TreeRare: Syntax Tree-Guided Retrieval and Reasoning for Knowledge-Intensive Question Answering

Anonymous ACL submission

Abstract

In real practice, questions are typically complex and knowledge-intensive, requiring Large Language Models (LLMs) to recognize the multifaceted nature of the question and reason across multiple information sources. Iterative and adaptive retrieval, where LLMs decide *when and what to retrieve* based on their reasoning, has been shown to be a promising approach to resolve complex, knowledge-intensive questions. However, the performance of such retrieval frameworks is limited by the accumulation of reasoning errors and misaligned retrieval results. To overcome these limitations, we propose TreeRare (Syntax **Tree**-Guided **R**etrieval **a**nd **Re**asoning), a framework that utilizes syntax trees to guide information retrieval and reasoning for question answering. Following the principle of compositionality, TreeRare traverses the syntax tree in a bottom-up fashion, and in each node, it generates subcomponent-based queries and retrieves relevant passages to resolve localized uncertainty. A subcomponent question answering module then synthesizes these passages into concise, context-aware evidence. Finally, TreeRare aggregates the evidence across the tree to form a final answer. Experiments across five question answering datasets involving ambiguous or multi-hop reasoning demonstrate that TreeRare achieves substantial improvements over existing state-of-the-art methods.

1 Introduction

Large Language Models (LLMs) (Chowdhery et al., 2023; Achiam et al., 2023) have demonstrated remarkable capabilities across a wide range of natural language processing (NLP) tasks, including text generation and question answering, often achieving strong performance in few-shot or even zero-shot settings without task-specific fine-tuning (Brown et al., 2020). Despite their impressive performance, LLMs generate plausible but factually incorrect
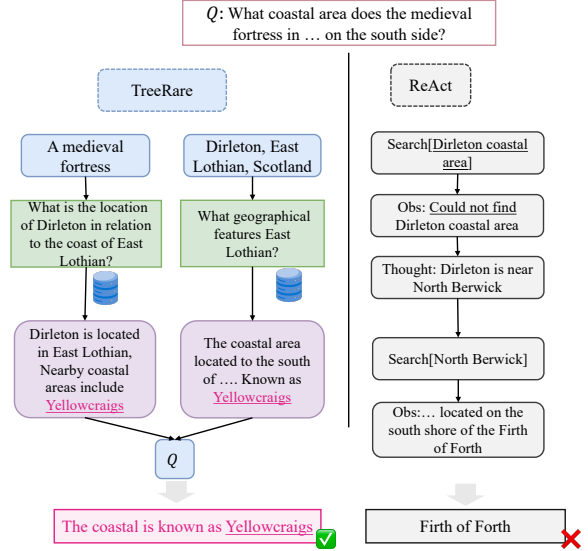

Figure 1: Comparison of TreeRare and ReAct(Yao et al., 2022) on a multihop question. TreeRare decomposes the question into structured sub-questions and retrieves focused evidence. In contrast, ReAct fails to generate useful query and misidentifies the region.

statements due to over-reliance on their parametric knowledge when tackling knowledge-intensive tasks that demand factual accuracy and external grounding (Roberts et al., 2020; Maynez et al., 2020; Chen et al., 2022). To address this issue, existing work has shown that retrieval-augmented generation (RAG) can largely reduce factual hallucination by incorporating LLMs with external knowledge sources (Lewis et al., 2020; Guu et al., 2020).

RAG enhances LLMs by integrating external knowledge retrieval into the generation process. Traditional RAG systems follow a "retrieve-then-read" paradigm, where a retriever selects top-k documents based on similarity metrics, and the LLM generates responses conditioned on retrieved documents (Lewis et al., 2020). While effective for well-posed queries, this paradigm often strug-

gles to answer ambiguous or multihop questions, as similarity-based retrieval may miss relevant evidence. To address this limitation, many works enhance retrieval quality via retrieval judgment, adaptive search mechanisms, or question decomposition to improve retrieval quality and performance of complex knowledge-intensive tasks. (Li et al., 2025; Asai et al., 2024; Tan et al., 2024; Yao et al., 2022) However, as illustrated in Figure 1, these methods often rely on internal reasoning in LLMs, which can cause errors to accumulate across steps. Mistakes made during reasoning or retrieval can result in information that is misaligned with the original intent of the question, leading to noisy inputs and incorrect final answers (Li et al., 2024).

Inspired by the principle of compositionality —*the meanings of complex expressions are constructed from the meanings of the less complex expressions that are their constituents* (Fodor and Lepore, 2002)— we then ask: Can the syntactic structure of complex, knowledge-intensive questions guide effective retrieval and inference toward correct answers? We leverage syntax trees as a basis for question decomposition, since parsing tree has been shown to be effective in capturing the syntactic relations between each phrase in a sentence (Li et al., 2015). Then, we propose a bottom-up traversal of the syntax tree, where each child node is processed first, and its output is used to guide the processing of its parent node. Additionally, observing that LLMs frequently fail to detect ambiguity or knowledge gaps present within sub-phrases (Piryani et al., 2024; Kim et al., 2024), we provide LLMs at each node with information from the child nodes to formulate queries that resolve the associated sub-phrase uncertainty. Furthermore, recognizing that LLM performance degrades when conditioned on long or noisy inputs (Liu et al., 2024b; Xu et al., 2024), we introduce a subcomponent question answering that synthesize the retrieved context into concise, phrase-relevant evidence.

Combining these modules, we propose syntax tree-guided retrieval and reasoning (**TreeRare**). TreeRare incrementally retrieves and resolves subcomponents of a question in accordance with its syntax structure. As illustrated in Figure 2, TreeRare traverses the syntax tree in a bottom-up manner, resolving the uncertainty at each node through a two-stage process, starting with subcomponent-based retrieval, followed by subcomponent question answering. Upon completing the traversal,

TreeRare constructs targeted, comprehensive evidence for each sub-phrase in the parsing tree and aggregates these evidence across nodes to generate a final answer.

Our contributions are as follows: (1) We propose TreeRare to handle complex, knowledge-intensive questions, which enhances LLMs' performance by interleaving retrieval with reasoning over the syntax tree. (2) We introduce retrieval-only counterpart, Tree-Retrieval, effectively improving the retrieval quality without involving any LLM reasoning. (3) We perform experiments across multiple multihop and ambiguous question answering (QA) benchmarks for three LLM backbones. On multihop QA benchmarks, TreeRare achieves an average relative improvement up to 17.8%. For ambiguous QA, TreeRare yields an average improvement of 23.7% across various evaluation metrics.

2 Related Work

2.1 LLMs for Reasoning

Significant efforts have been dedicated to enhance the reasoning capabilities of LLMs during the inference phase. Chain-of-prompting (CoT) (Wei et al., 2022) prompting introduces intermediate reasoning steps between the initial query and the final answer, thereby improving performance on complex tasks. Building upon this, self-consistency (SC) (Wang et al., 2022) generates multiple reasoning paths and employ majority voting to select the final answer; self-verification (Shinn et al., 2023) prompts LLMs to reflect on their outputs and iteratively refine them through feedback. Additionally, Tree-of-Thought (ToT) and Reasoning via Planning (RAP) prompting (Yao et al., 2023; Hao et al., 2023) extend the CoT approach. These methods further enhance LLMs' reasoning abilities by exploring multiple reasoning paths with different tree search algorithm. Above approaches rely solely on the internal reasoning traces of LLMs and lack structural guidance, such as syntax tree included in TreeRare.

2.2 Retrieval Augmented Generation

Common RAG systems proceed in a retriever-then-read paradigm, where it first retrieves relevant documents based on the user's query using either sparse or dense retrieval and then takes the retrieved information in addition to the question as input to generate final answer (Khandelwal et al., 2020; Shi et al., 2024; Borgeaud et al., 2022). This paradigm outper-
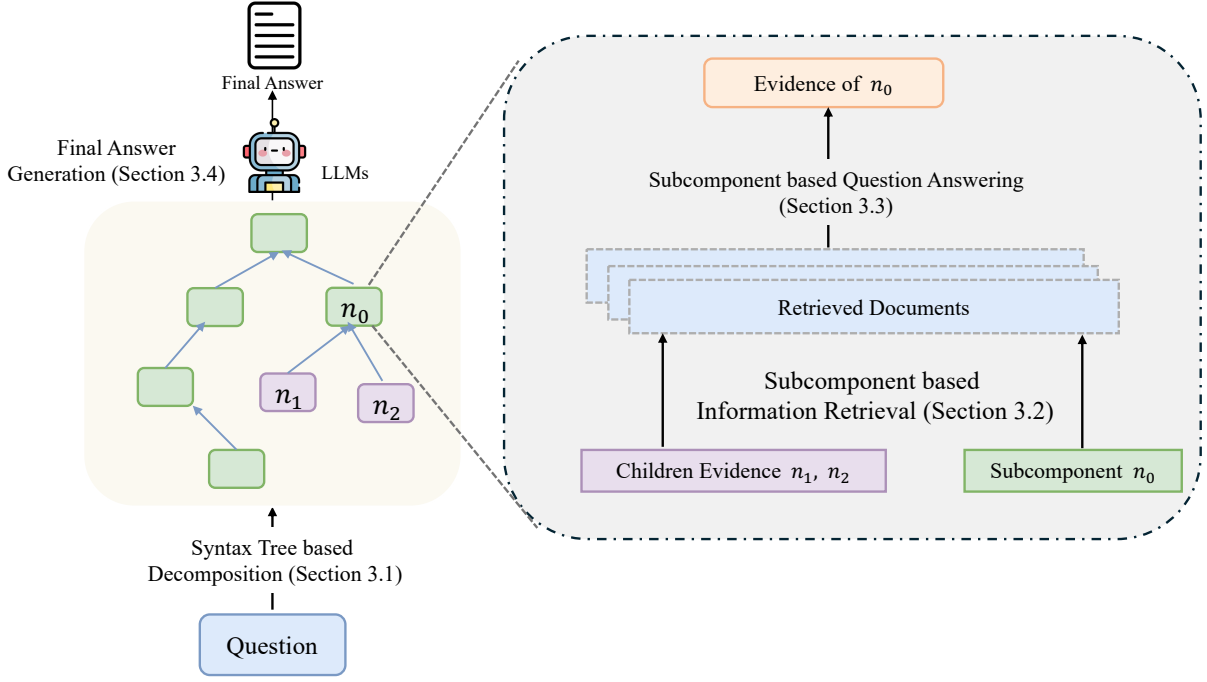
Figure 2: **Overview of the TreeRare framework**. Given a knowledge-intensive question, TreeRare first decomposes it into a syntax tree. It then traverses the tree in a bottom-up manner. At each node, the framework generates subcomponent-based queries conditioned on evidence from its child nodes and the current phrase. These queries guide document retrieval, and a subcomponent question answering module extracts evidence related to the generated queries. Finally, all node-level evidence is aggregated to produce the final answer.

forms regular LLMs, especially for the knowledge-intensive single-hop questions. In order to answer these single-hop questions, the needed information is evident from the question itself, such that a one-time retrieval can find the documents that contain the answer (Trivedi et al., 2022).

However, this paradigm is inadequate for complex, knowledge-intensive questions, such as multihop or ambiguous questions. Standard RAG systems retrieve documents based solely on the original query, without accounting for the evolving information needs of further reasoning steps (Talmor and Berant, 2018; Amouyal et al., 2023). In the case of ambiguous questions, such approaches run the risk of making an early commitment to a single interpretation and potentially overlooking alternative meanings necessary for accurate comprehension. (Lee et al., 2024; Gao et al., 2021). Notably, LLMs have demonstrated their strong ability in decomposing complex tasks into different sub queries to facilitate its performance on complex, knowledge-intensive tasks (Drozdov et al., 2022; Khot et al., 2023; Dua et al., 2022). Several approaches incorporate LLMs' generating contents, including intermediate reasoning steps or heuristic answers, to guide retrieval (He et al., 2022; Trivedi et al., 2023; Tan et al., 2024). Similar to how humans iteratively resolve complex questions by identifying salient information gaps, querying on search engine, and progressively narrowing uncertainty until reaching a final answer, further work has applied LLM agents for information retrieval by leveraging LLMs' reasoning capabilities to dynamically determine both *when and what to retrieve* (Jiang et al., 2023; Li et al., 2025; Yao et al., 2022). While these adaptive RAGs depend on reasoning traces or heuristic decision-making, TreeRare utilizes the syntactic structure of the question to guide the retrieval and reasoning.

3 TreeRare: Syntax Tree-Guided Retrieval and Reasoning

In this section, we give a detailed explanation of TreeRare. Our approach is built on three key intuitions: (1) answering complex reasoning questions requires addressing uncertainty within each phrase of a question; (2) the uncertainty associated with a phrase depends on clarifying its constituent subphrases, as understanding the parts is necessary to resolve the whole; and (3) effective retrieval should target diverse and fine-grained evidence that aligns with the phrase.

3

3.1 Syntax Tree-Based Decomposition

Given an input question Q, TreeRare converts it into its corresponding syntax tree. The resulting tree comprises a set of nodes N, where each node $n \in N$ corresponds to a sub-phrase s_n and is associated with a set of child nodes C_n. Specifically, s_n spans all tokens dominated by node n in the parse tree. These syntax structures make explicit the constituency-based or dependency-based relationships between phrases, displaying how meaning is composed from sub-units of the question. TreeRare interprets each sub-phrase as a constraint of the question and reasons over them incrementally. Following principles of compositional semantics (Fodor and Lepore, 2002), the reasoning process proceeds in a bottom-up tree traversal, ensuring that node n is processed only after resolving all its children C_n. Such ordering provides a basis for constructing a stepwise reasoning path grounded in syntax decomposition.

3.2 Subcomponent-Based Information Retrieval

At each node n in the syntax tree, TreeRare generates a set of subcomponent-based queries to address the latent knowledge gaps associated with its sub-phrase s_n. Complicated uncertainty often emerges not at the level of individual sub-phrases, but from their interaction. For instance, through the composition of sub-phrases, novel entities may emerge, and modifiers can introduce ambiguity or context-sensitive reinterpretations. Therefore, even if each child node contributes reliable evidence, E_c, this evidence alone may be insufficient to resolve the uncertainty within s_n. To bridge these gaps, TreeRare prompts the LLM to generate multiple subcomponent-based queries that are conditioned on global question, local sub-phrase and evidence from its children. Formally, the query set is constructed as: $Q_n = QG(Q, s_n, \{E_c | c \in C_n\})$, where QG is a function that prompts LLMs to generate a set of simple queries. This formulation ensures two key properties of generated queries:

- **Compositional Grounding**: Queries are crafted to resolve the information gap that emerges from interactions between sub-phrases or from novel information introduced through their composition.

- **Explicit Reasoning**: Queries function as an intermediate reasoning step to resolve s_n, which eventually leads to coherent inference toward the full question.

Following the generation of subcomponent queries, TreeRare initiates a targeted retrieval procedure aimed at acquiring external textual documents that directly resolve the subcomponent-based queries, Q_n. Each query $q \in Q_n$ guides the retriever over a large corpus to obtain a set of top-ranked documents $d_{n,q}$. The complete retrieved context for node n is then defined as $D_n = \bigcup_{q \in Q_n} d_{n,q}$.

3.3 Subcomponent Question Answering

Naively combining all retrieved documents leads to excessive input length and noise, especially harmful under the "Lost-in-the-Middle" effect in LLMs (Liu et al., 2024b). To mitigate this effect, TreeRare introduces subcomponent question answering module. It processes the retrieved content to retain information that is salient to sub-phrase s_n. At the same time, it tries to address remaining reasoning gaps that arise when integrating evidence from the subcomponents. These may include contradictions, underspecified relationships, or missing inferences needed to represent the full meaning of s_n from its children. Formally, given generated queries Q_n and retrieved documents D_n, the LLM is instructed to produce a concise set of answers. $E_n = SAG(Q_n, D_n)$, where SAG is a subcomponent answer generation function that resolves Q_n based on retrieved documents D_n.

3.4 Final Answer Generation

Once each node n in the syntax tree has resolved its local uncertainty with a set of evidence E_n, TreeRare advances to the final synthesis stage. To produce the final answer A, we prompt the LLM with the full set of node-level evidence $\{E_n | n \in N\}$ and original question Q. The model is guided to synthesize these into a unified response that addresses uncertainties across all sub-phrases: $A = FAG\{Q, \{E_n | n \in N\}\}$, where FAG is a final answer generation prompting function. This phase is responsible for aggregating the distributed, fine-grained inferences across the entire tree into a coherent answer to the original question Q. It enables the detection and reconciliation of inconsistencies or conflicting signals that may arise between different pieces of evidence. These inconsistencies can lead to multiple, potentially conflicting answers. Additionally, it ensures logical coherence across the whole tree, validating that intermediate infer-

ences collectively support a consistent global reasoning path.

4 Experiments

4.1 Experiment Setup

Datasets. We assess our method on five knowledge-intensive question answering benchmarks that challenge LLMs with multi-step reasoning and ambiguities. For each dataset, we run experiment on 500 randomly sampled questions. We analyze three multihop question answering datasets: (1) **HotpotQA** (Yang et al., 2018), which contains questions requiring reasoning over multiple supporting paragraphs; (2) **2WikiMultiHopQA (2WikiMQA)** (Ho et al., 2020), which consists of entity-centric questions that necessitate combining information from two or more distinct Wikipedia articles; and (3) **MuSiQue** (Trivedi et al., 2022), which features complex questions composed from simple single-hop questions. We also evaluate performance on ambiguous question answering using two datasets: (1) **AmbigDocQA** (Lee et al., 2024), which contains questions involving ambiguous mentions that may refer to multiple distinct entities, each associated with a different valid answer; and (2) **ASQA** (Stelmakh et al., 2022), which contains questions characterized by various types of multifacetedness.

Evaluation Metrics. We employ COVER-EM (Rosset et al., 2021), which assesses whether the generated answer includes the ground truth answer to evaluate multihop questions and ASQA datasets. Following (Stelmakh et al., 2022), we also use Disambig-F1 (Dis-F1) to evaluate performance on the ASQA dataset. For AmbigDocQA, we follow the standard evaluation framework introduced by Lee et al. (2024), using Answer Recall (AR) and Entity Recall (ER) as performance metrics.

Baselines. We evaluate TreeRare against a comprehensive suite of baselines that represent key prompting and planning paradigms under a unified retriever backbone (BM25 (Robertson et al., 2009)). (1) **Zero-shot and few-shot prompting** is introduced by Brown et al. (2020). These serve as foundational setups without any intermediate reasoning steps. (2) **Chain-of-Thought prompting** (CoT) (Wei et al., 2022)

encourages step-by-step reasoning by appending an instruction such as "Let's think step by step" to the input. (3) **Self-Consistency** (SC) (Wang et al., 2022) samples multiple reasoning paths and selects the final answer via majority voting. (4) **Tree-of-Thoughts** (ToT) (Yao et al., 2023) explores multiple structured reasoning trajectories using tree-based search and pruning strategies. (5) **ReAct** (Yao et al., 2022) interleaves reasoning and retrieval by prompting the model to decide dynamically when and what to retrieve. We adapt ReAct to use BM25 instead of web-based tools, denoted as ReAct*. Details of the baseline implementations are included in Appendix B.

Implementation details. We conduct experiments on three backbone LLMs: GPT-4o-mini (OpenAI, 2024), LLaMA3.3-70B (AI, 2024), and DeepSeek-V3 (Liu et al., 2024a). We implement two variants of TreeRare by adopting two different syntactic formalisms: TreeRare (DT), which leverages dependency trees (Culotta and Sorensen, 2004) to capture head-dependent relations, and TreeRare (CT), which utilizes constituency trees (LANGACKER, 1997) to reflect hierarchical phrase structures. We use the Stanza (Qi et al., 2020) toolkit's dependency and constituency parsers to obtain the required syntactic representations for our framework. For each term-specific query, we employ BM25 (Robertson et al., 2009) to retrieve the top fifteen relevant paragraphs. More specific details of TreeRare are included in Appendix A.

4.2 Results

Table 1 presents the evaluation results of TreeRare against competitive baselines across three LLM backbones. In general, TreeRare leads or closely matches top-performing baselines in both multihop and ambiguous QA tasks, demonstrating the effectiveness of TreeRare in handling complex, knowledge-intensive QA.

In multihop QA, TreeRare achieves the best or near-best scores in most cases. TreeRare (CT) under DeepSeek-V3 achieves the strongest performance, with an average COVER-EM of 0.515 on multihop QA and a relative improvement of 0.082 over the best-performing baseline. Similar trends are observed under LLaMA3.3-70B and GPT4o-mini. The only exception is 2WikiMQA under GPT4o-mini, where TOT slightly outperforms TreeRare, likely because the extended evidence

Model	Method	HotpotQA COV-EM	MuSiQue COV-EM	2WikiMQA COV-EM	AVG	AmbigDoc AR	AmbigDoc ER	ASQA COV-EM	ASQA Dis-F1
GPT4o-mini	zero-shot	0.459	0.146	0.493	0.366	0.472	0.601	0.319	0.370
	few-shot	0.473	0.151	0.520	0.381	0.409	0.539	0.328	0.423
	COT (zero-shot)	0.466	0.124	0.432	0.340	0.349	0.447	0.327	0.386
	COT (few-shot)	0.482	0.144	0.454	0.360	0.373	0.497	0.328	0.425
	SC (zero-shot)	0.502	0.141	0.546	0.396	0.375	0.486	0.282	0.349
	SC (few-shot)	0.484	0.144	0.437	0.355	0.407	0.542	0.314	0.391
	ReAct	0.454	0.208	0.574	0.412	0.359	0.496	0.286	0.331
	ReAct*	0.461	0.196	0.550	0.402	0.359	0.496	0.293	0.335
	TOT	0.491	0.205	**0.612**	0.436	0.187	0.535	0.255	0.265
	TreeRare (DT)	**0.544**	<u>0.240</u>	0.583	<u>0.457</u>	**0.592**	**0.722**	**0.381**	<u>0.547</u>
	TreeRare (CT)	<u>0.542</u>	**0.264**	<u>0.600</u>	**0.468**	<u>0.545</u>	<u>0.642</u>	<u>0.369</u>	**0.565**
	Rel. Impr.	0.083	0.269	-0.019	0.073	0.254	0.201	0.165	0.329
Llama3.3-70B	zero-shot	0.516	0.134	0.521	0.390	0.476	0.558	0.344	0.478
	few-shot	0.502	0.142	0.575	0.406	0.522	0.603	0.331	0.407
	COT (zero-shot)	0.468	0.152	0.352	0.324	0.404	0.512	0.303	0.364
	COT (few-shot)	0.508	0.164	0.478	0.383	0.419	0.635	0.336	0.410
	SC (zero-shot)	0.530	0.172	0.548	0.417	0.417	0.538	<u>0.345</u>	0.459
	SC (few-shot)	0.532	0.168	0.563	0.421	0.411	0.586	0.331	0.384
	ReAct	0.460	0.200	0.570	0.410	0.231	0.305	0.269	0.329
	ReAct*	0.440	0.190	0.540	0.390	0.258	0.405	0.296	0.385
	TOT	0.404	0.195	<u>0.603</u>	0.401	0.185	0.530	0.283	0.423
	TreeRare (DT)	**0.568**	**0.286**	**0.634**	**0.496**	**0.587**	<u>0.686</u>	0.341	**0.518**
	TreeRare (CT)	<u>0.540</u>	<u>0.244</u>	0.584	<u>0.456</u>	<u>0.568</u>	**0.704**	**0.357**	0.517
	Rel. Impr.	0.068	0.430	0.051	0.178	0.125	0.165	0.038	0.084
Deepseek-V3	zero-shot	0.512	0.146	0.547	0.401	0.521	0.632	0.348	0.451
	few-shot	0.526	0.154	0.550	0.410	0.545	0.661	0.358	0.411
	COT (zero-shot)	0.498	0.142	0.426	0.355	0.405	0.527	0.347	0.416
	COT (few-shot)	0.533	0.163	0.526	0.395	0.426	0.599	0.358	0.419
	SC (zero-shot)	0.513	0.146	0.574	0.411	0.446	0.631	0.362	0.465
	SC (few-shot)	0.524	0.152	0.574	0.417	0.421	0.578	0.352	0.448
	ReAct	0.503	0.252	<u>0.673</u>	0.476	0.266	0.352	0.283	0.308
	ReAct*	0.479	0.264	0.630	0.458	0.284	0.371	0.308	0.477
	TOT	0.505	0.273	0.551	0.443	0.174	0.513	0.214	0.308
	TreeRare (DT)	<u>0.572</u>	**0.280**	0.650	<u>0.501</u>	<u>0.567</u>	<u>0.667</u>	**0.406**	<u>0.558</u>
	TreeRare (CT)	**0.594**	<u>0.278</u>	**0.674**	**0.515**	**0.589**	**0.721**	<u>0.391</u>	**0.566**
	Rel. Impr.	0.114	0.026	0.001	0.082	0.131	0.091	0.122	0.187

Table 1: Performance of TreeRare (CT) TreeRare(DT), Chain-of-Thought (CoT), Self-Consistency (CoT-SC), ReAct, ReAct*, and Tree-of-Thoughts (ToT) across five different QA datasets. AVG indicates the average COV-EM on the three multihop datasets. ReAct* denotes a BM25-based variant of ReAct. **Bold** marks the best performance, and <u>underline</u> denotes the second-best under teh same setting. Rel. Impr. stands for relative improvement over the best baseline in the same setting.

derived from the tree structure exceeds GPT4o-mini's limited reasoning capacity. ReAct performs well with stronger backbones but degrades under smaller models, reflecting its reliance on effective prompt-based reasoning. In contrast, TreeRare demonstrates robust performance across different model scales.

TreeRare shows remarkable gains on Ambig-Doc and ASQA across all backbones. In particular, under DeepSeek-V3, TreeRare (DT) achieves the highest Dis-F1 score of 0.406 and the largest relative improvement in COVER-EM in ASQA. These results demonstrate that grounding retrieval in each sub-phrase and subsequently aggregating the collected information helps uncover signals of different plausible interpretations. Notably, on Am-bigDoc, TreeRare (CT) also achieves the highest AR score of 0.592, outperforming the best baseline

by a substantial margin. This highlights TreeRare's strength in disambiguating entities that share the same name.

5 Analysis

5.1 Tree-Retrieval

To demonstrate the contribution of syntactic decomposition to retrieval quality, we devise Tree-Retrieval that mirrors TreeRare while eliminating all the module that require LLM's reasoning. Specifically, we discard the subcomponent-based query generation module and instead directly utilize the corresponding sub-phrases to retrieve relevant documents from the corpus. Then we employ a reranking model to select the top 15 passages across the sub tree rooted at each node. These top passages serve as a substitute for the evidence generated by subcomponent question answering in

Retriever	HotpotQA COV-EM	MuSiQue COV-EM	2WikiMQA COV-EM	AVG	AmbigDoc AR	ER	ASQA COV-EM	Dis-F1
BM25	0.473	0.151	0.520	0.381	0.409	0.539	0.423	0.328
DPR	0.392	0.132	0.318	0.281	0.343	0.449	0.438	0.343
BM25+DPR	0.456	0.146	0.380	0.327	0.322	0.404	0.480	0.351
Tree-Retrieval (DT)	**0.528**	0.138	**0.562**	**0.409**	0.427	0.581	0.514	**0.361**
Tree-Retrieval (CT)	0.510	**0.156**	0.538	0.401	**0.558**	**0.681**	**0.533**	0.335

Table 2: Performance comparison of Tree-Retrieval and standard retrievers on multihop (HotpotQA, MuSiQue, 2WikiMQA) and ambiguous QA datasets (AmbigDoc, ASQA). The AVG column represents the mean COV-EM across multihop datasets.

TreeRare. Reranking model evaluates the initially retrieved documents using more sophisticated models to better assess their relevance to the subcomponent (Kratzwald et al., 2019). Implementation details and experiments on other backbone models are provided at Appendix D.

As shown in Table 2, Tree-Retrieval consistently surpasses classical retrieval approaches such as BM25, DPR(Karpukhin et al., 2020), and their hybrid combination (BM25+DPR) across all QA datasets. The performance gain shows that incorporating syntactic structure into the retrieval process enhances the relevance of retrieved documents. Additionally, these results suggest that the effectiveness of TreeRare cannot be solely attributed to enhanced downstream reasoning by the LLMs. Rather, a significant portion of its advantage stems from its retrieval stage, which is structurally guided to extract more fine-grained, contextually aligned documents.

5.2 Ablation studies

We conduct a series of systematic ablation experiments to evaluate the importance of each single module in TreeRare. Specifically, we evaluate the impact of (1) subcomponent-based query generation (QG), (2) information retrieval (IR), and (3) subcomponent question answering (SAG). In each setting, we selectively disable one module, and we additionally assess configurations where only a single module is retained. Implementation details of these ablation experiments are provided in the Appendix C.

Table 3 shows that removing any core module from TreeRare leads to a significant performance drop, confirming their complementary importance. First, the largest decreases are observed in removal of information retrieval, indicating that information retrieval is the most essential component for generating the correct answer. Second, the absence of question generation causes moderate perfor-
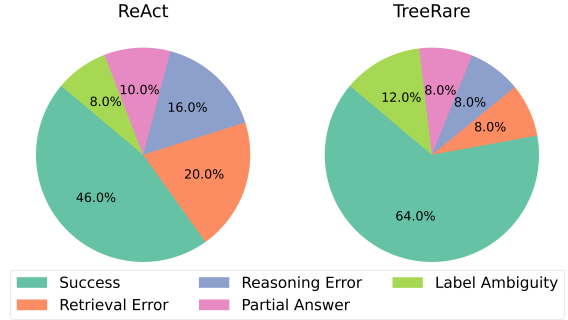

Figure 3: Distribution of outcome types for ReAct and TreeRare on randomly sampled multihop QA.

mance drop (average COVER-EM down to 0.419). Therefore, subcomponent-based queries enhance retrieval relevance and correctness of each reasoning step. Third, removing subcomponent question answering results in substantial degradation, with average COVER-EM decreasing to 0.318. This finding aligns with the "Lost-in-the-Middle" (Liu et al., 2024b) phenomenon. Interestingly, its removal subtly improves performance on AmbigDoc, suggesting that entity-specific cues might be lost during LLM-based filtering.

Among the single-module configurations, the IR-only setting achieves the highest overall performance, confirming the dominant role of information retrieval in knowledge-intensive QA. In contrast, QG-only and COT-only variants perform poorly. As QG-only generates target queries, it outperforms COT-only on ambiguous QA.

5.3 Error Analysis

To better understand the performance differences between ReAct and TreeRare, we conduct a human evaluation comparing their outputs on randomly sampled multihop questions. Each output is manually categorized into one of five distinct outcome types: *Success*, *Retrieval Error*, *Reasoning Error*, *Partial Answer*, and *Label Ambiguity*. The detailed definitions and illustrative examples of each error

Ablation	HotpotQA COV-EM	MuSiQue COV-EM	2WikiMQA COV-EM	AVG	AmbigDoc AR	AmbigDoc ER	ASQA COV-EM	ASQA Dis-F1
TreeRare (DT) *w/o* QG	0.514	0.214	0.537	0.422	0.624	0.797	0.337	0.454
TreeRare (DT) *w/o* SAG	0.514	0.155	0.472	0.380	0.589	0.766	0.360	0.485
TreeRare (DT) *w/o* IR	0.431	0.156	0.366	0.318	0.244	0.262	0.285	0.403
IR Only	0.450	0.146	0.490	0.362	0.472	0.601	0.319	0.371
QG Only	0.352	0.124	0.448	0.308	0.183	0.197	0.248	0.350
COT Only	0.373	0.146	0.334	0.284	0.172	0.186	0.284	0.333

Table 3: Ablation study on TreeRare based on Dependency Tree where the backbone model is GPT4o-mini. "QG","SAG","IR" refer to the subcomponent queries generation, subcomponent answer generation and information retrieval.

type are presented in Appendix F.

As shown in Table 3, TreeRare achievees a higher success rate compared to ReAct(Yao et al., 2022), indicating TreeRare's effectiveness in enhancing the LLM's ability to produce correct answers. ReAct exhibits a high rate of retrieval errors, suggesting a lack of effective guidance in query generation. This is primarily due to its heavy reliance on few-shot prompting and the model's reasoning abilities. In contrast, TreeRare offers explicit structural guidance through syntax tree, which results in more effective queries to guide retrieval and a large decrease in the retrieval error rate. Additionally, TreeRare reduces the rate of partial answer by 2% and reasoning errors by 8%, suggesting that its structured guidance mechanism better supports the reasoning alignment. Meanwhile, TreeRare has a higher proportion of label ambiguity. While reflecting a higher incidence of mismatch with labeled answers, TreeRare may in fact produce correct responses that differ from annotated references.

5.4 Cost Analysis

To assess the computational efficiency of TreeRare, we measure the total number of input and output tokens generated during inference on 500 randomly sampled examples per dataset. The token usage is translated into cost according to the GPT-4o-mini pricing scheme published by OpenAI. Figure 4 illustrates the cost breakdown across different methods. We observe that TreeRare (CT) incurs higher inference cost compared to TreeRare (DT). This difference can be attributed to the structural characteristics of constituency trees, which represent nested phrase structures and thus tend to include more nodes and sub-phrases than dependency trees. Since TreeRare performs query generation and retrieval at each node, deeper trees with more branches lead to increased token usage. Further-
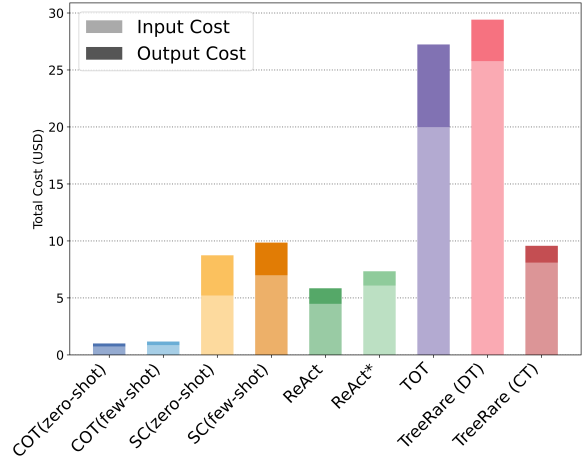

Figure 4: Total GPT-4o-mini API cost for TreeRare (CT), TreeRare (DT), TOT, ReAct, SC, and COT across HotpotQA, MuSiQue, 2WikiMQA, AmbigDoc, and ASQA

more, since methods such as ToT and SC involve extensive sampling of reasoning trajectories, they inflate both input and output token counts and exhibit significantly high computational costs. TreeRare (DT) offers a favorable trade-off, achieving better performance with moderate computational demands.

6 Conclusion

In conclusion, we propose TreeRare for knowledge-intensive question answering, utilizing syntax trees to guide information retrieval and structural reasoning. When traversing the syntax tree, our method performs subcomponent-based information retrieval and question answering. This structured approach enhances retrieval quality and models' ability to resolve information gaps at each node. Experimental results across knowledge-intensive benchmarks demonstrate that our method achieve significant performance improvements over state-of-the-art baselines.

Limitation

While TreeRare demonstrates strong performance across multihop and ambiguous question answering datasets, several limitations remain.

First, TreeRare relies on the quality and granularity of syntax parsers. Errors in dependency or constituency parsing may propagate through the bottom-up reasoning pipeline, leading to suboptimal subcomponent decomposition and misaligned query formulation.

Second, TreeRare incurs additional computational overhead due to its multi-stage decomposition, retrieval, and filtering pipeline. This overhead is particularly pronounced for the constituency tree variant, which typically produces deeper and more richly branched trees, resulting in increased token usage. Such cost implications may hinder TreeRare's scalability in latency- or budget-constrained deployment settings.

Third, TreeRare has been evaluated exclusively on factoid-style questions, where each query maps to discrete factual answers. Its performance on open-domain dialogue or generative settings—such as those requiring opinion modeling, pragmatic reasoning, or user intent tracking—remains unexplored.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altensch1midt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Meta AI. 2024. Llama 3.3 70b. https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct. Instruction-tuned multilingual large language model.

Samuel Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. 2023. QAMPARI: A benchmark for open-domain questions with many answers. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 97–110, Singapore. Association for Computational Linguistics.

Reinald Kim Amplayo, Kellie Webster, Michael Collins, Dipanjan Das, and Shashi Narayan. 2023. Query refinement prompts for closed-book long-form QA. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7997–8012, Toronto, Canada. Association for Computational Linguistics.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2024. Universal self-consistency for large language models. In *ICML 2024 Workshop on In-Context Learning*.

Xiuying Chen, Mingzhe Li, Xin Gao, and Xiangliang Zhang. 2022. Towards improving faithfulness in abstractive summarization. In *Advances in Neural Information Processing Systems*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 423–429, Barcelona, Spain.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. *arXiv e-prints*, pages arXiv–2209.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1251–1265, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jerry A Fodor and Ernest Lepore. 2002. *The compositionality papers*. Oxford University Press.

Yifan Gao, Henghui Zhu, Patrick Ng, Cicero Nogueira dos Santos, Zhiguo Wang, Feng Nan, Dejiao Zhang, Ramesh Nallapati, Andrew O. Arnold, and Bing Xiang. 2021. Answering ambiguous questions through generative evidence fusion and round-trip prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3263–3276, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Hyuhng Joon Kim, Youna Kim, Cheonbok Park, Junyeob Kim, Choonghyun Park, Kang Min Yoo, Sanggoo Lee, and Taeuk Kim. 2024. Aligning language models to explicitly handle ambiguity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1989–2007, Miami, Florida, USA. Association for Computational Linguistics.

Bernhard Kratzwald, Anna Eigenmann, and Stefan Feuerriegel. 2019. RankQA: Neural question answering with answer re-ranking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6076–6085, Florence, Italy. Association for Computational Linguistics.

RONALD W LANGACKER. 1997. Constituency, dependency, and conceptual grouping. *Cognitive Linguistics*, 8(1):1–32.

Yoonsang Lee, Xi Ye, and Eunsol Choi. 2024. Ambigdocs: Reasoning across documents on different entities under the same name. In *First Conference on Language Modeling*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Jiwei Li, Thang Luong, Dan Jurafsky, and Eduard Hovy. 2015. When are tree structures necessary for deep learning of representations? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2304–2314, Lisbon, Portugal. Association for Computational Linguistics.

Weijie Li, Jin Wang, Liang-Chih Yu, and Xuejie Zhang. 2025. Topology-of-question-decomposition: Enhancing large language models with information retrieval for knowledge-intensive tasks. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2814–2833, Abu Dhabi, UAE. Association for Computational Linguistics.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. 2024. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *The Twelfth International Conference on Learning Representations*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024a.

Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4o mini. https://platform.openai.com/docs/models/gpt-4o-mini. Large language model.

Bhawna Piryani, Abdelrahman Abdallah, Jamshid Mozafari, and Adam Jatowt. 2024. Detecting temporal ambiguity in questions. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9620–9634, Miami, Florida, USA. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Corbin L Rosset, Chenyan Xiong, Minh Phan, Xia Song, Paul N. Bennett, and saurabh tiwary. 2021. Pretrain knowledge-aware language models.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, Mexico City, Mexico. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

Jiejun Tan, Zhicheng Dou, Yutao Zhu, Peidong Guo, Kun Fang, and Ji-Rong Wen. 2024. Small models, big insights: Leveraging slim proxy models to decide when and what to retrieve for LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4420–4436, Bangkok, Thailand. Association for Computational Linguistics.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

11

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate problem solving with large language models. *Preprint*, arXiv:2305.10601.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62138–62160. PMLR.

A TreeRare Implementation Details

Parsing Module. To construct the syntax trees required by TreeRare, we utilize the dependency parser and constituency parser from Stanza(Qi et al., 2020). For each input question, we parse it into both dependency and constituency structures. We implement a unified interface to map parsed trees into a bottom-up traversal format, ensuring that each node contains: (i) its corresponding sub-phrase span, (ii) its child nodes. (iii) syntactic type (e.g., NP, VP for constituency, or head-dependent relation for dependency).

Traversing Details In the experiment, TreeRare conducts a pre-order traversal. Each node is processed only after all its child nodes are resolved. We maintain a processing queue initialized with leaf nodes. When processing a node: If it has no children, we directly use its text span to generate quries. If it has children, we first aggregate evidence from its children before proceeding to subcomponent-specific query generation. If it has no children, we directly use its text span as the initial evidence. To improve computational efficiency, we implement a pruning mechanism that skips nodes based on two criteria: Nodes with syntactic types typically considered non-informative (e.g., punctuation, determiners, conjunctions). Nodes whose associated sub-phrases are shorter than three tokens.

Subcomponent Query Generation. At each non-leaf node, we generate queries to resolve syntactic uncertainty associated with its sub-phrase. We prompt the LLM using templates detailed in Appendix G We generate up to five candidate queries and select the top three queries according to heuristic rules prioritizing coverage and specificity.

Retrieval Module. We use BM25 via Pyserini (Lin et al., 2021) as the retrieval backend. For each generated query, we retrieve the top-15 paragraphs from Wikipedia dump in (Karpukhin et al., 2020). If multiple queries exist for a node, their retrieved documents are merged. In Table 4, we present the comparison of the performance between BM25 and DPR(Karpukhin et al., 2020) on TreeRare. We observe that BM25 outperforms DPR under five benchmark, and thus we implement TreeRare with BM25 as the backbone retriever.

B Baseline Implementation Details

To ensure a fair comparison with TreeRare, we implemented all baseline prompting methods within a direct RAG setup using a shared retrieval backbone. Specifically, we employ BM25 (Robertson et al., 2009) as the sparse retriever and retrieve the top-20 most relevant passages from a Wikipedia corpus for each query. We directly use the implementation from Pyserini (Lin et al., 2021). The retrieved contexts are concatenated with the input prompt and passed to the large language model, leveraging its extended context window.

Few-shot Prompting. For few-shot prompting, we prepend three in-context examples drawn from the same dataset as the test instance

Chain-of-Thought Prompting. We follow the original CoT formulation (Wei et al., 2022), appending reasoning demonstrations to the prompt to elicit step-by-step inference. In the few-shot CoT setting, each demonstration consists of a question, a multi-step rationale, and the final answer. In zero-shot CoT, the test query is prefixed by the phrase "Let's think step by step."

Self-Consistency. For multihop question, we generate 10 independent reasoning trajectories using the few-shot CoT prompt. The model's final answer is selected by majority vote among the answers extracted from each reasoning trace. However, SC is not suitable for long-form answer generation. In AmbigDoc and ASQA, we follow the USC(Chen et al., 2024) protocol for self-consistency.

React. We closely follow the original ReAct framework as proposed by (Yao et al., 2022). The maximum number of steps is set to eight. If the model fails to reach a conclusive answer within this limit, we default to using Self-Consistency prompting to generate the final response. For ambiguous questions, we adopt the query refinement prompts introduced by (Amplayo et al., 2023) abd incorporate few-shot exemplars directly into the prompt.

Tree-of-Thoughts. The original Tree-of-Thought paper does not provide a pipeline tailored for multihop or ambiguous question answering. Therefore, we implement ToT following the setup in (Zhou et al., 2024). Instead of sampling multiple reasoning paths as in the original version, our implementation adopts the React framework to sample diverse plan-and-action paths, enabling interaction with Wikipedia and equipping ToT with

Retriever	HotpotQA COV-EM	MuSiQue COV-EM	2WikiMQA COV-EM	AVG	AmbigDoc AR	AmbigDoc ER	ASQA COV-EM	ASQA Dis-F1
Tree-Retrieval (DT) *w* BM25	0.528	0.138	0.562	0.409	0.427	0.581	0.514	0.361
Tree-Retrieval (DT) *w* DPR	0.482	0.16	0.504	0.382	0.409	0.479	0.399	0.332

Table 4: Comparison of Tree-Retrieval (DT) with BM25 and DPR as retriever backbone.
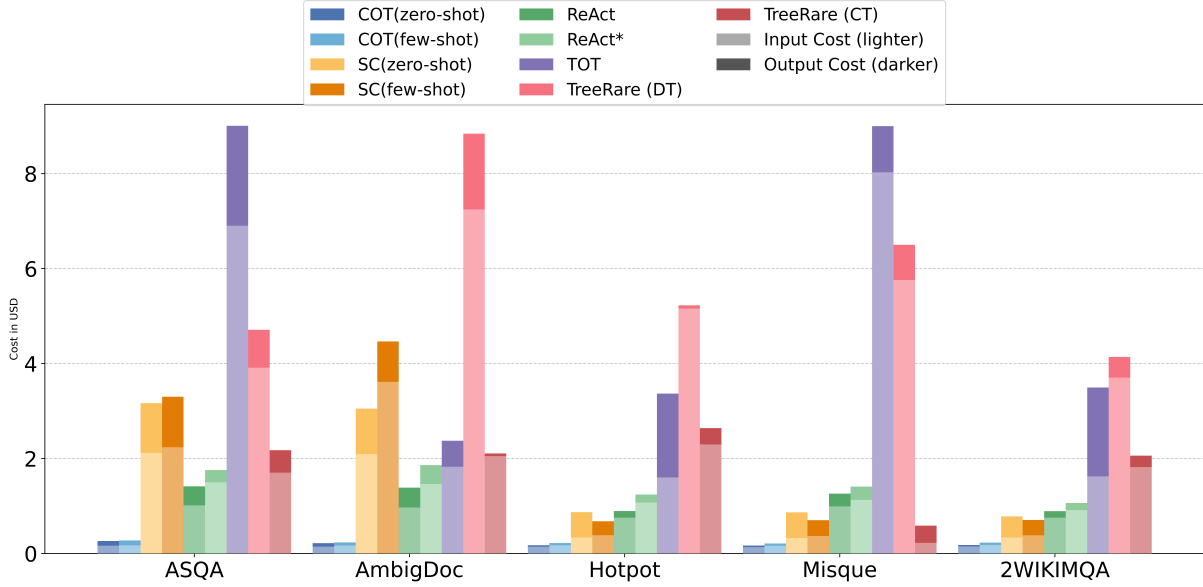

Figure 5: Total GPT-4o-mini API cost (input + output tokens) for TreeRare(CT), TreeRare(DT), TOT, ReAct, SC, and COT across HotpotQA, MuSiQue, 2WikiMQA, AmbigDoc, and ASQA based on OpenAI pricing.

enhanced capabilities for open-domain QA.

C Ablation Study Details

We do not include a setting with only subcomponent question answering in our ablation study. This is because it is not feasible to generate fine-grained answers without first performing query generation and retrieval.

TreeRare(DT) *w/o* CQ. To assess the impact of subcomponent-based query generation, we replace the TreeRare query generation module with a naive retrieval. Specifically, for each node in the syntactic tree, we bypass the LLM-generated term-specific queries and instead directly use the surface form of the corresponding sub-phrase as the retrieval query. In the downstream subcomponent answering stage, we prompt LLMs to directly answer the question with the retrieved documents.

TreeRare(DT) *w/o* QA. In this variant, we eliminate the intermediate reasoning step that resolves each node's syntactic uncertainty. Instead of prompting the LLM to process the retrieved evidence at each node, we directly aggregate all

retrieved documents across sub-nodes and forward the combined evidence to their parent node without further interpretation.

D Tree Retrieval

For each node in the syntax tree, we directly use sub-phrase p_n without any LLM-based reformulation. This phrase is used as a query to the retrieval systems. We retrieve the top10 passages from Wikipedia corpus using BM25. The retrieved passages across all nodes within a sub-tree are merged. To suppress noise and prioritize passages most relevant to the sub-tree's syntactic content, we apply a cross-encoder reranker MS-Marco-MiniLM-L-12-v2. In Table 4, we compared Tree-Retrieval with different both BM25 and DPR(Karpukhin et al., 2020).

E Further Cost Analysis

We measure the API cost of GPT-4o-mini by summing the number of input and output tokens processed for each method across five datasets: HotpotQA, MuSiQue, 2WikiMultihopQA, AmbigDoc, and ASQA. For each dataset, 500 examples were

14

randomly sampled and processed with each method. The input and output tokens were multiplied by OpenAI's published pricing for GPT-4o-mini (as of April 2025).

F   Case Study

Here is the definition for different categories:

- **Success**: Model produces a correct and complete answer aligned with the reference.

- **Retrieve Error**: Retrieved documents are irrelevant or do not aid answer resolution.

- **Reasoning Error**: Model performs faulty reasoning despite relevant evidence being available.

- **Partial Answer**: Answer is generally correct but lacks necessary specificity.

- **Label Ambiguity**: Prediction is plausible but does not exactly match the labeled answer.

we randomly sampled 50 multihop questions from the HotpotQA development set. For each sample, a human annotator examined the reasoning trace produced by both ReAct and TreeRare, along with the final answer and supporting documents. We present one representative example for each failure category.

> **Partial Answer**
>
> **TreeRare:**
> **Question:** What is the name of the fight song of the university whose main campus is in Lawrence, Kansas and whose branch campuses are in the Kansas City metropolitan area?
> ....
> **Final answer:** The fight song of the University of Kansas is "I'm a Jayhawk."(Label: Kansas Song)
> **ReAct:**
> **Question:** The director of the romantic comedy "Big Stone Gap" is based in what New York city?
> ....
> **Final answer:** New York City

Figure 7: Examples of *Partial Answer*. The partial answer is highlighted in red.

> **Search Error**
>
> **TreeRare:**
> **Question:** What type of forum did a former Soviet statesman initiate? **Generate Question:** What is the name of the former Soviet statesman who initiated the forum? ... **answer:** The name of the former Soviet statesman who initiated a notable forum is Yegor Gaidar....
> **Generate Question:** What was the name of the former Soviet statesman? ... **answer:** The name of the former Soviet statesman is Mikhail Suslov .... question: What was the role of Mikhail Suslov in the Soviet Union? ....
> answer: Mikhail Suslov served as the Second Secretary .....
> **Generate Question:** What topics are typically discussed in the forum organized by the former Soviet statesman? .... **answer:** The document does not contain specific information about a political forum initiated by a former Soviet statesman ...
> **Fianl answer:** former Soviet statesman, Yegor Gaidar, initiated the Gaidar Forum, which focuses on discussions among Russia's political and business elite regarding economic reforms and political strategies. **ReAct:**
> **Question:** This singer of A Rather Blustery Day also voiced what hedgehog?
> Search[A Rather Blustery Day] Thought 1: I need to search for the song "A Rather Blustery Day"...
> Action 1: Search[A Rather Blustery Day] Observation 1: Winnie the Pooh and the Blustery Day is a 1968 American animated musical fantasy short ...
> Lookup[singer of A Rather Blustery Day] Thought 2: The search didn't give me the specific singer ...
> Action 2: Lookup[singer of A Rather Blustery Day] Observation 2: No more results.
> Search[A Rather Blustery Day singer] Thought 3: The search did not yield any direct information
> Action 3: Search[A Rather Blustery Day singer] Observation 3: Could not find A Rather Blustery Day singer
> Search[hedgehog character voiced by a singer from Winnie the Pooh] Thought 4: The search did not provide specific details
> Action 4: Search[hedgehog character voiced by a singer from Winnie the Pooh] Observation 4: Could not find hedgehog character
> Search[hedgehog character in animated films] Thought 5: The search has not provided concrete information
> Action 5: Search[hedgehog character in animated films] Observation 5: Could not find hedgehog character
> Search[singer of Sonic the Hedgehog] Thought 6: It seems that I'm stuck
> Action 6: Search[singer of Sonic the Hedgehog] Observation 6: Could not find singer of Sonic the

Figure 8: Examples of *Search Error*. The search result or incorrect query is highlighted in red.

<div style="border: 2px solid blue; border-radius: 8px;">

Incorrect reasoning

TreeRare:

Question: question: Brown State Fishing Lake is in a country that has a population of how many inhabitants ?

Generate Question: What country is Brown State Fishing Lake located in?; **answer:** Brown State Fishing Lake is located in the United States ....

Generate Question: What is the population of the United States as of 2023? **answer:**The population of the United States as of 2023 is approximately 333 million people. ....

Final answer: The population of the United States, where Brown State Fishing Lake is located, is approximately 333 million as of 2023. (label: 9,984)

ReAct:

Question: Which performance act has a higher instrument to person ratio, Badly Drawn Boy or Wolf Alice? Search[Badly Drawn Boy] Thought 1: I need to search both Badly Drawn Boy and Wolf Alice ..... Action 1: Search[Badly Drawn Boy] Observation 1: Damon Michael Gough (born 2 October 1969), ... and multi-instrumentalist.

Lookup[Badly Drawn Boy performances] Thought 2: .... I need to search for more explicit information about his performances.

Action 2: Search[Wolf Alice] Observation 2: Wolf Alice are an English rock band fr.... acoustic duo comprising singer Ellie Rowsell and guitarist Joff Oddie, Wolf Alice have also featured bassist Theo Ellis and drummer Joel Amey since 2012 .... Lookup[Wolf Alice instruments] Thought 4: Wolf Alice consists of four members: Ellie Rowsell (vocals), Joff Oddie (guitar), Theo Ellis (bass), and Joel Amey (drums). .....

Action 3: Lookup[Wolf Alice instruments] Observation 3: No more results.

Finish[Wolf Alice] Thought 4: I know they reportedly consist of four members and typically feature guitar, bass, drums, and vocals whwereas Badly Drawn Boy only have one member . (label: Badly Drawn Boy)

</div>

Figure 9: Examples of *Reasoning Error*. The incorrect part is highlighted in red.

G   Prompt for TreeRare

Final answer generation prompt for multi hop QA

Answer the following question: {{questions}} , with following documents: {{documents}}.

Your response should strictly follow the format:
Explanations :[give your step by step Analysis here]

FINAL:(BE CONCISE, ONLY a FEW phrases)

let's think step by step

Subcomponent Question generation prompt for Ambiguous QA

You're a disambiguation expert analyzing "{{phrase}}" in: {{self.question}} Instruction:
1. Analyze the question by considering these potential ambiguities:
- Temporal: Check for unclear time references, periods, or temporal scope
- Entity: Identify names, references, or terms that could refer to multiple entities
- Semantic: Look for words with multiple meanings (polysemy/homonymy)
- Scope: Consider possible boundaries and levels of detail
- Intent: Examine possible purposes and expected answer types
- Cultural: Consider cultural-dependent interpretations
- Quantitative: Check for unclear measurements or numerical references
- Linguistic: Analyze syntax and referential clarity
- Categorical: Consider possible classification schemes
- Contextual: Examine required background knowledge and relationships
2. Analyze the question word by word. Return disambiguated question and its interperatation for each different meaning

Here is what we currently know Documents:{{context}}

pick top 5 questions that are best in disambiguating the question. (covers different meanings of the questions) and strictly FOLLOW the format: response: question1; question2;....

Final answer generation prompt for Ambiguous QA

The question may be ambiguous and have multiple correct answers, and in that case, you have to provide a long-form answer including all correct answers.
1. Carefully go through all the given documents.
2.The using your and context, provide answer.
Your response should strictly follow the format:
Explanations (Step 2):[give your step by step Analysis here]

FINAL(Step 2):
Please ONLY reply according to this format
Question: {{questions}} Document: {{documents}} let's think step by step

Subcomponent question answering prompt

Answer the {{questions}} based on on the document info. For each question find as many answers as possible. Response all the answers in a short paragraph (as specific as possible).
Relevant Document: {{context}}

Subcomponent Question generation prompt for multihop QA

You're a an expert analyzing "{{phrase}}" in main query: "{{self.question}}".
1.You should generate at most 5 simple questions that mainly ask for information about {{phrase}} that helps understand the main query. Question should be single-hop, clear and search-friendly.
2. Here is what we currently know
Documents:{{context}}
pick top 3 questions that are best in decomposing the main query and leading to final answer of the main query.
strictly FOLLOW the format: response: question1; question2;....