Coordinating Search-Informed Reasoning and Reasoning-Guided Search in Claim Verification

Anonymous ACL submission

Abstract

Multi-hop claim verification is inherently challenging, requiring multi-step reasoning to construct verification chains while iteratively searching for information to uncover hidden bridging facts. This process is fundamentally interleaved, as effective reasoning relies on dynamically retrieved evidence, while effective search demands reasoning to refine queries based on partial information. To achieve this, we propose **H**ierarchical **A**gent **R**easoning and **I**nformation **S**earch (HARIS), explicitly modeling the coordinated process of reasoning-driven searching and search-informed reasoning. HARIS consists of a high-level reasoning agent that focuses on constructing the main verification chain, generating factual questions when more information is needed, and a low-level search agent that iteratively retrieves more information, refining its search based on intermediate findings. This design allows each agent to specialize in its respective task, enhancing verification accuracy and interpretability. HARIS is trained using reinforcement learning with outcome-based rewards. Experimental results on the EX-FEVER and HOVER benchmarks demonstrate that HARIS achieves strong performance, greatly advancing multi-hop claim verification.

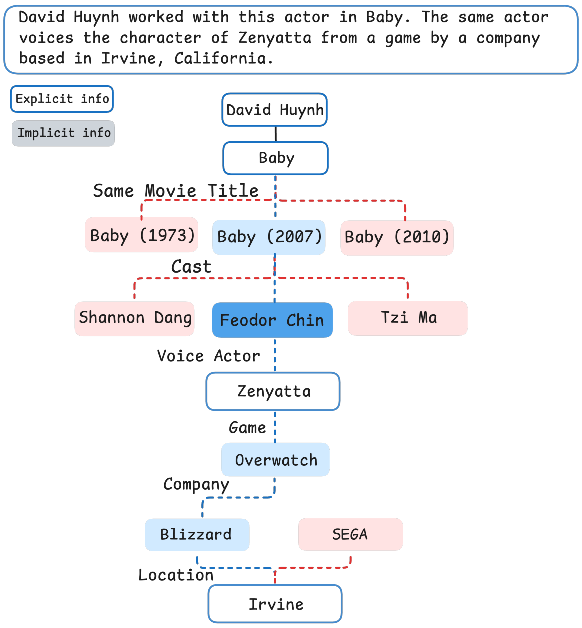


Figure 1: Example of a challenging multi-hop verification. Verifying the claim demands coordinating reasoning-guided search to disambiguate entities and search-informed reasoning to adapt based on retrieved evidence. Prematurely concluding on any distracting branch (in red) leads to incorrect judgment. The correct path—from *Baby (2007)* to *Overwatch* via *Feodor Chin*—emerges only through this dynamic interplay, not static decomposition or single-pass retrieval.

1 Introduction

Claim verification (Guo et al., 2022) has become a critical challenge as misinformation proliferates online. It requires systems to determine whether a given claim is supported or refuted based on retrieved evidence. While verifying simple claims involves shallow reasoning within a single document, the verification of complex, multi-hop claims presents a fundamentally different challenge. This difficulty stems from the fragmented nature of evidence (Pham et al., 2025; Atanasova et al., 2022). Effectively verifying such claims requires a joint process of multi-step reasoning and iterative information searching (Zheng et al., 2025).

A key challenge in multi-hop verification is identifying the correct bridging facts—implicit links that connect separate pieces of evidence but are not explicitly stated. As shown in Figure 1, verifying this claim requires identifying *Feodor Chin* as the critical bridging fact between the film *Baby (2007)* and the character *Zenyatta* from the game *Overwatch*, among other irrelevant paths. This process necessitates intensive interactions between **reasoning**, which proposes candidate hypotheses to identify potential bridging facts, and **iterative search**, which retrieves evidence to validate or eliminate certain hypotheses. In particular, reasoning is necessary to construct verification chains, but effective reasoning depends on relevant evidence, which often requires iterative search. Meanwhile, effective search relies on reasoning to formulate queries based on partially retrieved evidence. This reciprocal relationship, where reasoning shapes search and retrieved evidence refines ongoing reasoning, captures the recursive nature of multi-hop verification.

Conventional approaches typically involve decomposing complex claims into sub-claims or questions, followed by independent verification (Kamoi et al., 2023; Chen et al., 2024; Lu et al., 2025). However, this strategy can struggle when critical bridging facts are implicit and not directly recoverable from the claim. More advanced methods impose structured reasoning frameworks, such as graphs, reasoning programs, or First-Order Logic (FOL), to better coordinate evidence collection and reasoning (Pan et al., 2023b; Wang and Shu, 2023; Pham et al., 2025). However, these approaches often overlook the dynamic interplay between reasoning and information retrieval, which can be critical for accurate multi-hop verification.

To tackle these, we propose HARIS—**H**ierarchical **A**gent **R**easoning and **I**nformation **S**earch, explicitly modeling the coordinated process of reasoning-driven searching and search-informed reasoning. HARIS consists of two specialized large language model (LLM) agents: a high-level reasoning agent and a low-level search agent, both trained using reinforcement learning (RL) to optimize their respective tasks. The high-level agent forms the main verification chain, generating factual questions when more information is needed. The low-level agent handles these questions through dynamic search, iteratively refining its queries based on partial results to progressively build a comprehensive evidence base. This design allows each agent to specialize in its respective task, enhancing both verification accuracy and interpretability by clearly modeling the mutually reinforcing interaction between reasoning and information searching. To our knowledge, HARIS is among the first RL-based cooperative agent approach for claim verification.

Our contributions are summarized as follows:

- We propose HARIS, a hierarchical agent framework designed for complex multi-hop claim verification, explicitly modeling the coordinated process of reasoning-driven searching and search-informed reasoning.
- HARIS is trained end-to-end via Group Relative Policy Optimization with outcome-based rewards, directly optimizing task performance without intermediate supervision.
- HARIS demonstrates strong performance on two challenging benchmarks, EX-FEVER and HOVER, validating its effectiveness in tackling multi-hop claim verification.

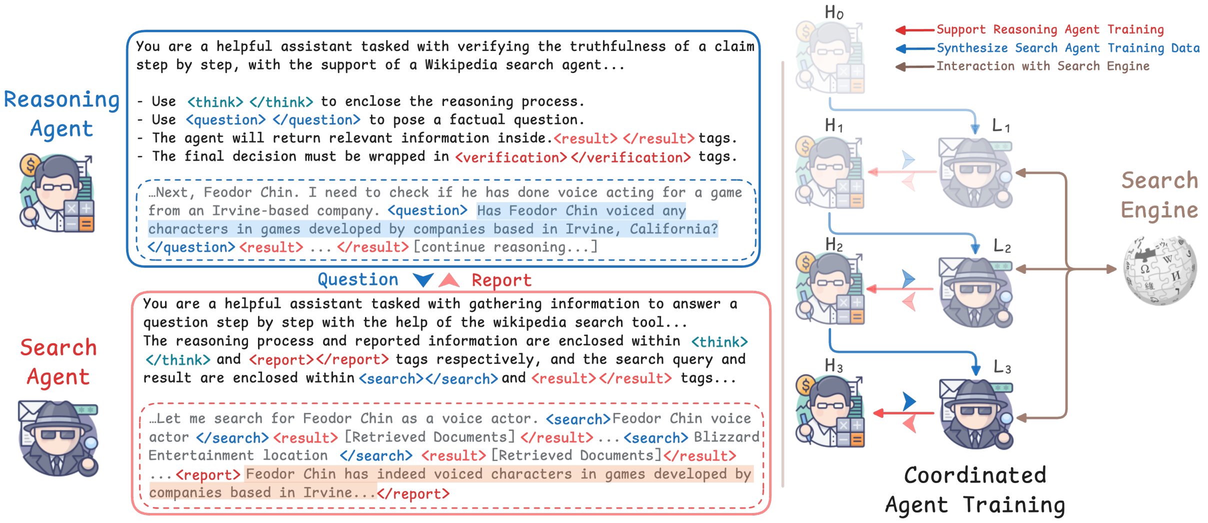
2 Related Work

2.1 Claim Verification

Claim verification research has been increasingly focusing on enhancing transparency during decision making (Zeng and Gao, 2024; Chen et al., 2024; Hu et al., 2025a). QACheck (Pan et al., 2023a) reformulates verification as a progressive question-answering (QA) task, validating claims through step-wise questioning. Structural approaches (Jeon and Lee, 2025; Pham et al., 2025), such as ProgramFC (Pan et al., 2023b) and FOLK (Wang and Shu, 2023), which use symbolic reasoning or reasoning program to enforce systematic verification. The Decompose-Then-Verify (Wanner et al., 2024b,a; Jiang et al., 2024; Hu et al., 2025b; Lu et al., 2025) paradigm focus on breaking down complex claims into simpler sub-claims for independent validation. Agentic approaches have emerged as a promising direction (Zhao et al., 2024). LoCal (Ma et al., 2025) employs a prompt-driven multi-agent framework emphasizing causal consistency, and BiDeV (Liu et al., 2025) uses specialized agents to address vagueness and redundancy. In contrast, HARIS formulates claim verification as a cooperative process between reasoning and search agents, and trains both agents jointly via reinforcement learning.

2.2 Reasoning & Searching

Recent work has expanded the reasoning capabilities of LLMs by integrating search mechanisms (Xiong et al., 2025; Guan et al., 2025; Sun et al., 2025). Notable methods like Search-o1 (Li et al., 2025) incorporate dynamic search into reasoning frameworks, improving factual accuracy in open-domain and multi-hop reasoning tasks leveraging LLMs. Furthermore, RL methods like Group Relative Policy Optimization (GRPO) (Shao et al., 2024) have been used to incentivize search capabilities of LLMs (Gao et al., 2025), encouraging


Reasoning Agent
Question
Report
Search Agent
H₀
H₁
H₂
H₃
L₁
L₂
L₃
Support Reasoning Agent Training
Synthesize Search Agent Training Data
Interaction with Search Engine
Search Engine
Coordinated Agent Training

```
You are a helpful assistant tasked with verifying the truthfulness of a claim
step by step, with the support of a Wikipedia search agent...

- Use <think> </think> to enclose the reasoning process.
- Use <question> </question> to pose a factual question.
- The agent will return relevant information inside.<result> </result> tags.
- The final decision must be wrapped in <verification></verification> tags.

…Next, Feodor Chin. I need to check if he has done voice acting for a game
from an Irvine-based company. <question> Has Feodor Chin voiced any
characters in games developed by companies based in Irvine, California?
</question><result> ... </result>[continue reasoning...]
```

```
You are a helpful assistant tasked with gathering information to answer a
question step by step with the help of the wikipedia search tool...
The reasoning process and reported information are enclosed within <think>
</think> and <report></report> tags respectively, and the search query and
result are enclosed within <search></search> and <result></result> tags...

…Let me search for Feodor Chin as a voice actor. <search>Feodor Chin voice
actor </search><result> [Retrieved Documents] </result>...<search> Blizzard
Entertainment location </search> <result> [Retrieved Documents]</result>
...<report> Feodor Chin has indeed voiced characters in games developed by
companies based in Irvine...</report>
```

Figure 2: Interaction workflow (left) and coordinated agent training process (right). The reasoning agent constructs the verification chain and issues questions (`<question>`) to the search agent. The search agent performs iterative retrievals (`<search>`) and report relevant information (`<report>`) back to the reasoning agent. During training, QA data from reasoning rollouts is used to update the search agent, which in turn supports reasoning agent updates, keeping both agents aligned.

models to generate context-aware queries and effectively integrate retrieved information (Jin et al., 2025; Song et al., 2025; Chen et al., 2025; Qian et al., 2025; Wu et al., 2025). Collectively, these approaches demonstrate that RL-trained search integration can effectively improve performance on knowledge-intensive tasks.

3 Methodology

3.1 Why Reasoning and Search Agents?

Verification reasoning and factual information searching demand fundamentally different capabilities. Reasoning requires multi-step planning, identifying hidden facts, and maintaining logical consistency across long contexts. In contrast, effective searching depends on precise query formulation, iterative refinement, and robust extraction of relevant evidence from noisy or incomplete results. Rather than overloading a single model with both tasks, we decouple these roles into two specialized agents: a reasoning agent that interprets the claim, tracks verification progress, and decides when new information is needed; and a search agent that dynamically retrieves and refines evidence through focused interaction with a retrieval system.

Inspired by human cognition, where individuals offload information gathering to collaborators to reduce burden, HARIS is designed to mimic this process. By delegating search and reasoning to distinct agents, we improve decision traceability in multi-hop claim verification. This design is especially effective in cases requiring nuanced disambiguation and step-wise evidence composition.

Meanwhile, the search and reasoning agents closely collaborate to enhance overall performance. Interactive generation allows the agents to iteratively guide each other's outputs. Coordinated training alternately optimizes the agents to strengthen their collaboration, as shown in Figure 2.

3.2 High-level Reasoning Agent

The high-level reasoning agent constructs the main verification chain, coordinating multi-step reasoning and generating factual questions for the search agent when additional information is needed.

3.2.1 Reasoning Agent Rollout

Following prior work (Chen et al., 2025; Jin et al., 2025), the reasoning agent's rollout process uses special tags to define question actions. Specifically, the tags `<question>` and `</question>` indicate that an action to call the search agent should be invoked. Upon detecting the `</question>` tag, the generation is paused and the enclosed content will be regarded as the factual question and sent to the search agent. The reported information from search agent is wrapped within `<result>` tags and appended to the sequence to enable continued rollout. The rollout ends when a final verification is derived, wrapped within `<verification>` tags. The prompt template is provided in Appendix C.1.

3.2.2 Reasoning Agent Reward

The reasoning agent is responsible for performing multi-step reasoning over the gathered evidence to verify the claim. It provides a binary verification decision, and the reward is the correctness of this decision. The overall reward R_{high} is given by:

$$R_{\text{high}} = \begin{cases} 1, & \text{if correct prediction} \\ 0.1, & \text{if wrong prediction, correct format} \\ 0, & \text{if wrong format} \end{cases} \tag{1}$$

3.3 Low-level Search Agent

The low-level search agent in HARIS is responsible for handling factual questions generated by the reasoning agent. Unlike conventional question-answering systems that aim to produce concise answers, the search agent here is designed to iteratively gather comprehensive and relevant information to support the high-level reasoning process.

3.3.1 Training Data Synthesis

To train the search agent, it is crucial that the training data not only includes diverse question-answer pairs but also closely aligns with the type of the questions generated by the high-level agent. This alignment ensures that the search agent can effectively collaborate with the reasoning agent during multi-hop verification. To create such training set, we sample questions from the high-level agent rollouts, pair them with the corresponding ground-truth evidence, and use GPT-4o (OpenAI, 2024b) to generate pseudo ground-truth answers. To maintain data quality, we filter out pairs where GPT-4o outputs "none" as the answer, typically removing about 10% of the data. This process synthesize data that is both contextually relevant and closely matched to the reasoning patterns of the high-level agent. Synthesis details and examples can be found in Appendix A.4 and Table 11.

3.3.2 Search Agent Rollout

The search agent rollout process is similar to the reasoning agent. When the generation process encounters a `</search>` tag, it pauses and extracts the content wrapped within the tags as the search query. This query is then used to perform top-k retrieval from the knowledge corpus. The retrieved text is wrapped in `<result>` and `</result>` tags and appended to the paused sequence, allowing the rollout to continue iteratively until the agent determines that sufficient information has been gathered. At this point, the agent reports the collected evidence within `<report>` and `</report>` tags, marking the completion of the rollout. The prompt template can be found in Appendix C.1.

3.3.3 Search Agent Reward

We use a combination of format and LLM-as-a-Judge approaches for search agent rewards. The LLM-as-a-Judge approach evaluates the quality the gathered information using an LLM, comparing the final output against the pseudo ground-truth answer[1]. The evaluation score is computed as:

$$S = \text{LLM-as-a-Judge}(a_{\text{pred}}, a_{\text{gt}}) \tag{2}$$

where a_{pred} is the search agent's final output and a_{gt} is the pseudo ground-truth answer. This approach is less strict than exact match (EM) metrics, better aligning with the search agent's goal of collecting comprehensive, contextually relevant information rather than just short, exact answers. The implementation can be found in Appendix A.5.1. Formally, the reward R_{low} is given by:

$$R_{\text{low}} = \begin{cases} 1, & \text{if S > 0} \\ 0.1, & \text{if S = 0, correct format} \\ 0, & \text{if wrong format} \end{cases} \tag{3}$$

To assess the reliability of LLM-as-a-Judge, we recruited two annotators[2] for assessing held-out samples using the same criteria as LLM. The results showed strong consistency with human judgment (Cohen's Kappa: 0.81; agreement: 93.3%). See Appendix D for complete human evaluation details.

3.4 Group Relative Policy Optimization

In this work, we leverage Group Relative Policy Optimization (GRPO) (Shao et al., 2024), an RL algorithm tailored for training LLMs with group-level reward normalization. GRPO introduces a relative advantage mechanism, which evaluates the quality of generated responses within groups corresponding to the same input. This design helps stabilize training by reducing variance in gradient updates, thereby promoting more consistent learning. The GRPO objective is formally defined as:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\Bigg[\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}\hat{A}_{i,t} - \beta D_{\text{KL}}[\pi_\theta \parallel \pi_{\theta_{\text{old}}}]\Bigg], \tag{4}$$

---

[1] We use GPT-4o-mini(OpenAI, 2024a) for this purpose, as it is cost-efficient.

[2] https://www.prolific.com/

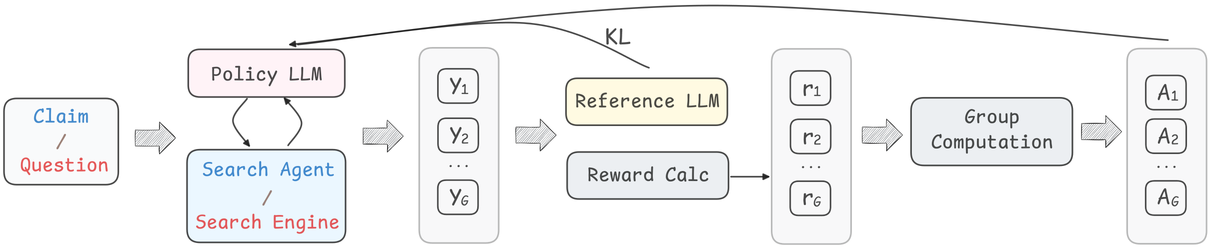


Figure 3: Demonstrations of GRPO for the reasoning and search agents in HARIS. During reasoning agent training, the policy LLM interacts with the search agent, while for search agent training, it interacts with the search engine.

where G is the number of groups, $\hat{A}_{i,t}$ is the normalized advantage within the group, and β controls the KL divergence penalty enforcing policy stability. Figure 3 presents an overview of GRPO for reasoning and search agents.

Retrieved Token Loss Masking Following prior work (Chen et al., 2025; Song et al., 2025), retrieved tokens are masked during loss calculation. This approach ensures that the policy gradient is computed only on generated tokens, reducing bias toward retrieved content and stabilizing training.

3.5 Coordinated Agent Training

To enable effective collaboration between the high-level reasoning agent and the low-level search agent, we adopt a *Coordinated Agent Training* strategy. This process consists of two stages: an initial **foundation stage**, where both agents develop core reasoning, search, and formatting abilities; and a subsequent **alternating training stage**, where the agents iteratively refine their specialized skills through mutual interaction.

Given a training set T of claim verification data, in the foundation stage, the untrained low-level search agent L_0 and high-level reasoning agent H_0 are trained sequentially to establish their foundational capabilities. First, we sample questions from H_0 to train L_0, producing the updated search agent L_1. This updated search agent then supports the training of H_0, producing the updated reasoning agent H_1. This stage establishes a foundation for both agents in reasoning, searching, and formatting.

In the alternating stage, we promote further coordination by repeatedly alternating training between the two agents. The dataset T is divided into N segments. For each segment, we sample questions from the current high-level agent H_i to train the low-level agent, producing L_{i+1}. The updated low-level agent L_{i+1} is then used to collaborate with the high-level agent, resulting in H_{i+1}. This alternating process continues for N rounds, fostering mutual adaptation and ensuring the agents remain closely aligned throughout training.

Algorithm 1 Coordinated Agent Training

```
Input: Initial low-level agent $L_0$, high-level agent $H_0$, training set $T$
Output: Trained low-level agent $L$ and high-level agent $H$
Stage 1: Foundation Training
$Q_0 \leftarrow \text{Synthesis}(T, H_0)$
$L_1 \leftarrow GRPO_{low}(Q_0, L_0)$
$H_1 \leftarrow GRPO_{high}(T, L_1)$
Stage 2: Alternating Training
Divide $T$ into $N$ segments $\{T_1, \dots, T_N\}$
for $i = 1$ to $N$ do
    $Q_i \leftarrow \text{Synthesis}(T_i, H_i)$
    $L_{i+1} \leftarrow GRPO_{low}(Q_i, L_i)$
    $H_{i+1} \leftarrow GRPO_{high}(T_i, L_{i+1})$
end for
$L \leftarrow L_{N+1}, H \leftarrow H_{N+1}$
Return $L, H$
```

4 Experiments

4.1 Datasets

We utilize the following datasets for training and evaluation:

- **EX-FEVER** (Ma et al., 2024): A benchmark for multi-hop claim verification, designed to assess a model's ability to verify complex claims through 2-hop and 3-hop reasoning over hyperlinked Wikipedia documents.

- **HOVER** (Jiang et al., 2020): A dataset created for many-hop claim verification, featuring claims that require 2 to 4-hop reasoning across multiple Wikipedia articles.

	HOVER						EX-FEVER				CHECKWHY
	2-hops		3-hops		4-hops		2-hops		3-hops		
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	Acc
RAG	58.96	59.20	56.59	56.60	55.06	55.20	68.83	69.00	64.41	64.80	56.40
Decomp-Verify	62.39	62.60	57.27	57.31	54.55	55.60	68.03	68.40	61.95	63.00	36.00
ProgramFC	66.84	66.80	55.35	56.80	48.30	52.60	71.28	71.60	60.34	62.40	21.60
BiDeV	64.51	65.00	57.59	58.60	54.94	57.00	67.75	67.80	61.52	62.00	24.60
QACheck	67.60	67.60	60.60	60.60	58.91	59.00	75.52	75.60	68.42	68.60	58.00
FOLK	67.22	67.60	59.89	61.20	50.90	55.20	75.55	75.80	67.24	68.40	50.70
Search-o1	68.72	69.00	59.34	59.80	54.90	56.60	77.41	77.80	72.08	72.80	52.40
HARIS	**69.31**	**69.40**	**62.33**	**62.80**	**59.84**	**61.00**	**80.12**	**80.20**	**73.93**	**74.20**	**60.80**

Table 1: Performance comparison of different methods on HOVER (2-hops/3-hops/4-hops), EX-FEVER (2-hops/3-hops) and CHECKWHY.

	Single		HARIS	
	F1	Acc	F1	Acc
EX-FEVER$_{\text{2hops}}$	77.59	77.60	80.12	80.20
EX-FEVER$_{\text{3hops}}$	73.20	73.40	73.93	74.20
HOVER$_{\text{2hops}}$	68.20	68.20	69.31	69.40
HOVER$_{\text{3hops}}$	61.55	62.00	62.33	62.80
HOVER$_{\text{4hops}}$	55.48	56.00	59.84	61.00

Table 2: Performance comparison between RL trained single agent and HARIS.

	N=1		N=3	
	F1	Acc	F1	Acc
EX-FEVER$_{\text{2hops}}$	78.77	78.80	80.12	80.20
EX-FEVER$_{\text{3hops}}$	73.20	73.60	73.93	74.20
HOVER$_{\text{2hops}}$	69.37	69.40	69.31	69.40
HOVER$_{\text{3hops}}$	61.08	61.60	62.33	62.80
HOVER$_{\text{4hops}}$	60.87	61.80	59.84	61.00

Table 3: Comparison of performance for coordination training rounds N.

For training, we sample 7,200 examples from the combined EX-FEVER and HOVER training data. For evaluation, following Wang and Shu (2023), we sample 500 instances from the test set of each dataset using stratified sampling, ensuring a balanced label distribution. We use F1-score and accuracy as the primary evaluation metrics. To further assess generalizability, we also evaluate the accuracy on 500 positive[3] test samples from CHECKWHY (Si et al., 2024) in the main experiments. Details of the datasets can refer to Appendix A.1.

4.2 Baselines

We include the following baselines:

RAG : A typical Retrieval-Augmented Generation (RAG) approach where retrieved documents and the input are provided to a LLM for verification. The verification module is implemented using DSPy (Khattab et al., 2024).

Decompose-Then-Verify A commonly used paradigm (Kamoi et al., 2023) involving: decomposing a claim into sub-claims, verifying each independently, and aggregating the results. We utilize the decomposition module from Kamoi et al. (2023) and prompt the LLM for verification, aggregating final results with logical `AND`.

ProgramFC Pan et al. (2023b) leveraged program-guided reasoning for claim verification, generating reasoning programs in a few-shot manner for execution.

QACheck Pan et al. (2023a) verifies claims through iterative question-answering until the LLM determines that sufficient information has been derived. We employ the default Retriever–Reader setting, where an LLM iteratively answers questions using the corpus.

FOLK Wang and Shu (2023) translates claims into First-Order Logic (FOL) clauses and applies FOL-guided reasoning over knowledge-grounded question-answer (QA) pairs. The QA pairs are grounded via an external API[4].

BiDeV Liu et al. (2025) propose two prompt-based LLM agents for defusing vagueness and redundancy: the former clarifies latent information, while the latter removes redundant evidence.

Search-o1 Li et al. (2025) enhances large reasoning models by integrating agentic RAG, allowing autonomous retrieval during multi-step reasoning. It further refines retrieved information through a Reason-in-Documents module.

For more baseline details, refer to Appendix A.3.

---

[3] CHECKWHY is a challenging benchmark. Its negative samples are created by modifying evidence to generate counterfactuals. Hence, we only use the positive samples.

[4] `https://serpapi.com/`

	F1		**LLM-as-a-Judge**	
	F1	Acc	F1	Acc
EX-FEVER$_{\text{2hops}}$	72.44	73.00	75.63	75.80
EX-FEVER$_{\text{3hops}}$	63.85	65.60	68.14	68.80
HOVER$_{\text{2hops}}$	65.94	66.40	67.49	67.60
HOVER$_{\text{3hops}}$	56.42	57.80	58.97	59.60
HOVER$_{\text{4hops}}$	52.38	55.60	57.42	58.80

Table 4: Final performance comparison between HARIS with F1-trained and LLM-as-a-Judge trained search agent.

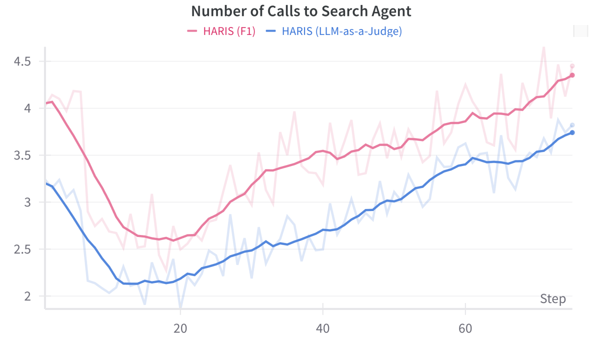


Figure 4: Comparision of calls to search agent during reasoning agent training, using search agents trained with F1 versus LLM-as-a-Judge rewards.

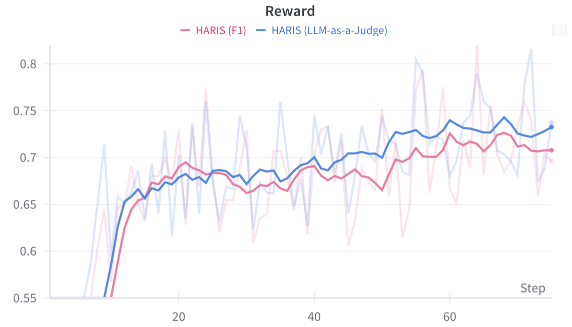


Figure 5: Comparison of high-level rewards during reasoning agent training with search agents trained using F1 and LLM-as-a-Judge rewards.

4.3 Experimental Setup

We conduct all training experiments using the Qwen3-4B model (Qwen, 2025). For HARIS, we train for one epoch in each stage. All baseline methods, except for Search-o1, utilize GPT-4o (OpenAI, 2024b) as the underlying LLM. Search-o1 employs the QwQ-32B-preview model (Qwen, 2024). All methods operate in the Open-Book setting (Pan et al., 2023b), where no ground-truth evidence is provided beforehand, requiring each method to retrieve supporting evidence using top-k ($k = 3$) retrieval. We utilize the Wikipedia corpus provided by FlashRAG (Jin et al., 2024) for this purpose, indexed using a E5-small model for dense retrieval. For more training and experimental details, please refer to Appendix A.5.

5 Result

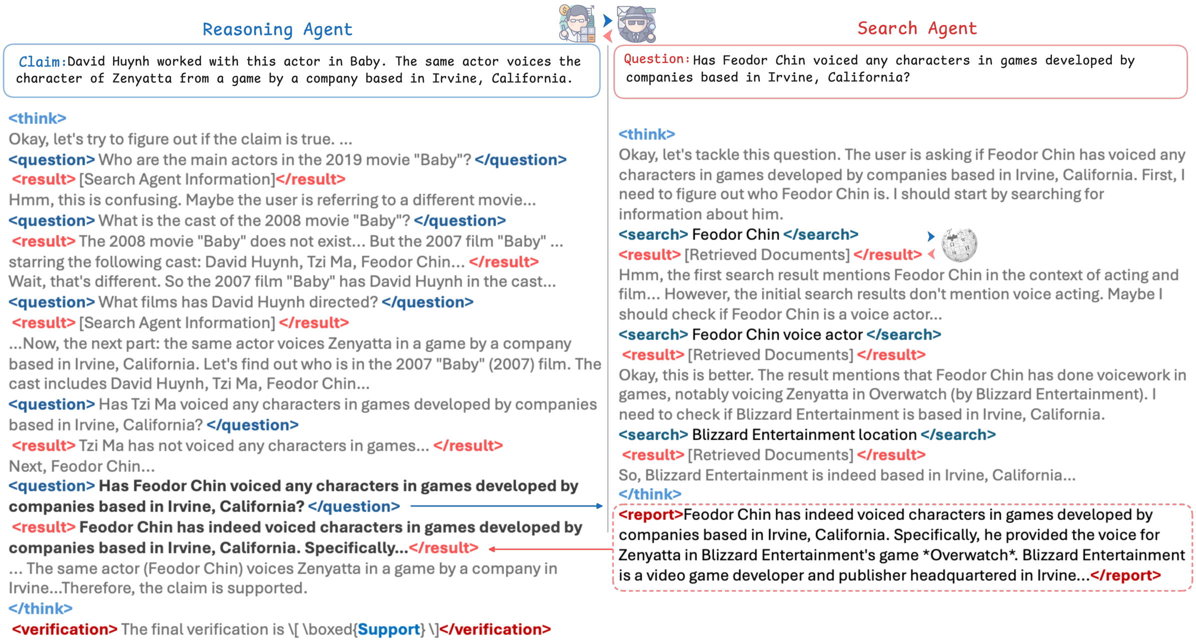
5.1 Main Result

As shown in Tables 1, our proposed method HARIS consistently outperforms all baseline methods across both the EX-FEVER and HOVER datasets, demonstrating superior multi-hop reasoning and evidence searching capabilities. Notably, HARIS achieves the highest F1 and accuracy scores across different hop counts, with particularly strong performance in the more challenging 3-hop and 4-hop settings. For example, on the HOVER dataset, HARIS achieves 62.80% accuracy in the 3-hop setting and 61.00% in the 4-hop setting, surpassing other strong GPT-4o-powered baselines. For direct comparison with Qwen3-4B results, see Table 5.

Notably, HARIS demonstrates strong generalization capabilities, achieving the best performance on the CHECKWHY benchmark. This result indicates that HARIS effectively handles more complex, causally structured claims, where gathering sufficient evidence and orchestrating multi-step reasoning is critical. This performance can be attributed to the explicit modeling of reasoning-driven searching and search-informed reasoning, which allows HARIS to dynamically refine verification paths based on partial evidence, reducing noise and improving verification consistency.

5.2 Ablations

Single-Agent vs Multi-Agent Multi-agent RL has shown strong performance in complex reasoning tasks (Wan et al., 2025), and reducing retrieval noise during intermediate steps is also known to benefit RAG systems like Search-o1 (Li et al., 2025). To assess the impact of our bi-level design, we compare HARIS with an RL-trained single-agent . As shown in Table 2, HARIS outperforms the single-agent setup across multiple datasets. This shows the advantage of decomposing reasoning and retrieval into specialized agents, each optimized for its specific role. The search agent efficiently provides relevant information to the reasoning agent, reducing noise and enhancing the verification capability. A qualitative case study in Appendix C.4 further illustrates this contrast on a shared example.



Claim: David Huynh worked with this actor in Baby. The same actor voices the character of Zenyatta from a game by a company based in Irvine, California.

<think>
Okay, let's try to figure out if the claim is true. ...
<question> Who are the main actors in the 2019 movie "Baby"? </question>
<result> [Search Agent Information]</result>
Hmm, this is confusing. Maybe the user is referring to a different movie...
<question> What is the cast of the 2008 movie "Baby"? </question>
<result> The 2008 movie "Baby" does not exist... But the 2007 film "Baby" ... starring the following cast: David Huynh, Tzi Ma, Feodor Chin... </result>
Wait, that's different. So the 2007 film "Baby" has David Huynh in the cast...
<question> What films has David Huynh directed? </question>
<result> [Search Agent Information] </result>
...Now, the next part: the same actor voices Zenyatta in a game by a company based in Irvine, California. Let's find out who is in the 2007 "Baby" (2007) film. The cast includes David Huynh, Tzi Ma, Feodor Chin...
<question> Has Tzi Ma voiced any characters in games developed by companies based in Irvine, California? </question>
<result> Tzi Ma has not voiced any characters in games... </result>
Next, Feodor Chin...
<question> **Has Feodor Chin voiced any characters in games developed by companies based in Irvine, California?** </question>
<result> **Feodor Chin has indeed voiced characters in games developed by companies based in Irvine, California. Specifically...**</result>
... The same actor (Feodor Chin) voices Zenyatta in a game by a company in Irvine...Therefore, the claim is supported.
</think>
<verification> The final verification is \[\boxed{Support} \]</verification>

Question: Has Feodor Chin voiced any characters in games developed by companies based in Irvine, California?

<think>
Okay, let's tackle this question. The user is asking if Feodor Chin has voiced any characters in games developed by companies based in Irvine, California. First, I need to figure out who Feodor Chin is. I should start by searching for information about him.
<search> Feodor Chin </search>
<result> [Retrieved Documents] </result>
Hmm, the first search result mentions Feodor Chin in the context of acting and film... However, the initial search results don't mention voice acting. Maybe I should check if Feodor Chin is a voice actor...
<search> Feodor Chin voice actor </search>
<result> [Retrieved Documents] </result>
Okay, this is better. The result mentions that Feodor Chin has done voicework in games, notably voicing Zenyatta in Overwatch (by Blizzard Entertainment). I need to check if Blizzard Entertainment is based in Irvine, California.
<search> Blizzard Entertainment location </search>
<result> [Retrieved Documents] </result>
So, Blizzard Entertainment is indeed based in Irvine, California...
</think>
<report>Feodor Chin has indeed voiced characters in games developed by companies based in Irvine, California. Specifically, he provided the voice for Zenyatta in Blizzard Entertainment's game *Overwatch*. Blizzard Entertainment is a video game developer and publisher headquartered in Irvine...</report>

Figure 6: Example of reasoning agent and search agent rollout interaction for a complex multi-hop claim.

Coordination Rounds We study how the number of coordination rounds (N) affects model performance. As shown in Table 3, increasing N generally leads to stronger performance. It shows that this approach helps balance learning dynamics and maintain alignment between high-level and low-level agents. By allowing each agent to iteratively refine its abilities while maintaining consistency, coordinated training supports more effective collaboration over multiple training cycles.

LLM-as-a-Judge vs F1 We examine the impact of the reward metric used for training the search agent using 3,600 training examples over one epoch. Specifically, we compare conventional QA F1-score and LLM-as-a-Judge as rewards. We find that F1-trained agents tend to generate more concise responses, as F1 favors answers closely matching the reference. In contrast, LLM-as-a-Judge rewards encourage more comprehensive and contextually relevant outputs. As shown in Figure 4 and 5, F1-based agents prompt the reasoning agent to trigger more searches, resulting in more follow-up questions. In comparison, LLM-as-a-Judge reduces search calls but achieves higher verification reward, indicating more thorough information improves the overall reasoning process. As shown in Table 4, the performance results suggest using LLM-as-a-Judge trained HARIS consistently improved the performance compared to F1-trained. On average it improves over 3% performance.

More experiments can be found in Appendix B.

6 Case Study

Figure 6 illustrates how HARIS resolves a complex claim through step-by-step, search-informed reasoning. The reasoning agent systematically probes plausible related actors, while the search agent continuously refines queries, shifting the search from *Feodor Chin* to *Blizzard* to gather sufficient evidence. In another example (Figure 7), the reasoning agent initially struggles to identify the correct *Baby* film, but the search agent's response about "Who is David Huynh..." provides crucial context, steering reasoning toward the correct verification. These cases highlight HARIS's collaborative process, with the reasoning agent refining its understanding as new information is retrieved until all critical connections are uncovered.

7 Conclusion

We propose Hierarchical Agent Reasoning and Information Search (HARIS), explicitly modeling the coordinated process of reasoning-driven searching and search-informed reasoning. By integrating high-level reasoning and low-level search agents, HARIS effectively captures complex reasoning chains while reducing noise in evidence retrieval. Our approach demonstrates strong performance across challenging benchmarks, highlighting its effectiveness for comprehensive claim verification.

Limitations

While HARIS demonstrates strong performance, due to limited computational resources, we train only on a 4B model. Using larger models are likely to achieve even stronger performance. Additionally, our study focuses on binary claim verification ('Support' or 'Refute'). While some benchmarks include additional classes such as 'Neutral' or 'Not Enough Info,' we do not explore them here. Claim verification is a key area within fact-checking, we do not explore tasks such as open-domain QA or counterfactual detection, as these differ substantially from multi-hop claim verification. Notably, our binary setting is consistent with strong baselines such as ProgramFC, QACheck, and FOLK in claim verification.

References

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2022. Fact checking with insufficient evidence. *Transactions of the Association for Computational Linguistics*, 10:746–763.

Jifan Chen, Grace Kim, Aniruddh Sriram, Greg Durrett, and Eunsol Choi. 2024. Complex claim verification with evidence retrieved in the wild. 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 3569–3587.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, Fan Yang, and 1 others. 2025. Research: Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*.

Yunfan Gao, Yun Xiong, Yijie Zhong, Yuxi Bi, Ming Xue, and Haofen Wang. 2025. Synergizing rag and reasoning: A systematic review. *arXiv preprint arXiv:2504.15909*.

Xinyan Guan, Jiali Zeng, Fandong Meng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, and Jie Zhou. 2025. Deeprag: Thinking to retrieval step by step for large language models. *arXiv preprint arXiv:2502.01142*.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Qisheng Hu, Quanyu Long, and Wenya Wang. 2025a. Boost: Bootstrapping strategy-driven reasoning programs for program-guided fact-checking. *arXiv preprint arXiv:2504.02467*.

Qisheng Hu, Quanyu Long, and Wenya Wang. 2025b. Decomposition dilemmas: Does claim decomposition boost or burden fact-checking performance? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6313–6336.

Hyewon Jeon and Jay-Yoon Lee. 2025. Graphcheck: Multi-path fact-checking with entity-relationship graphs. *arXiv preprint arXiv:2502.20785*.

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. HoVer: A dataset for many-hop fact extraction and claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3441–3460. Association for Computational Linguistics.

Zhengping Jiang, Jingyu Zhang, Nathaniel Weir, Seth Ebner, Miriam Wanner, Kate Sanders, Daniel Khashabi, Anqi Liu, and Benjamin Van Durme. 2024. Core: Robust factual precision scoring with informative sub-claim identification. *arXiv preprint arXiv:2407.03572*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. *CoRR*, abs/2405.13576.

Ryo Kamoi, Tanya Goyal, Juan Diego Rodriguez, and Greg Durrett. 2023. WiCE: Real-world entailment for claims in Wikipedia. In *Proceedings of EMNLP*.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. Dspy: Compiling declarative language model calls into self-improving pipelines.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Yuxuan Liu, Hongda Sun, Wenya Guo, Xinyan Xiao, Cunli Mao, Zhengtao Yu, and Rui Yan. 2025. Bidev:

Bilateral defusing verification for complex claim fact-checking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 541–549.

Yining Lu, Noah Ziems, Hy Dang, and Meng Jiang. 2025. Optimizing decomposition for optimal claim verification. *arXiv preprint arXiv:2503.15354*.

Huanhuan Ma, Weizhi Xu, Yifan Wei, Liuji Chen, Liang Wang, Qiang Liu, Shu Wu, and Liang Wang. 2024. EX-FEVER: A dataset for multi-hop explainable fact verification. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Jiatong Ma, Linmei Hu, Rang Li, and Wenbo Fu. 2025. Local: Logical and causal fact-checking with llm-based multi-agents. In *Proceedings of the ACM on Web Conference 2025*, pages 1614–1625.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of EACL*, pages 2014–2037.

OpenAI. 2024a. Gpt-4o mini: advancing cost-efficient intelligence. *OpenAI Blog*.

OpenAI. 2024b. Hello gpt-4o. *OpenAI Blog*.

Liangming Pan, Xinyuan Lu, Min-Yen Kan, and Preslav Nakov. 2023a. QACheck: A demonstration system for question-guided multi-hop fact-checking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 264–273.

Liangming Pan, Xiaobao Wu, Xinyuan Lu, Anh Tuan Luu, William Yang Wang, Min-Yen Kan, and Preslav Nakov. 2023b. Fact-checking complex claims with program-guided reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6981–7004.

Hoang Pham, Thanh-Do Nguyen, and Khac-Hoai Nam Bui. 2025. Verify-in-the-graph: Entity disambiguation enhancement for complex claim verification with interactive graph representation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5181–5197.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*.

Qwen. 2024. Qwq: Reflect deeply on the boundaries of the unknown.

Qwen. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Jiasheng Si, Yibo Zhao, Yingjie Zhu, Haiyang Zhu, Wenpeng Lu, and Deyu Zhou. 2024. CHECKWHY: Causal fact verification via argument structure. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15636–15659.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Fei Huang, and Yan Zhang. 2025. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.

Ziyu Wan, Yunxiang Li, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, and Ying Wen. 2025. Rema: Learning to meta-think for llms with multi-agent reinforcement learning. *arXiv preprint arXiv:2503.09501*.

Haoran Wang and Kai Shu. 2023. Explainable claim verification via knowledge-grounded reasoning with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6288–6304.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

Miriam Wanner, Seth Ebner, Zhengping Jiang, Mark Dredze, and Benjamin Van Durme. 2024a. A closer look at claim decomposition. In *Proceedings of the 13th Joint Conference on Lexical and Computational Semantics (*SEM 2024)*, pages 153–175.

Miriam Wanner, Benjamin Van Durme, and Mark Dredze. 2024b. Dndscore: Decontextualization and decomposition for factuality verification in long-form text generation. *arXiv preprint arXiv:2412.13175*.

Junde Wu, Jiayuan Zhu, and Yuyuan Liu. 2025. Agentic reasoning: Reasoning llms with tools for the deep research. *arXiv preprint arXiv:2502.04644*.

Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, and 1 others. 2025. Rag-gym: Optimizing reasoning and search agents with process supervision. *arXiv preprint arXiv:2502.13957*.

Fengzhu Zeng and Wei Gao. 2024. JustiLM: Few-shot justification generation for explainable fact-checking of real-world claims. *Transactions of the Association for Computational Linguistics*, pages 334–354.

Xiaoyan Zhao, Lingzhi Wang, Zhanghao Wang, Hong Cheng, Rui Zhang, and Kam-Fai Wong. 2024. Pacar: Automated fact-checking with planning and customized action reasoning using large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12564–12573.

Liwen Zheng, Chaozhuo Li, Litian Zhang, Haoran Jia, Senzhang Wang, Zheng Liu, and Xi Zhang. 2025. Mrr-fv: Unlocking complex fact verification with multi-hop retrieval and reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26066–26074.

A Experimental Settings

A.1 Dataset

HOVER(Jiang et al., 2020) is a multi-hop claim verification dataset containing 2, 3, and 4-hop data, designed to evaluate the ability of models to connect multiple pieces of evidence across different documents. EX-FEVER(Ma et al., 2024) is another multi-hop benchmark, primarily focused on 2 and 3-hop reasoning over hyperlinked Wikipedia documents.

For training, we sample 7,200 examples from the combined training sets of EX-FEVER and HOVER, maintaining an equal ratio across different hop lengths to ensure balanced coverage of multi-hop reasoning.

For evaluation, we follow Wang and Shu (2023) and use stratified sampling to select 500 instances for each hop setting, ensuring a balanced distribution of multi-hop complexity.

We also evaluate on CHECKWHY (Si et al., 2024), a challenging claim verification dataset where negative samples are constructed by modifying evidence to create counterfactuals. Given this design, we only sample from the positive claims and use accuracy for evaluation.

A.2 Retrieval Setting

We use the Wikipedia corpus processed by FlashRAG (Jin et al., 2024), which provide chunked passages. We adopt a dense retrieval method with ‘intfloat/multilingual-e5-small’ model (Wang et al., 2024), which offers a favorable balance between memory efficiency and performance on the MTEB benchmark (Muennighoff et al., 2023). For retrieval, we use a top-3 retrieval strategy.

A.3 Baselines

For consistency, we adapt each baseline to the same experimental setup wherever possible.

RAG We use the input claim as the retrieval query, providing the retrieved context and the claim to GPT-4o for final classification. The verification signature is provided in Table 10.

Decompose-Then-Verify We use the decomposition module from WICE (Kamoi et al., 2023) for breaking down complex claims into simpler sub-claims via few-shot in-context learning. Each sub-claim is then verified using the same retrieval and classification setup as RAG, with final results aggregated using logical AND.

ProgramFC (Pan et al., 2023b): We implement ProgramFC based on the official repository[5]. To ensure consistency, we replace the Flan-T5 model used for sub-task functions with GPT-4o.

Search-o1 (Li et al., 2025) Our implementation is based on the official implementation[6] and setting the maximum search limit to 10. We adapt the original QA prompt templates for claim verification.

FOLK (Wang and Shu, 2023) Our implementation follows the official repository[7]. Consistent with the original paper, we perform knowledge grounding using the Google Search API[8], ensuring accurate grounding for FOL-guided reasoning.

QACheck (Pan et al., 2023a) We use the official implementation[9]. We replace the original LLM components with GPT-4o to match our baseline settings and ensure consistent evaluation.

A.4 Search Agent Training Data Synthesis

To ensure the search agent can effectively address questions generated by the reasoning agent, we synthesize training data by having the reasoning agent perform rollouts on the training claims and sampling the generated questions.

For the first epoch training, we collect the first question proposed by the untrained reasoning agent H_0 in each rollout. This is because the initial untrained reasoning agent struggles with formatting, making longer rollouts less reliable. Starting from the second epoch, we sample from all questions generated during the rollout as the reasoning agent at this stage has developed a more stable question generation capability. For training efficiency, in postprocessing, we limit each claim verification data to a single question.

To generate answers for these sampled questions, we pair each question with the ground-truth evidence provided by the original dataset. For EX-FEVER, we use the human-annotated explanations as the evidence. The prompt signature used for this pairing is provided in Table 10.

---

[5] https://github.com/mbzuai-nlp/ProgramFC

[6] https://search-o1.github.io/

[7] https://github.com/wang2226/FOLK

[8] https://serpapi.com/

[9] https://github.com/XinyuanLu00/QACheck

[10] For Search-o1, its official implementation is specifically designed for QwQ reasoning models and is not directly configurable with Qwen3 models.

	HOVER						EX-FEVER			
	2-hops		3-hops		4-hops		2-hops		3-hops	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
RAG	57.47	57.60	49.49	52.40	50.74	53.00	68.67	68.80	68.54	68.60
Decomp-Verify	59.67	60.80	47.13	53.71	43.59	52.80	63.09	64.60	54.76	59.00
ProgramFC	55.41	55.60	50.39	51.20	51.15	52.80	56.60	56.80	56.31	57.60
BiDeV	59.38	62.40	50.55	56.20	42.09	52.60	58.73	62.20	51.60	57.60
QACheck	55.90	56.20	48.08	50.20	51.07	52.80	55.84	56.20	60.14	60.40
FOLK	61.08	61.40	59.07	59.20	57.65	58.20	80.12	80.20	73.93	74.20
HARIS	**69.31**	**69.40**	**62.33**	**62.80**	**59.84**	**61.00**	**80.12**	**80.20**	**73.93**	**74.20**

Table 5: Direct performance comparison of different methods (Qwen3-4B based) with HARIS.[10]

A.5 Training Settings

A.5.1 LLM-as-a-Judge

We use GPT-4o-mini as the judge for evaluating the final output usefulness of the search agent. This is implemented using DSPy (Khattab et al., 2024), which allows for customizing signature to define prompt-based LLM classification. The signature used in our experiments is provided in Table 10. The final score is set to 1 if the output ‘is_useful’ variable contains "yes" and 0 otherwise.

A.5.2 Hardware & Hyperparameter

All experiments, including HARIS and the baselines, were conducted on a server with 4×H20 GPUs and a cluster of 8×A100 nodes. Our implementation is based on the verl framework (Sheng et al., 2024). Key hyperparameters include: rollout group size of 5, tensor parallel size (tp) of 2, batch size of 48, temperature of 1.0, learning rate of 1e-6, and KL coefficient of 0.001.

HARIS experiments were mainly run on the 4×H20 server. Retrieval services were hosted on a single GPU using FastAPI. For reasoning agent training, we used vLLM (Kwon et al., 2023) to serve the search agent endpoint on one GPU, while the remaining GPUs were allocated to high-level agent training. Two GPUs for training the reasoning agent and one GPU for the search agent inference service. In the single-agent setting, two GPUs were used for training. Due to GPU memory constraints, we set the maximum context length to 8192 tokens.

B Additional Experiments

B.1 Direct comparison

To enable direct comparison on the same base model, we run the baselines using Qwen3-4B as the base LLM. The results are summarized in Table 5. As shown, when using the same base LLM, HARIS significantly outperforms the baselines, demonstrating its effectiveness.

B.2 Supervised Finetuning

We provide an experiment comparing HARIS with supervised finetuning (SFT). Specifically, in an explainable fact-checking setting, we enabled the thinking mode of Qwen3-4B to generate responses using the same training set as HARIS. To ensure the model learns the correct target sequence, for each claim in the training data, we repeatedly sampled responses until the final prediction was correct. The prompt used can be found in Table 9. Training epochs and learning rates were kept the same as HARIS's setting. The performance results are summarized in Table 6. Overall, HARIS outperforms supervised fine-tuning across all datasets and hop settings.

B.3 ReAct & Model Scaling

One might be concerned that decoupling the search and reasoning agents primarily compensates for the limitations of smaller models (such as our 4B backbone). However, in our main experiments, the Search-o1 baseline employs a larger 32B model, yet it still underperforms compared to HARIS. To further investigate the effect of model scaling, we implemented a ReAct LLM Agent baseline and conducted experiments using Qwen3-4B, 8B, and 14B. In this setup, the agent performs Wikipedia searches and leverages the retrieved documents as observations. The F1 results are presented in Table 7. As shown, simply increasing the model size does not always result in substantial performance gains. These results suggest that our multi-agent, decoupled approach offers distinct advantages.

	SFT		**HARIS**	
	F1	Acc	F1	Acc
EX-FEVER$_{\text{2hops}}$	71.93	72.60	80.12	80.20
EX-FEVER$_{\text{3hops}}$	63.95	66.20	73.93	74.20
HOVER$_{\text{2hops}}$	57.05	59.40	69.31	69.40
HOVER$_{\text{3hops}}$	42.03	51.40	62.33	62.80
HOVER$_{\text{4hops}}$	41.24	52.00	59.84	61.00

Table 6: Performance comparison between supervised finetuning(SFT) and HARIS.

		ReAct		**HARIS**
	4B	8B	14B	4B
EX-FEVER$_{\text{2hops}}$	72.17	71.56	72.58	80.12
EX-FEVER$_{\text{3hops}}$	67.94	63.91	70.91	73.93
HOVER$_{\text{2hops}}$	63.59	63.19	64.57	69.31
HOVER$_{\text{3hops}}$	54.94	58.19	55.76	62.33
HOVER$_{\text{4hops}}$	51.24	53.17	56.02	59.84

Table 7: F1 performance comparison between HARIS and ReAct agent based on different model sizes.

C Prompts & Examples

C.1 Prompt Template

The prompt templates used for reasoning agent and search agent are shown in Table 8.

C.2 Synthesized Training Data

Table 11 shows two examples of synthesized training data for the search agent. Example 2 leverages human-annotated explanations as evidence. The synthesis process uses a Chain-of-Thought (CoT) prompting format, with the `rationale` field capturing GPT-4o's intermediate reasoning before producing the final answer.

C.3 Synthesized Training Data

Table 11 shows two examples of synthesized training data for the search agent. Example 2 leverages human-annotated explanations as evidence. The synthesis process uses a Chain-of-Thought (CoT) prompting format, with the `rationale` field capturing GPT-4o's intermediate reasoning before producing the final answer.

C.4 Single-Agent & Multi-Agent Cases

To better understand the behavioral differences between single-agent and coordinated reasoning-search approaches, we compare two rollouts for the same claim in Figures 8 and 9.

In the single-agent case, the model issues several searches but fails to effectively refine its queries. For each aspect it explores, it stops short of deeper investigation and prematurely converges on partial evidence. With reasoning and retrieval entangled in a single generation loop, the agent lacks feedback mechanisms to reassess or adjust its direction, ultimately producing an incorrect verification.

In contrast, HARIS decouples reasoning and search into specialized agents. The reasoning agent identifies uncertain links and formulates precise questions, while the search agent iteratively gathers relevant evidence to support or refute each hypothesis. This coordinated process enables effective disambiguation, deeper exploration, and accurate multi-hop reasoning. The comparison highlights how HARIS's multi-agent design leads to more robust, interpretable verification under ambiguity and incomplete evidence.

D Human Evaluation Details

As described in Section 3.3.3, we conducted a human evaluation to assess the reliability of our LLM-as-a-Judge setup. We sampled 150 questions from a held-out set of synthesized QA data, using the trained HARIS search agent to gather information for each. Two annotators from the Prolific platform[11], each paid £20, independently evaluated 75 responses following the same guidelines as the LLM-as-a-Judge. They judged whether the retrieved information was sufficient and useful for deriving the correct answer. The results showed a Cohen's Kappa of 0.81 and a 93.3% agreement rate, indicating strong consistency. These findings confirm that our LLM-as-a-Judge metric closely aligns with human judgments. An example of the annotation panel is shown in Figure 10. For data consent, we selected the AI task annotation category on the platform, and annotators were informed that the collected data would be used to evaluate LLM outputs.

---

[11] `https://www.prolific.com/`

Prompt Template for Low-Level Search Agent
You are a helpful assistant tasked with gathering information to answer a question step by step with the help of the wikipedia search tool. Given a question, you need to think about the reasoning process in the mind and how to gather sufficient information to finally report the gathered information clearly based on the information you have found. Your task includes answering the question and reporting relevant information you have found clearly. During thinking, you can invoke the wikipedia search tool to search for fact information about specific topics if needed. The reasoning process and reported information are enclosed within <think> </think> and <report> </report> tags respectively, and the search query and result are enclosed within <search> </search> and <result> </result> tags respectively...
Prompt Template for High-Level Reasoning Agent
You are a helpful assistant tasked with verifying the truthfulness of a claim step by step, with the support of a Wikipedia search agent. Given a claim, you need to think about the reasoning process in the mind and then provide the verification result (Support or Refute). During thinking, if needed, ask factual questions to the Wikipedia search agent. This is a multi-hop claim verification task, the reasoning may involve identifying intermediate facts (bridging facts) that are not explicitly mentioned in the claim but are necessary to verify its truthfulness. For the wikipedia agent to clearly understand the question, follow these guidelines: 1. Begin the question with clear interrogatives. 2. Questions must be self-contained—do not refer to "the claim" or use vague pronouns like "it" or "that". 3. Avoid context-dependent phrases like "in the claim" or "based on that". The reasoning and questioning process should be interleaved using the following tags: - Use <think> </think> to enclose the reasoning process. - Use <question> </question> to pose a factual question. - The agent will return relevant information inside <result> </result> tags. - The final binary decision—**Support** or **Refute**—must be wrapped in LaTeX format as \boxed{Support} or \boxed{Refute} inside the <verification> tag...

Table 8: Prompt templates for the low-level search agent and high-level reasoning agent.

Prompt Template for Single Agent

You are a helpful assistant tasked with verifying the truthfulness of a claim step by step, with the help of the wikipedia search tool. Given a claim, you need to first think about the reasoning process in the mind and then provide the boolean verification result (Support or Refute). During thinking, you can invoke the wikipedia search tool to search for fact information about specific topics if needed. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags respectively, and the search query and result are enclosed within <search> </search> and <result> </result> tags respectively...

Prompt Template for Supervised Finetuning

<|im_start|>user

Given a claim and its retrieved evidence, determine whether the claim is 'Support' or 'Refute'.

Claim: claim

Evidence: retrieved_evidence

Wrap your final answer in <answer> and </answer> tags (e.g. <answer>Support</answer> or <answer>Refute</answer>)<|im_end|>

<|im_start|>assistant

<think>

Table 9: Prompt templates for the single agent and supervised finetuning.

LLM-as-a-Judge DSPy Signature

```
class SearchAgentRewardSignature(dspy.Signature):
  question: str = dspy.InputField(desc="The question for which information must be gathered")
  ground_truth_answer: str = dspy.InputField(desc="The correct answer to the question")
  gathered_information: str = dspy.InputField(desc="Information gathered by the search agent intended to help answer the question")
  is_useful: Literal["yes", "no"] = dspy.OutputField(desc="Determine whether the gathered information is sufficient and useful to derive the correct answer")
```

Pseudo Ground-Truth Answer Signature

```
class PseudoGroundTruthQA(dspy.Signature):
  claim: str = dspy.InputField()
  veracity: Literal["true", "false"] = dspy.InputField(desc="The veracity of the claim")
  evidence: dict[str, list[str]] = dspy.InputField(desc="Supporting evidence/explanation for the veracity of the claim")
  question: str = dspy.InputField(desc="A relevant question asked by a fact-checking agent")
  answer: str = dspy.OutputField(desc="The answer to the question. If no answer is applicable, return ’None’")
```

Classification Signature for RAG and Decompose-Then-Verify

```
class ClaimVerificationSignature(dspy.Signature):
  claim: str = dspy.InputField(desc="The claim to be checked")
  context: str = dspy.InputField(desc="The retrieved evidence for the claim")
  veracity: Literal[’Support’, ’Refute’] = dspy.OutputField(desc="Given the claim and the retrieved evidence, determine whether the claim is ’Support’ or ’Refute’")
```

Table 10: Signature definitions for search agent reward evaluation, search agent training data synthesis, and baselines claim verification.

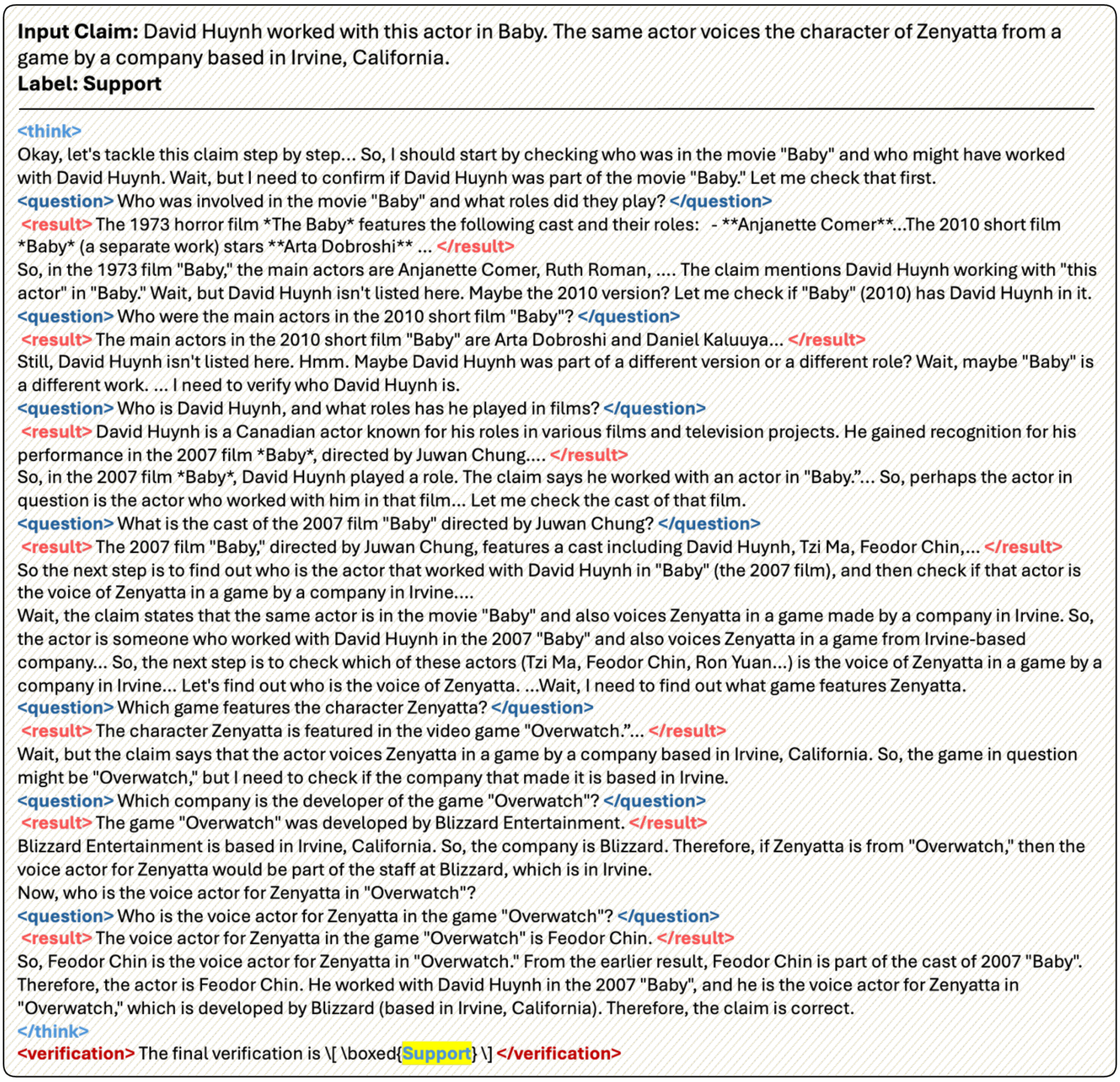
Synthesized Raw Training Data Example 1

```
{
  "input": "94 episodes were in the original run of the HBO show in which
  the voice actress of Miss Spider played the character Charlotte York
  Goldenblatt.",
  "evidence": "Miss Flora Spider is the title character in the bestselling
   children's books by David Kirk. She originally appeared in cartoon
  format in the movie \"Miss Spider's Sunny Patch Kids\" (with Brooke
  Shields providing Miss Spider's voice), which spun off the \"Miss Spider
  's Sunny Patch Friends\" series (with Kristin Davis providing her voice
  this time around), and \"Miss Spider's Sunny Patch Friends - Harvest
  Time Hop and Fly\" video game...",
  "explanation": null,
  "label": true,
  "question": "Who is the voice actress for the character \"Miss Spider
  \"?",
  "qa_response": {
    "rationale": "To produce the answer. We start by identifying the
    character \"Miss Spider\" and the voice actress associated with her.
    According to the evidence, Miss Spider is a character in the children'
    s books by David Kirk and has appeared in various media formats. The
    evidence states that Brooke Shields provided Miss Spider's voice in
    the movie \"Miss Spider's Sunny Patch Kids,\" while Kristin Davis
    provided her voice in the \"Miss Spider's Sunny Patch Friends\" series
    . Since the question asks for the voice actress for the character \"
    Miss Spider,\" we need to determine which context is being referred to
    . However, given the context of the claim and the evidence provided,
    it is most relevant to identify Kristin Davis as the voice actress for
     the series, as she is also the actress who played Charlotte York
    Goldenblatt in \"Sex and the City.\" Therefore, the answer to the
    question is Kristin Davis.",
    "answer": "Kristin Davis"
  }
}
```

Synthesized Raw Training Data Example 2

```
{
  "input": "Journey is an American rock band formed in 1974 by former
  members of band such as an American rock band formed in 1966.",
  "evidence": null,
  "explanation": "Journey is formed in 1973 by former members of band such
   as Steve Miller Band, not 1974. The Steve Miller Band is an American
  rock band formed in 1966.",
  "label": false,
  "question": "What American rock band was formed in 1966 and had members
  who later joined Journey?",
  "qa_response": {
    "rationale": "To produce the answer. We start by identifying the
    American rock band formed in 1966. According to the evidence, the
    Steve Miller Band was formed in 1966. Next, we need to determine if
    any members from this band later joined Journey. The evidence states
    that Journey was formed by former members of bands such as the Steve
    Miller Band. Therefore, the Steve Miller Band is the band in question
    .",
    "answer": "The Steve Miller Band."
  }
}
```

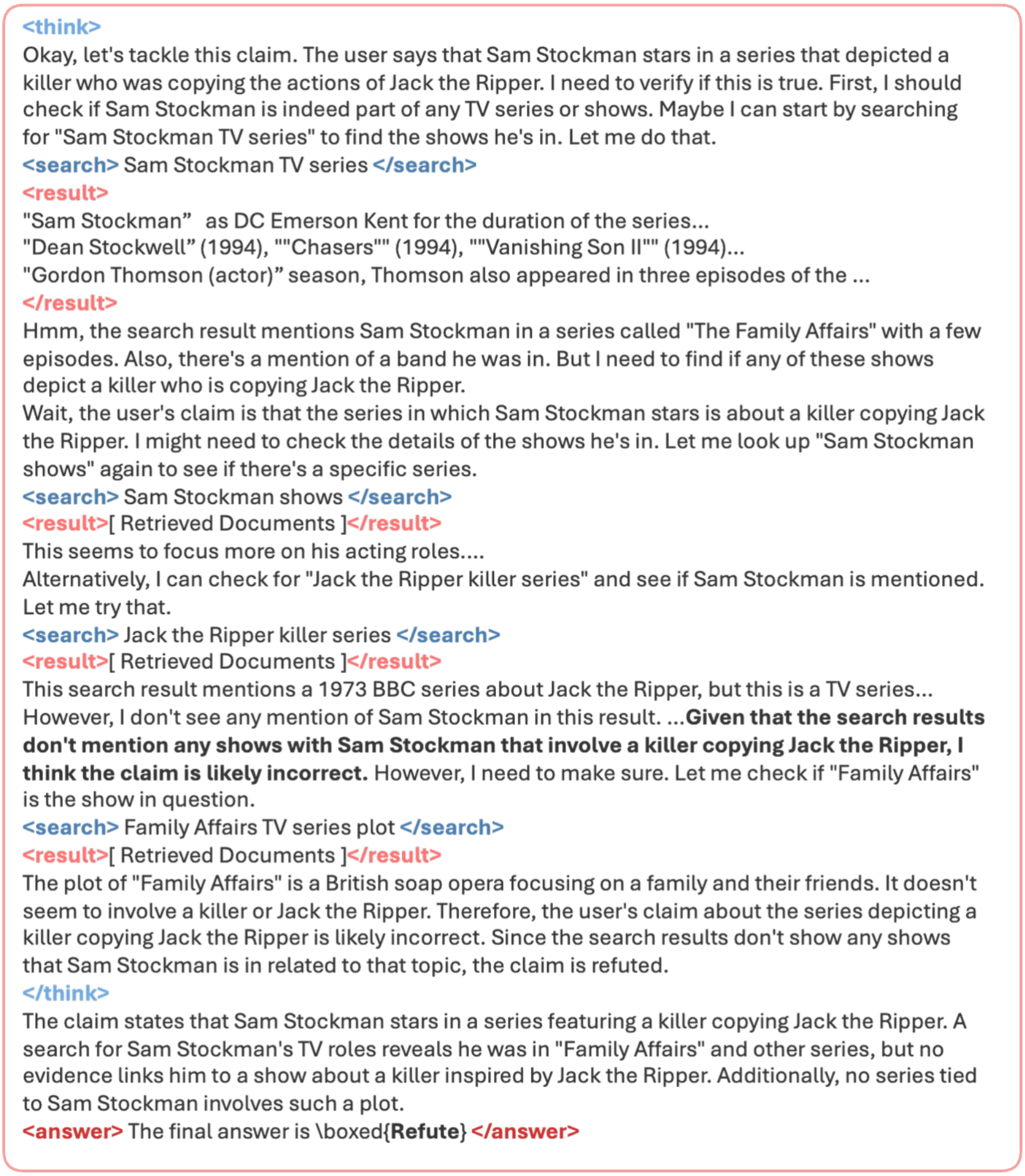
Table 11: Synthesized raw data examples for training the reasoning and search agents.



Input Claim: David Huynh worked with this actor in Baby. The same actor voices the character of Zenyatta from a game by a company based in Irvine, California.
Label: Support

<think>
Okay, let's tackle this claim step by step... So, I should start by checking who was in the movie "Baby" and who might have worked with David Huynh. Wait, but I need to confirm if David Huynh was part of the movie "Baby." Let me check that first.
<question> Who was involved in the movie "Baby" and what roles did they play? </question>
<result> The 1973 horror film *The Baby* features the following cast and their roles: - **Anjanette Comer**...The 2010 short film *Baby* (a separate work) stars **Arta Dobroshi** ... </result>
So, in the 1973 film "Baby," the main actors are Anjanette Comer, Ruth Roman, ... The claim mentions David Huynh working with "this actor" in "Baby." Wait, but David Huynh isn't listed here. Maybe the 2010 version? Let me check if "Baby" (2010) has David Huynh in it.
<question> Who were the main actors in the 2010 short film "Baby"? </question>
<result> The main actors in the 2010 short film "Baby" are Arta Dobroshi and Daniel Kaluuya... </result>
Still, David Huynh isn't listed here. Hmm. Maybe David Huynh was part of a different version or a different role? Wait, maybe "Baby" is a different work. ... I need to verify who David Huynh is.
<question> Who is David Huynh, and what roles has he played in films? </question>
<result> David Huynh is a Canadian actor known for his roles in various films and television projects. He gained recognition for his performance in the 2007 film *Baby*, directed by Juwan Chung.... </result>
So, in the 2007 film *Baby*, David Huynh played a role. The claim says he worked with an actor in "Baby."... So, perhaps the actor in question is the actor who worked with him in that film... Let me check the cast of that film.
<question> What is the cast of the 2007 film "Baby" directed by Juwan Chung? </question>
<result> The 2007 film "Baby," directed by Juwan Chung, features a cast including David Huynh, Tzi Ma, Feodor Chin,... </result>
So the next step is to find out who is the actor that worked with David Huynh in "Baby" (the 2007 film), and then check if that actor is the voice of Zenyatta in a game by a company in Irvine....
Wait, the claim states that the same actor is in the movie "Baby" and also voices Zenyatta in a game made by a company in Irvine. So, the actor is someone who worked with David Huynh in the 2007 "Baby" and also voices Zenyatta in a game from Irvine-based company... So, the next step is to check which of these actors (Tzi Ma, Feodor Chin, Ron Yuan...) is the voice of Zenyatta in a game by a company in Irvine... Let's find out who is the voice of Zenyatta. ...Wait, I need to find out what game features Zenyatta.
<question> Which game features the character Zenyatta? </question>
<result> The character Zenyatta is featured in the video game "Overwatch."... </result>
Wait, but the claim says that the actor voices Zenyatta in a game by a company based in Irvine, California. So, the game in question might be "Overwatch," but I need to check if the company that made it is based in Irvine.
<question> Which company is the developer of the game "Overwatch"? </question>
<result> The game "Overwatch" was developed by Blizzard Entertainment. </result>
Blizzard Entertainment is based in Irvine, California. So, the company is Blizzard. Therefore, if Zenyatta is from "Overwatch," then the voice actor for Zenyatta would be part of the staff at Blizzard, which is in Irvine.
Now, who is the voice actor for Zenyatta in "Overwatch"?
<question> Who is the voice actor for Zenyatta in the game "Overwatch"? </question>
<result> The voice actor for Zenyatta in the game "Overwatch" is Feodor Chin. </result>
So, Feodor Chin is the voice actor for Zenyatta in "Overwatch." From the earlier result, Feodor Chin is part of the cast of 2007 "Baby". Therefore, the actor is Feodor Chin. He worked with David Huynh in the 2007 "Baby", and he is the voice actor for Zenyatta in "Overwatch," which is developed by Blizzard (based in Irvine, California). Therefore, the claim is correct.
</think>
<verification> The final verification is \[\boxed{Support} \] </verification>

Figure 7: Different rollout example from reasoning agent for a complex multi-hop claim.



<think>

Okay, let's tackle this claim. The user says that Sam Stockman stars in a series that depicted a killer who was copying the actions of Jack the Ripper. I need to verify if this is true. First, I should check if Sam Stockman is indeed part of any TV series or shows. Maybe I can start by searching for "Sam Stockman TV series" to find the shows he's in. Let me do that.

<search> Sam Stockman TV series **</search>**

<result>

"Sam Stockman" as DC Emerson Kent for the duration of the series...
"Dean Stockwell" (1994), ""Chasers"" (1994), ""Vanishing Son II"" (1994)...
"Gordon Thomson (actor)" season, Thomson also appeared in three episodes of the ...

</result>

Hmm, the search result mentions Sam Stockman in a series called "The Family Affairs" with a few episodes. Also, there's a mention of a band he was in. But I need to find if any of these shows depict a killer who is copying Jack the Ripper.

Wait, the user's claim is that the series in which Sam Stockman stars is about a killer copying Jack the Ripper. I might need to check the details of the shows he's in. Let me look up "Sam Stockman shows" again to see if there's a specific series.

<search> Sam Stockman shows **</search>**

<result>[Retrieved Documents]**</result>**

This seems to focus more on his acting roles....

Alternatively, I can check for "Jack the Ripper killer series" and see if Sam Stockman is mentioned. Let me try that.

<search> Jack the Ripper killer series **</search>**

<result>[Retrieved Documents]**</result>**

This search result mentions a 1973 BBC series about Jack the Ripper, but this is a TV series... However, I don't see any mention of Sam Stockman in this result. ...**Given that the search results don't mention any shows with Sam Stockman that involve a killer copying Jack the Ripper, I think the claim is likely incorrect.** However, I need to make sure. Let me check if "Family Affairs" is the show in question.

<search> Family Affairs TV series plot **</search>**

<result>[Retrieved Documents]**</result>**

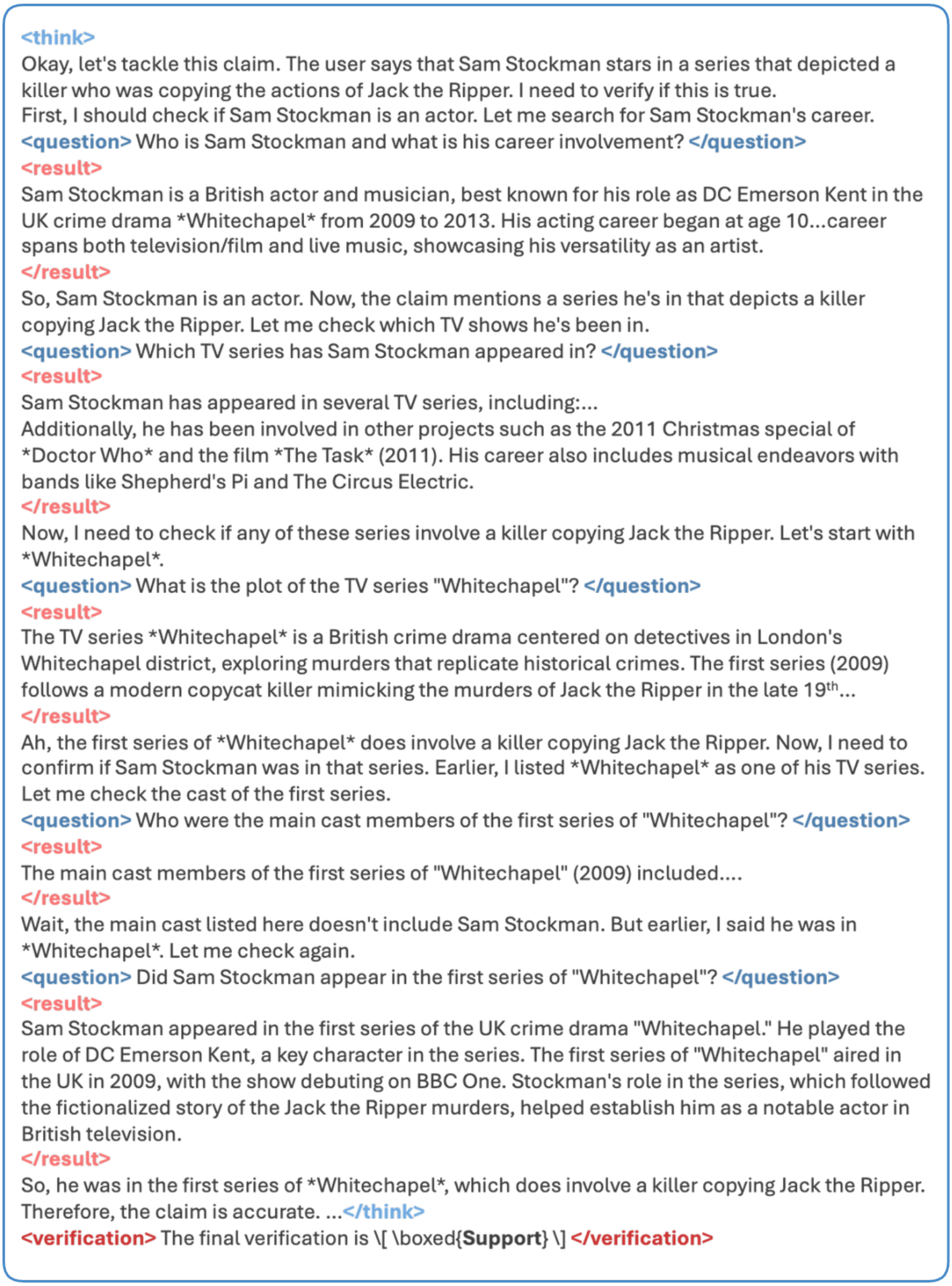
The plot of "Family Affairs" is a British soap opera focusing on a family and their friends. It doesn't seem to involve a killer or Jack the Ripper. Therefore, the user's claim about the series depicting a killer copying Jack the Ripper is likely incorrect. Since the search results don't show any shows that Sam Stockman is in related to that topic, the claim is refuted.

</think>

The claim states that Sam Stockman stars in a series featuring a killer copying Jack the Ripper. A search for Sam Stockman's TV roles reveals he was in "Family Affairs" and other series, but no evidence links him to a show about a killer inspired by Jack the Ripper. Additionally, no series tied to Sam Stockman involves such a plot.

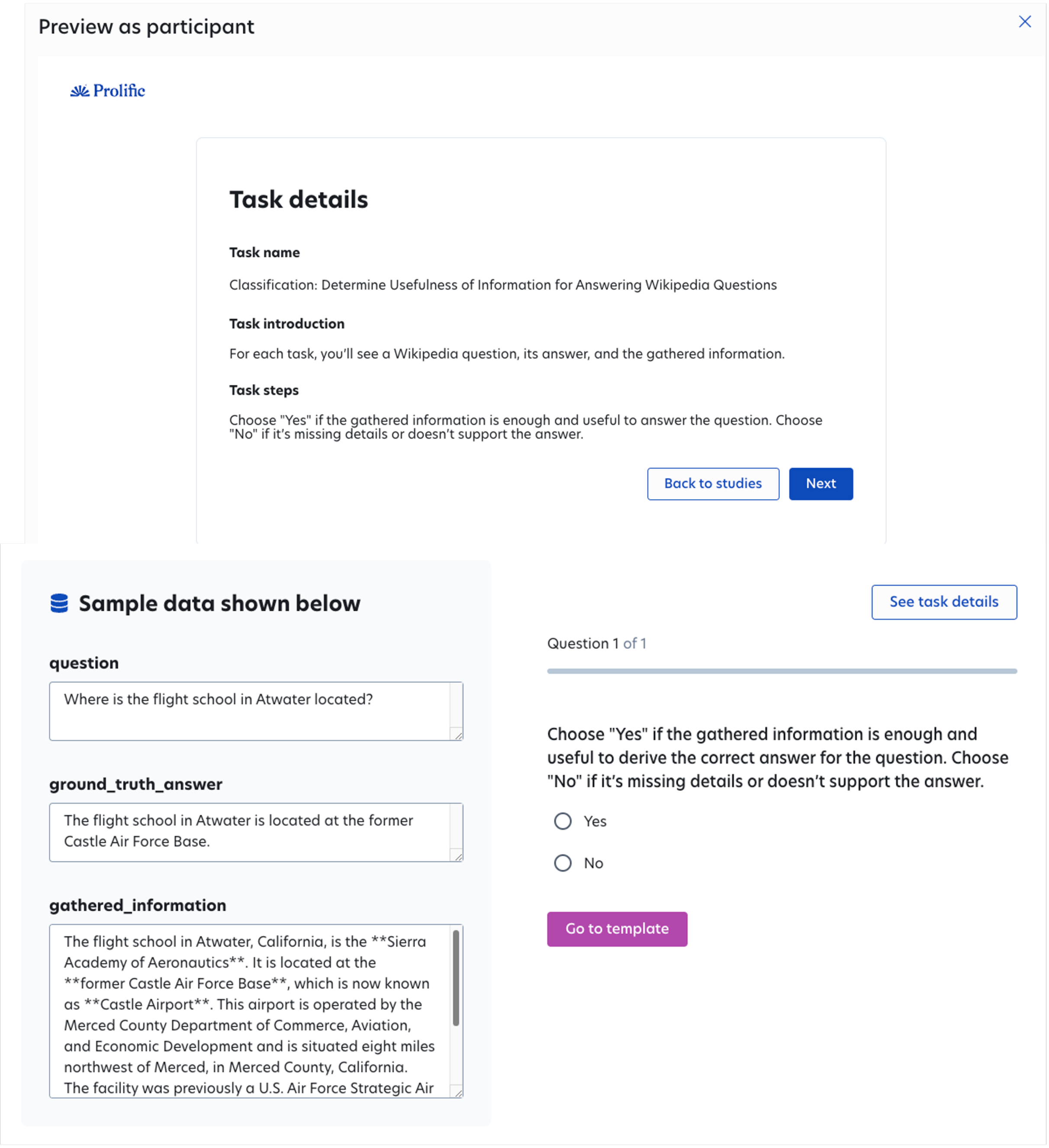
<answer> The final answer is \boxed{**Refute**} **</answer>**

Figure 8: Example of a trained single agent rollout. The input claim 'Sam Stockman stars in a series that depicted a killer. The killer was copying the actions of Jack the Ripper.' is correct. The agent issues several queries but fails to retrieve sufficient evidence. Without a clear separation of roles, it struggles to iteratively refine its search direction and prematurely converges on partial evidence, ultimately leading to an incorrect verification.



<think>
Okay, let's tackle this claim. The user says that Sam Stockman stars in a series that depicted a killer who was copying the actions of Jack the Ripper. I need to verify if this is true.
First, I should check if Sam Stockman is an actor. Let me search for Sam Stockman's career.
<question> Who is Sam Stockman and what is his career involvement? **</question>**
<result>
Sam Stockman is a British actor and musician, best known for his role as DC Emerson Kent in the UK crime drama *Whitechapel* from 2009 to 2013. His acting career began at age 10...career spans both television/film and live music, showcasing his versatility as an artist.
</result>
So, Sam Stockman is an actor. Now, the claim mentions a series he's in that depicts a killer copying Jack the Ripper. Let me check which TV shows he's been in.
<question> Which TV series has Sam Stockman appeared in? **</question>**
<result>
Sam Stockman has appeared in several TV series, including:...
Additionally, he has been involved in other projects such as the 2011 Christmas special of *Doctor Who* and the film *The Task* (2011). His career also includes musical endeavors with bands like Shepherd's Pi and The Circus Electric.
</result>
Now, I need to check if any of these series involve a killer copying Jack the Ripper. Let's start with *Whitechapel*.
<question> What is the plot of the TV series "Whitechapel"? **</question>**
<result>
The TV series *Whitechapel* is a British crime drama centered on detectives in London's Whitechapel district, exploring murders that replicate historical crimes. The first series (2009) follows a modern copycat killer mimicking the murders of Jack the Ripper in the late 19th...
</result>
Ah, the first series of *Whitechapel* does involve a killer copying Jack the Ripper. Now, I need to confirm if Sam Stockman was in that series. Earlier, I listed *Whitechapel* as one of his TV series. Let me check the cast of the first series.
<question> Who were the main cast members of the first series of "Whitechapel"? **</question>**
<result>
The main cast members of the first series of "Whitechapel" (2009) included....
</result>
Wait, the main cast listed here doesn't include Sam Stockman. But earlier, I said he was in *Whitechapel*. Let me check again.
<question> Did Sam Stockman appear in the first series of "Whitechapel"? **</question>**
<result>
Sam Stockman appeared in the first series of the UK crime drama "Whitechapel." He played the role of DC Emerson Kent, a key character in the series. The first series of "Whitechapel" aired in the UK in 2009, with the show debuting on BBC One. Stockman's role in the series, which followed the fictionalized story of the Jack the Ripper murders, helped establish him as a notable actor in British television.
</result>
So, he was in the first series of *Whitechapel*, which does involve a killer copying Jack the Ripper. Therefore, the claim is accurate. ...</think>
<verification> The final verification is \[\boxed{**Support**} \] **</verification>**

Figure 9: Example of HARIS's reasoning agent rollout. The input claim 'Sam Stockman stars in a series that depicted a killer. The killer was copying the actions of Jack the Ripper.' is correct. The reasoning and search agents coordinate effectively: the reasoning agent identifies uncertain links and delegates targeted queries, while the search agent retrieves precise evidence, enabling correct verification.



Task details

Task name

Classification: Determine Usefulness of Information for Answering Wikipedia Questions

Task introduction

For each task, you'll see a Wikipedia question, its answer, and the gathered information.

Task steps

Choose "Yes" if the gathered information is enough and useful to answer the question. Choose "No" if it's missing details or doesn't support the answer.

question

Where is the flight school in Atwater located?

ground_truth_answer

The flight school in Atwater is located at the former Castle Air Force Base.

gathered_information

The flight school in Atwater, California, is the **Sierra Academy of Aeronautics**. It is located at the **former Castle Air Force Base**, which is now known as **Castle Airport**. This airport is operated by the Merced County Department of Commerce, Aviation, and Economic Development and is situated eight miles northwest of Merced, in Merced County, California. The facility was previously a U.S. Air Force Strategic Air

Choose "Yes" if the gathered information is enough and useful to derive the correct answer for the question. Choose "No" if it's missing details or doesn't support the answer.

○ Yes

○ No

Figure 10: Preview of the human evaluation panel.