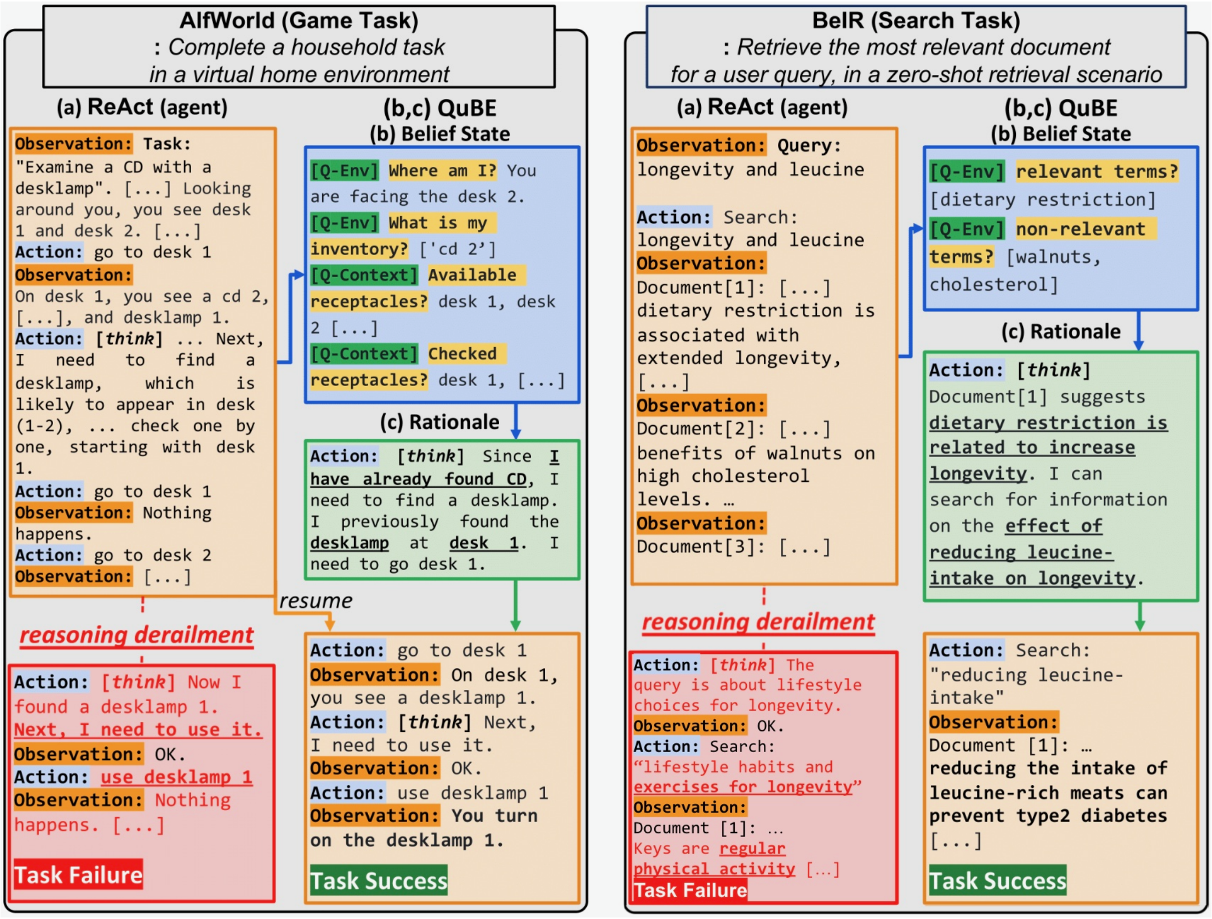

Figure 1: Comparative illustration of baseline agent (a) ReAct, with the proposed method, (b,c) QuBE, on game (ALFWorld) and search (BEIR) tasks. The baseline agent's trajectory is shown in the orange box, with reasoning derailment indicated in red, eventually leading to task failure. As a solution, QuBE first (b) constructs belief states from the raw observation, using a question-answering process, shown in blue, and then (c) provides the agent with feedback in the form of a textual rationale, indicated by `[think]`, shown in the green box. By resolving the noise and incompleteness of partial observations, QuBE addresses reasoning derailment.

history, as well as the environment itself, we ensure the construction of a belief state that resolves the ambiguity and noisiness of partial environment observations in a grounded manner.

In Fig.1-(b,c), we illustrate how QuBE addresses reasoning derailment through the construction of a belief state. Using a series of queries, QuBE constructs a concise, task-relevant belief state (shown in the blue box) which fills in the gaps due to partial observability. In the ALFWorld task, the belief state includes details like checked receptacles, inferred from the agent's context history (Q-Context), as well as the agent's current inventory and location, which are issued as queries to the game engine (Q-Env). Next, we repurpose the LLM to produce a textual rationale based on the belief state, as shown in the green box. By conditioning on the constructed belief state, the rationale accurately locates the target object—desklamp 1 at desk 1—enabling QuBE to resolve derailment, and guide the agent's next actions toward task success.

We conduct experiments on two unique partial observability scenarios, on a canonical scenario using the text-based game engine, ALFWorld (Shridhar et al., 2021), and a challenging, interactive retrieval scenario using a search engine, on the BEIR benchmark (Thakur et al., 2021). In ALFWorld, QuBE demonstrates significant improvements over established baselines, ReAct and Reflexion, by 28% and 24% absolute in task success rate, respectively. Moreover, in the BEIR benchmark, QuBE consistently enhances performance, achieving a notable increase of 1.3% in nDCG@10 over ReAct, on the challenging task of zero-shot retrieval.

The main contributions of our work are summarized as follows:

- We propose QuBE, which mitigates reasoning derailment in partially observable environments, through belief state generation using tool-enabled question answering.

- We validate the effectiveness of QuBE through improvements in task performance in canonical (game engines) and real-world (search engines) interactive scenarios, over strong baselines.
- To provide insight into the derailment phenomenon, we perform qualitative analysis which shows that derailment errors are caused by partial observability, and show that QuBE effectively reduces derailment errors.

2 Environments

In this section, we describe in detail our selection of environments, which aims to represent a diverse selection encompassing different task types and reward availabillity.

2.1 Tasks in Partially Observable Environments

We target scenarios in which an LLM-based agent interacts with a partially observable environment to solve a given task. In a partially observable environment, the state observed by an agent is an incomplete representation of an unobserved *true environment state*, which explicitly represents all factors that can influence the agent's task execution. As the spectrum of possible environments is vast, we choose two environments based on 1) representativeness, and 2) reward availability, to achieve a diverse selection representing the range of partially observable scenarios.

Task in GE First, for the GE scenario, we consider ALFWorld (Shridhar et al., 2021), an interactive embodied agent task, widely studied in recent works (Yao et al., 2023; Shinn et al., 2023; Prasad et al., 2024; Zhao et al., 2024), and agent benchmarks (Liu et al., 2024; Gioacchini et al., 2024). ALFWorld is a text-based game engine designed to align with a 3-D virtual home environment simulator, ALFRED (Shridhar et al., 2020). In ALFWorld, the agent is tasked to complete a given instruction, e.g. "Find a desklamp and turn it on.", by navigating and interacting with the environment.

Partial Observability in GE Text-based games, such as ALFWorld, are canonical examples of partially observed environments, where agents must reason about the world solely through observations which consist of incomplete textual descriptions about the game state (Hausknecht et al., 2019; Côté et al., 2018; Ammanabrolu et al., 2020).

In these observations, factors that may be critical for carrying out the task can be often omitted. For example, the underlying game state in an ALFWorld scenario may contain dozens of objects in different locations throughout the virtual home environment, but any given observation will only provide a limited piece of information, e.g. "On the towelholder 1, you see a towel 1". Often, the agent must *infer* information beyond the raw observation alone to progress the task: For example, if the agent's task is to interact with "a cup 2", it needs to infer from the above observation that it must investigate a location other than the towelholder 1, and that it does not need to visit this location again. This incurs an overhead of additional reasoning complexity, and in the following section, we show that this is where the reasoning derailment errors of LLM agents occur.

Task in SE Another representative scenario we consider involves using interactive RAG for document retrieval tasks. We choose the BEIR zero-shot retrieval benchmark (Thakur et al., 2021), a popular benchmark for testing the zero-shot retrieval capabilities of models (Ni et al., 2022; Wang et al., 2022a), where single-turn retrieval models are reported to struggle due to the difficulty of generalizing to out-of-distribution domains (Kim et al., 2023). Addressing this challenge, we consider agentic LLM, which engages in an interactive session of queries. In this setup, the documents ranked by a search engine serve as the observations within the environment. The primary objective of the agent is to enhance the ranking quality, assessed by the Normalized Cumulative Discount Gain (nDCG@k) metric (Järvelin and Kekäläinen, 2002; Wang et al., 2013).

Partial Observability in SE In contrast to the multi-hop QA scenario adopted in prior works such as ReAct and Reflexion (Yao et al., 2023; Shinn et al., 2023), where the query intents are simple and explicit, this is rarely the case in the more general search scenarios, where the user query is only a partial observation of true information need (Rieh et al., 2006), containing ambiguities or being underspecified.

Especially in the zero-shot setting of BEIR, the problem of inferring true user query intent involves a higher degree of uncertainty, and is more prone

Errors in GE	Definitions	ReAct	QuBE
Location	The model makes incorrect assumptions about its own location or the spatial properties of the environment, leading to actions that are spatially incoherent.	16%	0%
Object	The model fails to accurately perceive and track objects, leading to an incorrect understanding of the relationship between itself and objects in the environment.	14%	8%
Task State	The model incorrectly assesses the current state of the task, becoming unable to progress the task.	6%	2%

Table 1: Types of reasoning errors in 50 examples of ALFWorld, and their rate of occurrence. Errors are identified by evaluating the examples manually, examining the agent's action and reasoning trace. We identify three major error types, related to location and object perception, and task state tracking.

Errors in SE	Definitions	ReAct	QuBE
Relevant Terms	The model fails to perceive relevant terms in observed documents, omitting them as rationale terms.	16%	14%
Non-relevant Terms	The model fails to correctly perceive non-relevant terms from the observed documents, incorrectly extracting them as rationale terms.	14%	13%

Table 2: Types of reasoning errors in 50 examples of the TREC-COVID (v2) dataset of BeIR, and their rate of occurrence. Errors are identified by evaluating the examples manually, examining the agent's action and reasoning trace. We present a major error type, related to identifying the relevant and non-relevant terms with respect to the query.

to reasoning derailment. Thus, relying solely on the innate commonsense knowledge and reasoning capability of the LLM is often insufficient.

2.2 Reward Availability in Environments

A key distinction between GE and SE is availability of reward signals from the environment. We evaluate our proposed method both on environments with explicit rewards (GE) and implicit rewards (SE). The text-based game environment of GE returns a sparse but explicit reward at the end of the episode in the form of binary task success/failure. On the other hand, a search environment generally does not return such a reward, since this would require knowing the user's satisfaction with the search results. Both are salient scenarios in agent deployment, and both require addressing the derailment phenomenon, as we show in following sections.

2.3 Preliminary Study: Analysis of Reasoning Derailment

As a preliminary study, we manually analyze reasoning derailment in a baseline ReAct agent. Tables 1 and 2 present the main error types identified in occurrences of reasoning derailment in GE and SE, respectively. For GE, we find that derailment is caused by errors in the agent's inferences with respect to: location perception, object perception, and task state tracking, producing error rates 16%, 14%, and 6%. Similarly, in SE, we find that the baseline agent often fails to perceive relevant terms and non-relevant terms among terms in documents given by the search engine, with error rates 16% and 14%, respectively, indicating the challenge of discerning the true search intent from the surface form provided by the user.

As shown in the details of each error type for each task, these errors would be easily resolved in a fully observable environment, i.e. a GE scenario in which the engine returns a full and exhaustive description of the environment state, and a search scenario in which the user's search intent is fully specified without ambiguity. The prevalence of these errors indicate that despite the capabilities of LLM agents, they are not currently able to resolve partial observability in a satisfactory manner.

3 Method

Motivated by our preliminary study, we present our proposed method, QuBE, in the following section. Next, we begin by providing a broad description of standard LLM-based agents, within which we discuss the systematic cause of reasoning derailment.

3.1 Reasoning Derailment in LLM Agents

Formally, the task policy of a rationale-action interleaving LLM agent (i.e., ReAct) at timestep t is parametrized by the LLM, as $\pi(a|c_t)$, a probability distribution over actions $a \in A$, where the action space A is the space of natural language. The policy conditions on the context $c_t = (o_1, a_1, \cdots, o_{t-1}, a_{t-1}, o_t)$, which is also the prompt input provided to the LLM, consisting of the agent's past actions and the partially observed history from the environment, denoted by $a_{t'|t'<t}$ and $o_{t'|t'\leq t}$, respectively. Importantly, a_t can be either a *rationale*, or an *action*, where the former indirectly influences the policy through c_t, but only the latter is actually executed in the environment.

To illustrate the distinction between rationales and actions, we modify the policy formulation to separately indicate rationales, such that c_t refers to history of actions and observations, and rationales are denoted by the set $E_t = \{e_{t'}\}_{t'=1}^{t}$, rewriting $\pi(a|c_t)$ to $\pi(a|c_t, E_t)$.[2]

In Alg.1, we illustrate the full operation of the agent, and describe how the derailment described in Sec.1 occurs: The LLM agent's input context c_t, which is a simple accumulation of partial observations with noisy and incomplete information, can misguide the policy $\pi(a|c_t, E_t)$ (*Line*. 6), leading to the generation incorrect rationales, or unsuccessful actions. Such generations are then incorporated back into c_{t+1} or E_{t+1} (*Line*. 9, *Line*. 16), further harming the subsequent generations, incurring *derailment*, as observed in Sec.2.3.

3.2 Proposed Approach: QuBE

To address reasoning derailment, our proposed approach QuBE aims to alleviate partial observabillity in c_t, by constructing a focused and task-relevant belief state b_t. We show how QuBE modifies the operation of the agent in Alg.1. We leverage an LLM as a tool-enabled question-answering model Q, using it to both direct questions about both the agent's context c_t and E_t, and the environment, as well as answer them (*Line*. 11). After the construction of b_t which rectifies the gaps in the agent's raw observation context, QuBE uses a rationale generation model G, to generate the belief-informed rationale e_t^* (*Line*. 12), which is seamlessly integrated back into the agent via $E_t \cup \{e_t^*\}$. Through this process, e_t^* acts as an update signal for the policy $\pi(a|c_t, E_t)$, mitigating derailment. Note that, a trigger mechanism which leverages environment signals can selectively activate QuBE, for better efficiency. In practice, we use this mechanism in the GE setting[3].

In the following section, we provide detailed descriptions of the implementation of QuBE components in each scenario.

Algorithm 1 Rationale-Action Interleaving LLM Agent. The blue lines indicate augmenting the agent with QuBE.

Input: environment Env, LLM Agent policy π, QuBE LLM components Q and G

```
1:  t ← 0
2:  n ← max timesteps
3:  c_t ← {}
4:  E_t ← {}
5:  while t < n do
6:      a_t ← π(a|c_t, E_t)
7:      if type(a_t) = action then                  ▷ Acting
8:          o_{t+1} ← Env(a_t)
9:          c_{t+1} ← c_t + (a_t, o_{t+1})
10:         if trigger(o_{t+1}) then                 ▷ QuBE
11:             b_t = Q(c_t, E_t, Env)               ▷ Belief State Const.
12:             e_t^* = G(b_t, c_t, E_t)             ▷ Rationale Gen.
13:             E_{t+1} ← E_t ∪ {e_t^*}
14:         end if
15:     else if type(a_t) = rationale then          ▷ Reasoning
16:         E_{t+1} ← E_t ∪ {a_t}
17:     end if
18:     t ← t + 1
19: end while
```

Belief State Construction To construct the belief state, we utilize a question-answering model Q, implemented as prompted LLMs. We guide Q to ask two question types, Q-Env and Q-Context. Q-Env are queries which can be answered by referring to an environment-attached tool, while Q-Context are those answered by the LLM, by re-analyzing the agent's context c_t and E_t to infer new information. When issuing both Q-Env and Q-Context questions, we execute the former questions first, and allow these QA pairs to condition the answers to the latter questions, to improve grounding.

In GE, belief state construction aims to enhance the agent's estimate of the true environment state, by resolving uncertainties and discovering links between different portions of the raw partially observed history. Therefore, we leverage Q-Env for ego-centric information gathering (Ammanabrolu and Hausknecht, 2020), by issuing queries to the environment. More specifically, the LLM can leverage the text-based game environment as a tool using the commands, `env.step("look")` and `env.step("inventory")`. Further, using Q-

[2] Concretely, a rationale e_t is indicated in the LLM output, with the prefix "`[think]`".

[3] Refer to Appendix A.5 for further details.

Context, we *infer* task-relevant information, i.e. the state and task-relevant properties of objects such as receptacles, answered by the LLM from the agent's context history. The question and answer pairs, shown in Fig.1-(b)(*left*), comprise the belief state in GE.

In SE, belief state construction aims to improve the agent's estimate of the true query intent, with the downstream goal of optimizing the retrieval outcome. Following our interactive formulation of retrieval, we treat the retrieved documents as feedback signals (Rocchio Jr, 1971), which can be further leveraged towards enhancing the alignment between the query intent and its representation (Li et al., 2022). Specifically, as Q-Env, we utilize a tool easily attached to the search engine environment, to establish an accurate list of possible keywords that are relevant and non-relevant to the query[4]. To do so, we give Q access to the DeepImpact (Mallia et al., 2021) term-importance estimator, which is trained using contrastive learning to produce an *impact score* of each term within a document, measuring its relevance contribution. Hence, the question and answer pairs for relevant and non-relevant keywords, shown in Fig.1-(b)(*right*), comprise the belief state in SE.

Rationale Generation After constructing the belief state, we generate a rationale e_t^* using another prompted LLM G. Through this, the LLM can concisely integrate existing information, and even generate new inferences through reasoning, grounded on the belief state. In GE, the goal of G is to rationalize the observed derailment, and find an alternative next action to progress the task. In SE, G uses the search results as retrieved context, to further enhances the estimate of the query intent in a detailed, textual form, by verbalizing additional task-relevant information through RAG.

Finally, the generated rationale is integrated back into the agent, as described in Sec.3.2. In GE, the agent proceeds by generating the next action, and in SE, the agent proceeds by generating an updated query. The full prompts of the QuBE components can be found in Appendix A.7.1 and Appendix A.7.2.

4 Experiments

To validate the effectiveness of QuBE, we conduct experiments on our two target settings in GE and SE. In both scenarios, QuBE components are implemented using LLMs, in conjunction with a ReAct LLM agent[5]

4.1 Game Engine Scenario: ALFWorld

We compare QuBE against the following baselines on the ALFWorld benchmark. Following previous works, we evaluate models on the 134 ALFWorld test environments across six different task types, measuring task success rate.

BUTLER BUTLER (Shridhar et al., 2021) is the baseline offline reinforcement learning agent for ALFWorld, trained on 10^5 expert trajectories for each task type.

ReAct ReAct is the agentic LLM method using a prompting approach which interleaves action and reasoning. By making interaction possible for LLMs, ReAct significantly outperforms previous state-of-the-art agents on ALFWorld, including BUTLER.

Reflexion Reflexion extends ReAct, to allow the agent to engage in a task over multiple episodes or trials, and learn from mistakes in previous trials. By providing the agent access to summarized task memories from previously failed trials, Reflexion can rectify reasoning derailment in an iterative manner.

ExpeL ExpeL (Zhao et al., 2024) expands the idea of Reflexion, to utilize an LLM agent to first collect success and failure experiences on the task into an experience pool, and then extracts knowledge from these experiences as insights to guide the LLM.

ADAPT ADAPT (Prasad et al., 2024) builds upon ReAct, by recursively decomposing tasks into sub-tasks, in order to enhance the LLM agent's ability to handle complex tasks.

4.2 Search Engine Scenario: BEIR

On the BEIR benchmark, we compare QuBE against the following baselines, evaluating with the official metric, nDCG@10. The metric measures the quality of the top-10 results, scoring zero when none are relevant and increasing as relevant documents are ranked higher.

[4]Implementation details of the keyword extraction process are provided in A.6.

[5]Implementation details of LLMs are provided in Appendix A.3.

Model	Trial 0	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9
BUTLER	37%	-	-	-	-	-	-	-	-	-
ReAct	57%	-	-	-	-	-	-	-	-	-
Reflexion	53%	57%	66%	69%	69%	72%	72%	72%	72%	72%
ExpeL	59%	60%	63%	64%	-	-	-	-	-	-
ADAPT	-	-	80%	-	-	-	-	-	-	-
QuBE (Ours)	**81%**†	**90%**†	**93%**†	**94%**†	**95%**†	**95%**†	**95%**†	**96%**†	**96%**†	**96%**†

Table 3: Results on AlfWorld, using task success rate as the evaluation metric. The best results are highlighted in **bold**. Statistically significant improvement of QuBE over Reflexion is indicated by †, with negligibly small p-values across all trials.

	NFCorpus	TREC-COVID (v2)	Touché	FiQA	SCIDOCS	SciFact	Average
ColBERT	34.2	76.2	25.3	35.2	15.8	69.0	42.6
ColBERT-PRF	34.8	75.7	28.5	34.5	15.6	68.5	42.9
ReAct/Reflexion	$35.2_{\pm 0.1}$	$75.2_{\pm 0.8}$	$28.3_{\pm 0.3}$	$35.5_{\pm 0.3}$	$15.9_{\pm 0.1}$	$69.2_{\pm 0.4}$	43.2
QuBE (Ours)	$\mathbf{35.7}_{\pm 0.1}$ (15.5%)	$\mathbf{77.6}^{\dagger}{}_{\pm 0.8}$ (6.0%)	$\mathbf{30.3}_{\pm 0.6}$ (10.8%)	$\mathbf{36.5}^{\dagger}{}_{\pm 0.2}$ (1.5%)	$\mathbf{16.2}^{\dagger}{}_{\pm 0.1}$ (7.6%)	$\mathbf{70.9}^{\dagger}{}_{\pm 0.9}$ (2.0%)	**44.5**† (7.2%)

Table 4: Results on BEIR, using nDCG@10 as the evaluation metric. For the LLM-based systems, we conduct 3 runs and report the average and variance of performance. The best results for each dataset are highlighted in **bold**. Statistically significant improvement of QuBE over ReAct/Reflexion is indicated by †. P-values are micro-averaged across the runs to secure valid sample sizes for testing, and reported in brackets below.

ColBERT ColBERT is a widely adopted retrieval model that encodes queries and documents as multiple contextualized embeddings, representing tokens. Relevant documents are identified based on maximum similarity over the query and document embeddings. Since we adopt ColBERT[6] as the search engine, its performance is the base search engine performance.

ColBERT-PRF ColBERT-PRF is a state-of-the-art query expansion approach which identifies expansion terms based on clustering informed by inverse document frequency (IDF) statistics. We compare with ColBERT-PRF as the baseline representing a classical query refinement approach, which does not employ reasoning in an explicit manner, as with LLMs.

ReAct As an agentic LLM baseline for BEIR, we leverage ReAct for interactive query refinement, given top-3 documents from the search engine as observations. The agent interleaves reasoning, which analyzes the query and documents, with the action step, which generates new terms to expand the query with[7]. Note that, since the action space in SE is restricted to issuing a query to the search engine, we view each search operation as a trial, making ReAct and Reflexion equivalent. In practice, since the success or failure of the task cannot be informed during inference in SE, we perform interactions up to a fixed number of timesteps, t, for both ReAct and QuBE. We set t as 2, i.e. one rationale step and one action step, based on our preliminary analysis using ReAct, where the nDCG@10 performance did not improve by performing multiple steps of reasoning. Finally, considering the non-determinism of LLM inference in SE (for which we set the temperature by 0.5), we run experiments 3 times on each dataset, for both ReAct and QuBE, and report average performance along with standard deviation.

[6]We use the open-sourced ColBERT checkpoint, with settings described in the official GitHub repository: https://github.com/stanford-futuredata/ColBERT

[7]Based on empirical findings in ColBERT-PRF, we generate at most 10 query terms for both ReAct and QuBE.

5 Results and Analysis

5.1 Results on AlfWorld

We report the results of experiments on AlfWorld in Table 3. First, we examine the results without applying Reflexion and observe that our method improves over ReAct significantly, outperforming it by 28% on task success rate. Compared to the recent baselines which build on the ReAct paradigm, ExpeL and ADAPT, QuBE shows strong performance, outperforming both. These results demonstrate that QuBE effectively addresses reasoning derailment through belief state construction, enhancing LLM agent performance significantly.

Next, we report the results (trials 2~9) combining our approach with Reflexion. We observe that the performance of QuBE improves meaningfully, around 15%, over the course of 10 trials, outperforming Reflexion in final performance by 24%, and solving nearly all AlfWorld tasks. Additionally, we observe that QuBE outperforms the maximum performance of all compared methods with only a single trial, indicating that its approach of resolving partial observability can translate to *significant gains in sample efficiency*. These results support the value of our approach in addressing reasoning derailment in LLM agents.

5.2 Results on BEIR

We report the results of experiments on the BEIR benchmark in Table 4. Compared to the ColBERT baseline using the initial query for the search task, ColBERT-PRF, which issues a new query using augmented query terms from observed documents, shows limited improvements. Meanwhile, the ReAct LLM agent utilizes reasoning for the task, outperforming ColBERT-PRF, but only marginally. These results indicate the challenge of query refinement in the zero-shot setting. Further, the results indicate that raw partial observations, in the form of retrieved documents, are not necessarily beneficial in query refinement, and that even with LLM's reasoning capability, this remains a difficult task.

In contrast, by addressing partial observability in the observations through belief state construction, QuBE is able to leverage LLM reasoning more effectively, and consistently outperforms ReAct on all datasets by a notable margin, validating the effectiveness of our approach for enhancing LLM agents. Note that, though the improvements of 1 or 2 points in nDCG@10 might appear modest, they are significant in the context of zero-shot settings, where enhancing the query representation according to the knowledge embedded in the retriever is challenging.

5.3 QuBE Mitigates Reasoning Derailment

Extending our preliminary study of reasoning derailment in Sec.2.3, we present a qualitative analysis of the effectiveness of QuBE to address reasoning derailment. In Table 1, it can be observed that accurately establishing the belief state and generating relevant rationale enables the agent to successfully handle derailment, with lower error rates for QuBE compared to ReAct, on all measured error types. Similarly, in Table 2, QuBE shows lower error rates in identifying relevant and non-relevant terms, compared to ReAct, which is achieved by concisely organizing the belief state and subsequently generating the rationale. These results support our hypothesis, that grounding the rationale on an enhanced belief state empowers the LLM to address reasoning derailment [8].

5.4 Analysis of Computational Efficiency

To assess the computational costs and potential trade-offs of the additional question-answering and rationale generation steps of QuBE, we perform a comparison of the average LLM API calls of Reflexion and QuBE. In SE, both the Reflexion baseline and QuBE use 3 LLM API calls. For each, the input token length is $\sim$250 tokens + query tokens + documents' tokens, and $\sim$450 tokens + query tokens + documents' tokens, respectively. Since the input context is dominated by the documents, the difference in $\sim$200 tokens is negligible, indicating that in SE, QuBE's steps are more effective, at nearly identical cost to Reflexion. In GE, the Reflexion baseline and QuBE use on average 174.0 and 134.5 LLM API calls per task, cumulative over all trials, respectively. While QuBE calls in GE incur additional LLM API calls, we find that the overall efficiency is improved, because unrectified reasoning errors in the baseline agent result in derailment causes wasted API calls, leading to the higher number of calls for Reflexion.

5.5 Generalizability to Smaller LLMs

To study the generalizability of QuBE to smaller and open-source LLMs, we conduct additional experiments using Llama3-8B and Mistral-7B[9]. As shown in Tables 6 and 7, we find that derailment poses an equally important challenge for agents based on smaller LLMs. The results indicate that the derailment phenomenon occurs across model sizes, impacting performance significantly. This is particularly evident in GE, where we observe that a baseline Reflexion agent suffers derailment in nearly all cases. A likely reason is that agents based on smaller LLMs become more derailment-prone due to the longer agent trajectories, highlighting the importance of addressing derailment in such settings. In contrast, the results of QuBE show that it is consistently effective at handling derailment even in smaller LLMs, confirming the

[8] We provide full qualitative examples in Appendix A.8

[9] Model details are provided in Appendix A.4.

generalizability of its effectiveness across model scales.

6 Related Work

6.1 Language Models as Agents

Language-model based agents have a long and extensive history, with a notable early example in text-based games, as an effective testbed for agent interaction (Hausknecht et al., 2019; Côté et al., 2018; Ammanabrolu and Riedl, 2019; Ammanabrolu and Hausknecht, 2020; Yao et al., 2020). More recently, there has been growing interest in using text-based games to test agents' abilities in more complex environments and sophisticated scenarios including virtual home environments, interactive scientific reasoning, and web navigation (Shridhar et al., 2021; Wang et al., 2022b; Yao et al., 2022).

A parallel development has been the significant advances in agents based on large language models (LLMs), with a key driver being the combination of reasoning with action. ReAct (Yao et al., 2023), exemplifies this agentic approach, based on prompts interleaving rationales with actions, with the former facilitating high-level reasoning and planning for guiding the agent's actions. Building on ReAct, Reflexion (Shinn et al., 2023) incorporates a persistent memory to store condensed insights over trials, enabling iterative improvement of the agent. Recent works have further extended ReAct, by learning from a pool of success and failure experiences (Zhao et al., 2024), and recursively decomposing tasks into sub-tasks (Prasad et al., 2024). On the game Minecraft, Voyager (Wang et al., 2024a) enhances LLM's planning through a combination of execution error incorporation and self-verification, augmented by retrieving code-based skills from a skill library. Finally, LLM agents are increasingly utilized in realistic and complex scenarios, including task automation on real-world websites (Gur et al., 2024), and enterprise settings requiring the use APIs and tools (Muthusamy et al., 2023).

While such existing works on LLM agents focus primarily on driving performance improvements, we diagnose and tackle a fundamental challenge facing LLM agents, the partial observability of interactive environments.

6.2 Reasoning in Large Language Models

LLMs guided by human-demonstrated rationales, such as Chain of Thoughts (Wei et al., 2022), Least-to-Most prompting (Zhou et al., 2023), have been proven effective in reasoning tasks. Nevertheless, such rationalizing strategies often fail to generalize to unseen, complex scenarios which diverge from the few-shot demonstrations (Zhou et al., 2023).

To improve the accuracy and explainability of LLM reasoning, Lyu et al. (2023) propose Faithful CoT, which uses a deterministic solver for solving intermediate steps in math word problem and question answering tasks. Creswell and Shanahan (2022) propose to finetune a language model to better ensure the logical validity of LLM reasoning, in logic and question answering tasks. Radhakrishnan et al. (2023) propose reasoning by decomposing questions into subquestions, for question answering tasks.

While our work also pursues reliable and accurate LLM reasoning, our proposed solution of belief state construction and grounded rationale generation solves the unique problem of reasoning derailment in LLM agents.

7 Conclusion

In this work, we studied deployment of agentic LLMs within complex interactive environments. Due to the challenge of partial observability, existing LLMs produce reasoning errors, which are even more pronounced in agentic LLMs requiring multiple interactions. To tackle this, we propose QuBE, which constructs a belief state using question-answering, as an estimate of the unobserved true environment state. Using the constructed belief state, QuBE performs rationale generation to resolve noise and uncertainty in the raw partial observations and enhance the LLM agent's action policy. Experimented on search and game environment scenarios, QuBE reduces reasoning errors, and improves task performance, outperforming strong baselines.

Limitations

A limitation of our work is the variable nature of interactive environments in which agents are deployed for complex tasks, which can lead to different challenges depending on the environment.

To address this, we evaluated QuBE on two diverse partial observability settings, both a canonical text-based game setting as well as an interactive RAG scenario. Future work should expand the exploration of the reasoning derailment phenomenon to wider arrays of interactive scenarios, and promising directions for solutions include combining more powerful faithful reasoning approaches with agentic LLMs.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2021-0-0268, Artificial Intelligence Innovation Hub (Artificial Intelligence Institute, Seoul National University)], National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2024-00414981), and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)]

References

AI@Meta. 2024. Llama 3 model card.

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. In International Conference on Learning Representations.

Prithviraj Ammanabrolu and Mark Riedl. 2019. Playing text-adventure games with graph-based deep reinforcement learning. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3557–3565, Minneapolis, Minnesota. Association for Computational Linguistics.

Prithviraj Ammanabrolu, Ethan Tien, Matthew Hausknecht, and Mark O. Riedl. 2020. How to avoid being eaten by a grue: Structured exploration strategies for textual worlds. CoRR, abs/2006.07409.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In Proceedings of the 24th International Conference on Machine Learning, pages 33–40.

Benjamin Börschinger and Mark Johnson. 2011. A particle filter algorithm for Bayesian wordsegmentation. In Proceedings of the Australasian Language Technology Association Workshop 2011, pages 10–18, Canberra, Australia.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. In Workshop on Computer Games, pages 41–75. Springer.

Antonia Creswell and Murray Shanahan. 2022. Faithful reasoning using large language models.

Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. CoRR, abs/1910.10687.

Luca Gioacchini, Giuseppe Siracusano, Davide Sanvito, Kiril Gashteovski, David Friede, Roberto Bifulco, and Carolin Lawrence. 2024. AgentQuest: A modular benchmark framework to measure progress and improve LLM agents. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: System Demonstrations), pages 185–193, Mexico City, Mexico. Association for Computational Linguistics.

James Goodman, Andreas Vlachos, and Jason Naradowsky. 2016. Noise reduction and targeted exploration in imitation learning for Abstract Meaning Representation parsing. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1–11, Berlin, Germany. Association for Computational Linguistics.

Izzeddin Gur, Hiroki Furuta, Austin V Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2024. A real-world webagent with planning, long context understanding, and program synthesis. In The Twelfth International Conference on Learning Representations.

Mary Harper. 2014. Learning from 26 languages: Program management and science in the babel program. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, page 1, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Matthew Hausknecht, Prithviraj Ammanabrolu, Côté Marc-Alexandre, and Yuan Xingdi. 2019. Interactive fiction games: A colossal adventure. CoRR, abs/1909.05398.

Matthew J. Hausknecht and Peter Stone. 2015. Deep recurrent q-learning for partially observable mdps. CoRR, abs/1507.06527.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. 2022. Inner monologue: Embodied reasoning through planning with language models. In arXiv preprint arXiv:2207.05608.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. ACM Trans. Inf. Syst., 20(4):422–446.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Jihyuk Kim, Minsoo Kim, Joonsuk Park, and Seungwon Hwang. 2023. Relevance-assisted generation for robust zero-shot retrieval. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 723–731, Singapore. Association for Computational Linguistics.

Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2022. Pseudo relevance feedback with deep language models and dense retrievers: Successes and pitfalls.

Qinghua Liu, Csaba Szepesvari, and Chi Jin. 2022. Sample-efficient reinforcement learning of partially observable markov games. In Advances in Neural Information Processing Systems.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 2024. Agentbench: Evaluating LLMs as agents. In The Twelfth International Conference on Learning Representations.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. arXiv preprint arXiv:2301.13379.

Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning passage impacts for inverted indexes. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1723–1727.

Vinod Muthusamy, Yara Rizk, Kiran Kate, Praveen Venkateswaran, Vatche Isahagian, Ashu Gulati, and Parijat Dube. 2023. Towards large language model-based personal agents in the enterprise: Current trends and open problems. In Findings of the Association for Computational Linguistics: EMNLP 2023, pages 6909–6921, Singapore. Association for Computational Linguistics.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Marco Pleines, Matthias Pallasch, Frank Zimmer, and Mike Preuss. 2023. Memory gym: Partially observable challenges to memory-based agents. In The Eleventh International Conference on Learning Representations.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. ADaPT: As-needed decomposition and planning with language models. In Findings of the Association for Computational Linguistics: NAACL 2024, pages 4226–4252, Mexico City, Mexico. Association for Computational Linguistics.

Ansh Radhakrishnan, Karina Nguyen, Anna Chen, Carol Chen, Carson Denison, Danny Hernandez, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Sam McCandlish, Sheer El Showk, Tamera Lanham, Tim Maxwell, Venkatesa Chandrasekaran, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. Question decomposition improves the faithfulness of model-generated reasoning.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. Computing Research Repository, arXiv:1503.06733. Version 2.

Soo Young Rieh et al. 2006. Analysis of multiple query reformulations on the web: The interactive information retrieval context. Information Processing & Management, 42(3):751–768.

Joseph John Rocchio Jr. 1971. Relevance feedback in information retrieval. The SMART retrieval system: experiments in automatic document processing.

Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. arXiv preprint arXiv:2303.11366.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. {ALFW}orld: Aligning text and embodied environments for interactive learning. In International Conference on Learning Representations.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. 2023. Cognitive architectures for language agents.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024a. Voyager: An open-ended embodied agent with large language models. Transactions on Machine Learning Research.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022a. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2345–2360, Seattle, United States. Association for Computational Linguistics.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024b. A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6).

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022b. ScienceWorld: Is your agent smarter than a 5th grader? In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 11279–11298, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2023. Colbert-prf: Semantic pseudo-relevance feedback for dense passage and document retrieval. ACM Transactions on the Web, 17(1):1–39.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Tie-Yan Liu, and Wei Chen. 2013. A theoretical analysis of ndcg type ranking measures. Journal of Machine Learning Research, 30.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In Advances in Neural Information Processing Systems, volume 35, pages 24824–24837. Curran Associates, Inc.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. In Advances in Neural Information Processing Systems, volume 35, pages 20744–20757. Curran Associates, Inc.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 8736–8754, Online. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In International Conference on Learning Representations (ICLR).

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. In Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada, pages 19632–19642. AAAI Press.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In The Eleventh International Conference on Learning Representations.

A Appendix

A.1 Dataset Details: BEIR

The statistics for the BEIR datasets are available in Table 5, including the number of queries evaluated, number of documents, and average query length.

A.2 Dataset Details: ALFWorld

ALFWorld consists of 134 tasks, comprised of 6 different task types in a virtual household environment: Pick (24), Clean (18), Heat (31), Cool (23), Examine (21), and Pick two (17).

A.3 LLM API Details

We access LLMs through the OpenAI API. To manage API costs, we use `code-davinci-002` (Codex) via free researcher access, for the agent LLM. As Codex is token-rate limited, for QuBE components, we use `gpt-3.5-turbo-0301`, due to its low cost. We note that QuBE is agnostic to the choice of in-context-learning LLM, and does not rely on any zero-shot/chat capabilities.

A.4 Details of Smaller LLMs

For experiments with smaller LLMs, we leverage Llama3-8B (AI@Meta, 2024)[10] and Mistral-

[10]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

	NFCorpus	TREC-COVID (v2)	Webis-Touche2020	FiQA	SCIDOCS	SciFact
Total # of Test Query	323	50	49	648	1000	300
Total # of Documents	3.6k	171.3k	382.5k	57.6k	25.6k	5.2k
Avg. Word Length of Query	3.3	10.6	6.55	10.77	9.38	12.37

Table 5: Number of queries in BEIR datasets studied.

Model	Trial 0	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9
Llama3-8B										
Reflexion	0%	1%	2%	2%	2%	2%	2%	2%	2%	2%
QuBE (Ours)	**7%**	**10%**	**18%**	**24%**	**27%**	**28%**	**30%**	**33%**	**37%**	**39%**
Mistral-7B										
Reflexion	2%	3%	4%	7%	7%	10%	10%	10%	10%	10%
QuBE (Ours)	**3%**	**8%**	**10%**	**10%**	**13%**	**13%**	**16%**	**18%**	**20%**	**20%**

Table 6: Experiment results with smaller language models on ALFWorld, using task success rate as the evaluation metric. The best results are highlighted in **bold**.

	NFCorpus	TREC-COVID (v2)	Touché	FiQA	SCIDOCS	SciFact	Average
Llama3-8B							
ReAct/Reflexion	$35.2_{\pm 0.1}$	$77.1_{\pm 0.5}$	$26.9_{\pm 0.3}$	$35.6_{\pm 0.0}$	$16.1_{\pm 0.1}$	$69.4_{\pm 0.1}$	$43.4_{\pm 0.0}$
QuBE (Ours)	$\mathbf{36.4}_{\pm 2.1}$	$\mathbf{78.0}_{\pm 1.2}$	$\mathbf{28.5}_{\pm 0.3}$	$\mathbf{35.9}_{\pm 0.4}$	$\mathbf{16.3}_{\pm 0.1}$	$\mathbf{69.7}_{\pm 0.4}$	$\mathbf{44.1}_{\pm 0.0}$
Mistral-7B							
ReAct/Reflexion	$34.9_{\pm 0.1}$	$\mathbf{76.8}_{\pm 0.4}$	$27.0_{\pm 0.4}$	$35.6_{\pm 0.1}$	$15.9_{\pm 0.1}$	$68.8_{\pm 0.6}$	$43.2_{\pm 0.0}$
QuBE (Ours)	$\mathbf{35.3}_{\pm 0.2}$	$\mathbf{76.8}_{\pm 1.2}$	$\mathbf{29.3}_{\pm 0.5}$	$\mathbf{36.1}_{\pm 0.2}$	$\mathbf{16.2}_{\pm 0.1}$	$\mathbf{69.6}_{\pm 0.5}$	$\mathbf{43.9}_{\pm 0.0}$

Table 7: Experiment results with smaller language models on BEIR, using nDCG@10 as the evaluation metric. For the LLM-based systems, we conduct 3 runs and report the average and variance of performance. The best results for each dataset are highlighted in **bold**.

7B (Jiang et al., 2023)[11].

A.5 Trigger Mechanism for Efficiency

We can equip QuBE to enhance the efficiency in terms of API calls, by selectively activating QuBE with a derailment detection mechanism. In GE, we leverage the following easily detected signals available from the AlfWorld environment: 1) The agent repeats the same action Ω[12] times in succession 2) issuing an action command to the engine leads to the observation "*Nothing happens*", which is the engine's indicator that the action failed to execute. In SE, since the search engine does not provide explicit success or failure signals, and the number of LLM calls are small, we always activate QuBE.

A.6 QuBE Implementation Details in SE

To obtain keywords from documents, following ColBERT-PRF (Wang et al., 2023), we leverage centroid terms within retrieved documents by performing k-means clustering on term vectors encoded by a BERT encoder[13].

We extract relevant terms and non-relevant terms from top-3 and top-30 documents, respectively, based on the intuition that the top-3 documents are more likely to be relevant to the query compared to top-30 documents. For extracting non-relevant terms, when performing clustering, we assign uniform weights on terms in the top-30 documents. By doing so, the extracted centroid terms, when used as a search query, will likely return non-relevant documents as search results.

In contrast, for extracting relevant terms, we allow each term to contribute differently based on its relevance to the given query. Specifically, we employ weighted k-means clustering, where the weight for each term is set by the product of the relevance score of the term and the document where

[11] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[12] We set Ω to 2, following Shinn et al. (2023).

[13] For the k value, we set the number of relevant terms smaller than that of non-relevant terms, setting k as 10 and 24, for relevant and non-relevant terms, respectively.

the term appears. For the document score, we leverage the relevance score returned by the search engine, ColBERT. For the term score, we employ DeepImpact (Mallia et al., 2021), which produces an impact score of each term within a document in identifying the document as relevant. We use DeepImpact due to its empirical superiority reported in Dai and Callan (2019), though others are equally applicable.

A.7 Prompts

A.7.1 Game Engine - ALFWorld

ReAct Prompt For ReAct, we use the original prompts which can be found in Yao et al. (2023).

QuBE Prompt For QuBE, we use the prompts shown below. All demonstrations used are identical to the original ones utilized in ReAct.

QuBE (Belief State Generation) Prompt

System:
You are an agent acting in a virtual household environment, operating by following goals at any given time.

Input:
As a world model, your job is to accurately provide information about the current environment.
Here is your current trajectory: {Current Trajectory}.
Answer the following questions. Here are the questions:
1) Where am I now?
2) What is my inventory?
3) Which receptacles are available?
4) Which receptacles do not need to be checked again?
For each question, repeat the question itself, then answer it.
If you do not know an answer, answer I don't know.

QuBE (Rationale Generation) Prompt

System:
You are an agent acting in a virtual household environment, operating by following goals at any given time.

Input:
You are solving a task in a virtual household environment.

Here are demonstrations from randomly selected successful instances of a similar environment, followed by the current task: {Demonstrations and Current Trajectory}.
This is what you know so far: {Belief State}
Your next trajectory originally was {Failed Trajectory Portion}, but it led to a failure. Write a new thought that is more likely to lead to a successful trajectory, and only base your answer on what you know.

A.7.2 Search Engine - BEIR

ReAct Think Prompt We present the prompt used for ReAct, which interleaves reasoning (i.e., think about the query and documents) and action (i.e., generating missing query terms), denoted by "Think:" and "Expand[Query]:" in the prompt, respectively. For clarity, we divide each interaction step in the agent's trajectory, into its own box.

ReAct Instruction and Fewshot example Prompt

System: You are an intelligent assistant that can help users to write better queries.

Input: ▷ Task instruction with examples
Solve a query expansion task with interleaving Observation, Thought, Action steps.
Observe[Query, Doc]: observe the query and document.
Think: reason about the current situation.
Expand[query]: extracts terms from given document to expand query.
Here are two examples. ▷ Few-shot examples
###
{Example1}
###
{Example2}
Do you understand the task?

ReAct: Yes.

ReAct Think Prompt

Input:
Observe[Query]: {query}
Observe[Doc 1]: {top-1 ranked document}
Observe[Doc 2]: {top-2 ranked document}
Observe[Doc 3]: {top-3 ranked document}

ReAct:
Think: {thought}

ReAct Action (Query Expansion) Prompt

Input:
Extract relevant terms from the documents, that are missing in the original query.
Return only expansion terms, do not give any other explanations.
Total number of terms to be extracted is at most 10.

ReAct:
Expand[Query]: { [expansion term 1, . . .] }

QuBE Prompt We present the prompt used for QuBE, which builds upon the prompt for ReAct with minor modifications. Unlike GE, SE utilizes Q-Env without LLM, directly using the environment-attached tool, as described in Sec.A.6. We show the prompt for rationale generation below, highlighting the portions related to the belief state in orange. For clarity, we divide each interaction step in the agent's trajectory, into its own box.

QuBE Instruction and Fewshot example Prompt

System: You are an intelligent assistant that can help users to write better queries.

Input: ▷ Task instruction with examples
Solve a query expansion task with interleaving observation, thought, action steps.
Observe[Query, Doc, Relevant Terms in Doc, Non-relevant Terms]: read the query, the top-k documents retrieved by the search engine, and the retrieved document keywords.
Think[Intent]: think about the intent of the query. What would be the purpose of this query?
Expand[Query]: extract the terms from given documents and document keywords to expand the query.
Here are two examples. ▷ Few-shot examples
###
{Example1}
###
{Example2}
Do you understand the task?

QuBE: Yes.

QuBE (Rationale Generation) Prompt

Input:
Now here is the actual task.
###
Observe[Query]: {query}
Observe[Doc 1]: {top-1 ranked document}
Observe[Relevant Terms in Doc 1]: {relevant terms in document 1}
Observe[Doc 2]: {top-2 ranked document}
Observe[Relevant Terms in Doc 2]: {relevant terms in document 2}
Observe[Doc 3]: {top-3 ranked document}
Observe[Relevant Terms in Doc 3]: {relevant terms in document 3}
Observe[Non-relevant terms]: {Non-relevant terms from document 1-3}

QuBE:
Think[Intent]: {query intent}

QuBE Action (Query Expansion) Prompt

Input:
Extract relevant terms from the documents, that are missing in the original query.
Return only expansion terms, do not give any other explanations.
Total number of terms to be extracted is at most 10.

QuBE:
Expand[Query]: { [expansion term 1, . . .] }

A.8 Qualitative Examples

A.8.1 Game Engine - ALFWorld

In this example, the baseline ReAct agent fails to complete the task, "clean some soapbar and put it in cabinet", due to reasoning derailment, in which it continuously tries to take a soapbar 1 from various locations. QuBE repairs this error, by constructing a belief state through question answering, which aids its subsequent reasoning, guiding it to check other locations. The soapbar 1 is found on the countertop 1, which eventually leads to success on the task.

ReAct

Task

Your task is to: clean some soapbar and put it in cabinet.

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1.

> **think**: To solve the task, I need to find and take a soapbar, then clean it with sinkbasin, then put it in cabinet. OK.

> **think**: First I need to find a soapbar. A soapbar is more likely to appear in sinkbasin (1-2), garbagecan (1), cabinet (1-4), toilet (1), toiletpaperhanger (1), towelholder (1), handtowelholder (1-2). I can check one by one, starting with sinkbasin 1.
OK.

...

> open cabinet 3
You open the cabinet 3. The cabinet 3 is open. In it, you see nothing.
> go to cabinet 4
The cabinet 4 is closed.
> open cabinet 4
You open the cabinet 4. The cabinet 4 is open. In it, you see a candle 1, and a cloth 2.
> go to toilet 1
On the toilet 1, you see a spraybottle 1, and a toiletpaper 2.
> go to toiletpaperhanger 1
On the toiletpaperhanger 1, you see a toiletpaper 1.
> go to towelholder 1
On the towelholder 1, you see a towel 1.
> go to handtowelholder 1
On the handtowelholder 1, you see a handtowel 1.
> go to handtowelholder 2
On the handtowelholder 2, you see a handtowel 2.
> think: Now I find a soapbar (1). Next, I need to take it.
OK.
> take soapbar 1 from handtowelholder 2
Nothing happens.
> take soapbar 1 from handtowelholder 1
Nothing happens.
> take soapbar 1 from towelholder 1
Nothing happens.
> take soapbar 1 from toiletpaperhanger 1
Nothing happens.
> take soapbar 1 from toilet 1
Nothing happens.
> take soapbar 1 from cabinet 4
Nothing happens.
> take soapbar 1 from garbagecan 1
Nothing happens.
> take soapbar 1 from sinkbasin 2
Nothing happens.
> take soapbar 1 from sinkbasin 1
Nothing happens.
> think: Now I take a soapbar (1). Next, I need to go to sinkbasin (1-2) and clean it.

...

(TASK FAILURE)

QuBE

Task
Your task is to: clean some soapbar and put it in cabinet.

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1.
...
> go to toiletpaperhanger 1
On the toiletpaperhanger 1, you see a toiletpaper 1.
> **think**: Now I find a soapbar (1). Next, I need to take it.
OK.
> take soapbar 1 from toiletpaperhanger 1
Nothing happens.

QuBE [Belief State]
1) Where am I now?
You are facing the toiletpaperhanger 1. Next to it, you see nothing.
2) What is my inventory?
You are not carrying anything.
3) Which receptacles are available?
a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, a towelholder 1
4) Which receptacles do not need to be checked again?
cabinet (1-3), garbagecan (1), handtowelholder (1-2), sinkbasin (1-2), toiletpaperhanger (1), towelholder (1)

QuBE [Rationale]
> **think**: Since I couldn't find a soapbar in the locations I checked, I need to expand my search. A soapbar could also be in/on the countertop (1) or in the cabinet (4). I will check the countertop (1) first.
OK.
> go to countertop 1
On the countertop 1, you see a soapbar 1, and a soapbottle 1.
> **think**: Now I find a soapbar (1). Next, I need to take it.
OK.
> take soapbar 1 from countertop 1
You pick up the soapbar 1 from the countertop 1.
> **think**: Now I take a soapbar (1). Next, I need to go to sinkbasin (1) and clean it.
OK.
> go to sinkbasin 1
On the sinkbasin 1, you see nothing.
> go to sinkbasin 2
On the sinkbasin 2, you see nothing.
> clean soapbar 1 with sinkbasin 2
You clean the soapbar 1 using the sinkbasin 2.
> **think**: Now I clean a soapbar (1). Next, I need to put it in/on cabinet 4.
OK.
> go to cabinet 4
The cabinet 4 is open. In it, you see a candle 1, and a cloth 2.
> put soapbar 1 in/on cabinet 4
You put the soapbar 1 in/on the cabinet 4.
(TASK SUCCESS)

A.8.2 Search Engine - BEIR

In this example, the query is "AMP-activated protein kinase (AMPK) activation increases inflammation-related fibrosis in the lungs.". Compared to the ColBERT baseline, QuBE successfully improves the retrieval results (higher nDCG@10), while ReAct experiences a drop in nDCG@10.

During the reasoning step, QuBE contemplates the query and refines its reasoning based on the environment feedback, which constitutes the retrieved document, and enhanced via the constructed belief state. On the other hand, ReAct focuses on the specifics of the provided documents, reducing focus on the intent of the query. Consequently, QuBE incorporates important information from key terms into its estimate of the query intent, while ReAct expands terms based solely on its contemplation of the documents.

ReAct

[Think]:
It appears that the query is asking about the relationship between AMP-activated protein kinase (AMPK) activation and inflammation-related fibrosis in the lungs. The given documents talk about the role of AMPK in regulating metabolism and suppressing inflammation, and how activation of AMPK inhibits the Janus kinase (JAK)–signal transducer and activator of transcription (STAT) pathway in various cells. Also mentioned are how AMPK activators like metformin and salicylate can enhance the inhibitory phosphorylation of endogenous JAK1 and inhibit STAT3 phosphorylation. Therefore, one possible way to expand the query could be by adding terms related to JAK-STAT pathway and its association with lung fibrosis.

Query Expansion:
[JAK1, STAT, phosphorylation, metformin, salicylate, endogenous, V658F-mutant, fibrosarcoma cells, vascular endothelial cells, inflammatory signaling]

QuBE

[Belief State]:
Relevant Terms in Doc 1: [ser515, ser518, enhanced, jak1, ampk]
Relevant Terms in Doc 2: [ampk, cardiovascular, homeostasis]
Relevant Terms in Doc 3: [ho, ampk, nrf2]
Non-relevant Terms: [fibrosarcoma, show, pharmacological, phosphorylation, signaling, phosphorylated]

[Think]:
The intent of the query is to find information about the relationship between AMP-activated protein kinase (AMPK) and inflammation-related fibrosis in the lungs, and how AMPK activation affects anti-inflammatory signaling and Nrf2/HO-1 signaling in cardiovascular tissues.

Query Expansion:
[inflammation, fibrosis, lungs, Nrf2, HO-1, cardiovascular, anticontractile, anti-inflammatory, energy homeostasis, redox]

A.9 Use of AI Assistants

We used ChatGPT for grammatical corrections.