DEEPRUBRIC: Evidence-Tree Rubric Supervision for Efficient Reinforcement Learning of Deep Research Agents

Anonymous ACL submission

Abstract

Deep research agents synthesize long-form reports by searching and reasoning over retrieved evidence. Reinforcement learning with rubric-based rewards improves these agents by optimizing them against checkable criteria that translate report quality into reward signals, but its efficiency depends on whether those criteria reliably capture the task scope and evidence needs. Most existing studies ask an LLM to generate rubrics for a given query, but when the model fails to infer the underlying information needs, the generated rubrics may be incomplete and reduce RL efficiency. To obtain more reliable query–rubric supervision, we introduce DEEPRUBRIC, a data construction framework that reverses this process: instead of inferring evaluation criteria for a given query, it first determines what an evidence-backed report should be evaluated on and then synthesizes aligned query–rubric pairs from those evaluation targets. Starting from a sampled seed topic, DEEPRUBRIC builds an evidence tree by recursively expanding evidence-backed sub-questions, whose leaves serve as atomic and verifiable evaluation targets. It then uses the evidence tree to synthesize the training query and rubrics, ensuring that the reward evaluates exactly the information requested by the query. Using DEEPRUBRIC, we construct 9K query–rubric supervision examples and train DEEPRUBRIC-8B with rubric-based GRPO, achieving comparable performance to prior open state-of-the-art deep research models across three benchmarks with roughly 13× fewer RL GPU-hours. [1]

1 Introduction

Deep research agents synthesize long-form reports by searching open corpora and reasoning over retrieved documents (OpenAI, 2025; Li et al., 2026b; Shi et al., 2025; Hu et al., 2025). Unlike short-form question answering, these agents must identify relevant sources, integrate evidence across documents, support claims with citations, and organize findings into reports that satisfy complex information needs (Yifei et al., 2025; Sharma et al., 2025; Du et al., 2025; Li et al., 2026a; Han et al., 2025). Recently, reinforcement learning (RL) with rubric-based rewards has become an increasingly common way to improve such systems (Gunjal et al., 2025; Shen et al., 2026; Shao et al., 2025). In this paradigm, rubrics decompose report quality into checkable criteria, including coverage of important sub-questions, appropriate use of evidence, and citation-supported claims, and then translate them into reward signals to optimize agents (Ruan et al., 2025; Viswanathan et al., 2026). The effectiveness of this training paradigm therefore depends on whether the rubric faithfully captures the task scope and evidence needs behind the research query.

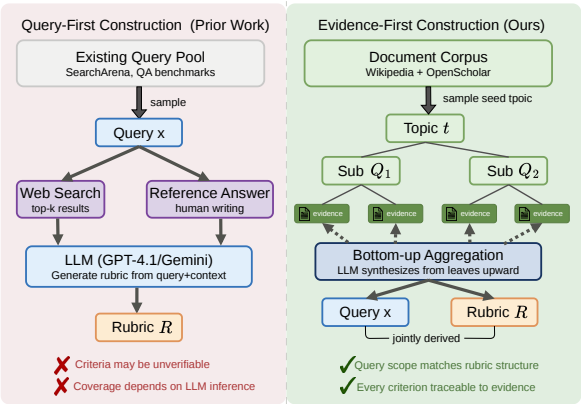

Figure 1: **Query-first vs. Evidence-first rubric construction.** Query-first pipelines infer rubrics from a given query; DEEPRUBRIC builds an evidence tree first and derives both the query and rubric from the same structure, making criteria traceable and query-aligned.

Ensuring such rubric faithfulness is especially important for expensive and time-consuming RL systems, because each rollout may require multi-step tool use and long-form report generation, and

[1] Code is available on Anonymous GITHUB.

prior systems have required thousands to tens of thousands of GPU-hours (Shao et al., 2025; Gao et al., 2025). However, most existing pipelines construct rubrics in a query-first manner in Figure 1. Given a user query, they ask a strong language model or human annotator to infer evaluation criteria, often using retrieved documents or reference answers as auxiliary context. This query-first design is natural, but it is also underdetermined. A user query rarely reveals the full evidence landscape, and auxiliary context offers only a partial view of what should be evaluated. As a result, generated rubrics may miss important answerable aspects or include weakly supported criteria, yielding noisy rewards that waste costly rollouts.

To improve the efficiency of deep research RL with more reliable reward supervision, we introduce DEEPRUBRIC, a data construction framework that derives training queries and rubric criteria from the same information structure. Unlike prior pipelines that infer rubrics for a given query, DEEPRUBRIC first determines what an evidence-backed report should be evaluated on, and then derives both the rubric and the training query from these evaluation requirements. Concretely, starting from a sampled seed topic, it recursively expands related sub-questions and grounds them in retrieved documents until reaching fine-grained information needs, which form the leaf nodes of an evidence tree. These leaf nodes are then converted into atomic rubric criteria, while the training query is synthesized bottom-up from the same tree into a coherent long-form research task. By deriving the rubric and query from the same tree, DEEPRUBRIC makes each criterion traceable to supporting evidence and keeps the generated query aligned with what the reward evaluates.

Using DEEPRUBRIC, we construct 9K query–rubric examples from local corpora Wikipedia and OpenScholar without manual expert annotation and train DEEPRUBRIC-8B with rubric-based GRPO (Shao et al., 2024). During RL rollouts, the agent retrieves only from the same local corpora used to construct the training tasks, rather than calling external web search APIs. With only 750 GPU-hours of RL training, DEEPRUBRIC-8B matches or surpasses the closest open rubric-RL baseline, DR Tulu-8B (Shao et al., 2025), while using roughly 13× fewer RL GPU-hours. Although DEEPRUBRIC constructs supervision from local corpora, the learned behavior is not tied to a fixed retrieval environment. At evaluation time, we replace the local retriever with public online search tools and evaluate on out-of-domain open-retrieval deep research benchmarks, including AstaBench-ScholarQA-CS2 (Bragg et al., 2025), ResearchQA (Yifei et al., 2025), and DeepResearchBench (Du et al., 2025). DEEPRUBRIC-8B achieves competitive or superior performance against prior open deep research models, outperforming larger open models with up to 32B parameters, including Tongyi-DR-30B (Team et al., 2025) and WebThinker-32B (Li et al., 2026b). It also narrows the gap to proprietary systems such as OpenAI Deep Research.

Our contributions are as follows: (i) we introduce DEEPRUBRIC, an evidence-first framework for constructing aligned query–rubric supervision from raw corpora; (ii) we train an 8B deep research model with 9K constructed examples, reaching strong open-retrieval performance with only 750 GPU-hours of RL training; and (iii) we show through ablations and analyses that evidence-tree construction, bottom-up query synthesis, and query–rubric alignment are key to the gains.

2 Related Work

Deep research agents. Search-augmented agents are increasingly expected to move beyond short factual answers and produce long-form, evidence-grounded research reports. Early work mainly studies short-answer or bounded search settings, improving search-augmented reasoning, exploration, and asynchronous RL in interactive environments (Jin et al., 2025; Zheng et al., 2025; Liu et al., 2025b; Gao et al., 2025). Recent deep research systems extend this direction to long-form report generation through specialized workflows and report-writing agents (Li et al., 2026b; Hu et al., 2025; Li et al., 2026c; Yang et al., 2026), larger multi-tool agents (Team et al., 2025), and synthetic information-seeking queries or trajectories (Sun et al., 2025; Wu et al., 2026; Tao et al., 2025; Chen et al., 2026). These works construct agents, workflows, or behavior traces for producing research reports, but RL still requires training tasks paired with reliable criteria for judging the reports. DEEPRUBRIC focuses on constructing training data for deep research, where each task is paired with rubric-based supervision for long-form answers.

Rubrics for evaluation and RL Rubrics provide a practical interface for evaluating open-ended generation by decomposing answer quality into

checkable criteria. They are used in long-form evaluation benchmarks (Ruan et al., 2025; Bragg et al., 2025; Yifei et al., 2025; Sharma et al., 2025; Li et al., 2026a), and automatic methods generate or refine criteria from task descriptions, retrieved evidence, reference answers, trajectories, or model outputs (Shen et al., 2026; Ma et al., 2025; Jia et al., 2025; Gao et al., 2026; Wadhwa et al., 2025). Rubric rewards further enable RL without exact verifiers: Rubrics as Rewards trains with generated criteria (Gunjal et al., 2025), DR Tulu evolves search-grounded rubrics from policy rollouts (Shao et al., 2025), and related work adapts criteria or critics online (Rezaei et al., 2025; Wu et al., 2025). These criteria generators improve how a given task is judged, but they are largely query-first: they start from a fixed prompt, question, response set, or coarse rubric and infer evaluation criteria afterward. DEEPRUBRIC instead treats evidence-backed information needs as the starting point, building an evidence tree first and co-generating the training query and rubric from the same structure.

3 Method

We introduce DEEPRUBRIC, an evidence-first pipeline that constructs query-rubric supervision from document corpora for RL post-training of deep research agents. Given a corpus, DEEPRUBRIC first builds an evidence tree by recursively expanding a sampled seed topic into evidence-grounded sub-queries (§3.1), then co-synthesizes a training query and verifiable rubric criteria from the same tree (§3.2), and finally uses the rubric as the primary content reward during RL post-training (§3.3). We depart from the usual query-first pipeline by first identifying corpus-verifiable information needs and then formulating a query around them, rather than starting from a query and inferring evaluation criteria afterward.

3.1 Evidence Tree Construction

Task representation. We represent each training task as a rooted evidence tree $\mathcal{T} = (\mathcal{V}, \mathcal{E}, v_0)$ with root node v_0. Each node $v \in \mathcal{V}$ contains three components: (1) a sub-query q_v that specifies what information this node seeks; (2) a set of retrieved documents $\mathcal{P}_v = \{p_{v,1}, \ldots, p_{v,k_v}\}$ that provide evidence relevant to q_v; and (3) a set of children $\mathrm{Ch}(v)$, each representing a finer-grained sub-query derived from q_v. A node is called a *leaf* if it has no children: $\mathcal{L}(\mathcal{T}) = \{v \in \mathcal{V} : \mathrm{Ch}(v) = \emptyset\}$. The tree is a recursive decomposition of a broad topic (the root v_0) into progressively more specific, evidence-grounded sub-queries.

Root initialization. Given a corpus $\mathcal{C}$ and retriever ρ, we sample a seed topic t from $\mathcal{C}$, convert it into a broad research sub-query q_{v_0}, and retrieve documents $\mathcal{P}_{v_0} \leftarrow \rho(q_{v_0})$.

Top-down expansion. The tree is built by breadth-first expansion. For each frontier node v, an LLM $\mathcal{M}$ observes its sub-queryq_v, retrieved documents$\mathcal{P}_v$, and ancestor path $\mathrm{Anc}(v)$, and proposes complementary child sub-queries that decompose q_v into finer-grained aspects:

$$\widetilde{\mathcal{Q}}_v = \mathcal{M}\big(q_v, \mathcal{P}_v, \mathrm{Anc}(v)\big). \tag{1}$$

For each proposed child sub-query $\tilde{q} \in \widetilde{\mathcal{Q}}_v$, we retrieve its document set $\tilde{\mathcal{P}} = \rho(\tilde{q})$ and ask $\mathcal{M}$ to ground the child node by selecting the documents that support answering $\tilde{q}$. The resulting grounded pair $(\tilde{q}, \tilde{\mathcal{P}})$ is added as a child of v. If $\mathcal{M}$ cannot propose further meaningful child sub-queries, or if the proposed children cannot be grounded in retrieved evidence, v becomes a leaf.

To prevent unbounded growth, we impose structural budgets on the evidence tree: a maximum depth of 3, with depth-aware branching that allows up to 6 children at the root, 4 at intermediate levels, and 3 at deeper levels. The output of this stage is an evidence tree $\mathcal{T}$, a hierarchical decomposition of the topic into progressively finer sub-queries, each grounded in retrieved documents. Further details are provided in Appendix C.2 and Algorithm 1.

3.2 Query-Rubric Co-Generation and Revision

The tree $\mathcal{T}$ captures the information structure of a topic, but training an RL agent requires a concrete query-rubric pair $(x, \mathcal{R}(\mathcal{T}))$. We co-generate both by presenting $\mathcal{T}$ to an LLM $\mathcal{M}$, which synthesizes them in a bottom-up pass guided by the tree, then verify the result with a separate model.

Bottom-up synthesis. Given the tree $\mathcal{T}$, we prompt $\mathcal{M}$ to reason from the leaves upward, progressively merging leaf-level content into more abstract summaries at each internal node (the full prompt is provided in Appendix C.2). At the root, the model produces two outputs: (1) a natural research query x that integrates the information captured by the tree, and (2) a set of rubric criteria $\mathcal{R}(\mathcal{T})$ derived from the leaf content. The merge

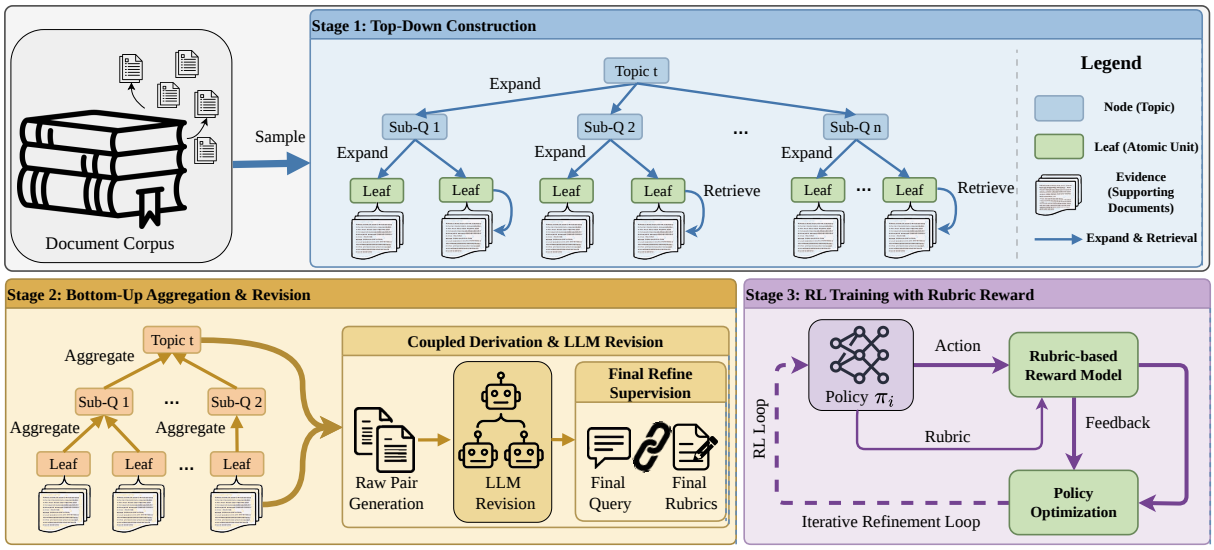

Figure 2: **Method overview. Stage 1**: An evidence tree is constructed top-down from a document corpus; a root topic is recursively decomposed into sub-queries, each grounded in retrieved evidence. **Stage 2**: The tree is aggregated bottom-up to co-generate a query–rubric pair $(x, \mathcal{R}(\mathcal{T}))$; a verification-and-revision step checks evidence support, query–rubric alignment, and criterion quality. **Stage 3**: The policy π_θ generates reports conditioned on the query, a rubric-based reward model scores them against $\mathcal{R}(\mathcal{T})$, and the policy is updated via GRPO.

process makes the query's scope follow the evidence structure of $\mathcal{T}$ rather than being generated independently.

Rubric structure. Each criterion is a typed, weighted, evidence-grounded check:

$$r = (c_r, \mathcal{P}_r, \tau_r, w_r), \qquad r \in \mathcal{R}(\mathcal{T}), \tag{2}$$

where c_r is a natural-language verifiable criterion, $\tau_r \in \{\text{FACTUAL}, \text{LOGICAL}\}$ is its type, and $w_r \in [0, 1]$ is its importance weight. For FACTUAL criteria, $\mathcal{P}_r$ contains supporting documents from selected leaves and the report should state information supported by those documents. For LOGICAL criteria, $\mathcal{P}_r = \emptyset$ because the criterion evaluates synthesis, comparison, or reasoning over the report rather than the mention of a specific evidence fact. The model may discard peripheral or redundant leaves during synthesis, so $\mathcal{R}(\mathcal{T})$ selects the criteria that jointly define a well-scoped, answerable research task.

Verification and revision. To improve the consistency of the synthesized supervision, we apply a separate verifier $\mathcal{M}_{\text{verify}}$ to check each tuple $(x, \mathcal{T}, \mathcal{R}(\mathcal{T}))$:

$$(d, x', \mathcal{R}') = \mathcal{M}_{\text{verify}}(x, T, \mathcal{R}(\mathcal{T})), \tag{3}$$

where $d \in \{\text{KEEP}, \text{REVISE}, \text{DROP}\}$ denotes the verifier decision. The verifier checks evidence support, query–rubric scope alignment, and criterion quality, including atomicity and redundancy. For KEEP examples, we retain the original tuple; for REVISE examples, we use the corrected query x' and rubric $\mathcal{R}'$; and for DROP examples, we remove the tuple from training. We provide the full verification prompt together with an analysis of revision and drop decisions in Appendix C.3 and C.4.

3.3 RL with Rubric Rewards

We use the verified query–rubric pairs as reward supervision for RL post-training. Given a training query x, the policy π_θ samples reports $y \sim \pi_\theta(\cdot \mid x)$. We use standard GRPO for optimization, and the reward is computed by applying the synthesized rubric to each sampled report y.

Rubric reward. For each criterion $r \in \mathcal{R}(\mathcal{T})$, we use LLM-as-a-judge to score whether the generated report y satisfies the criterion statement c_r. The score is assigned on a 0-4 integer scale and normalized to $s_r(y) \in [0, 1]$. The full scoring prompt is provided in Appendix A.1. The rubric reward is the importance-weighted mean:

$$R_{\text{rubric}}(y) = \frac{\sum_{r \in \mathcal{R}(\mathcal{T})} w_r \cdot s_r(y)}{\sum_{r \in \mathcal{R}(\mathcal{T})} w_r}, \tag{4}$$

where $s_r(y) \in [0, 1]$ is the normalized judge score for criterion r.

Composite reward. Following prior deep-research RL training (Shao et al., 2025), we

combine the rubric reward with auxiliary rewards for format, citation quality, and tool use:

$$R(y) = 0.5 \cdot R_{\text{rubric}}(y) + 0.2 \cdot R_{\text{format}}(y) + 0.2 \cdot R_{\text{cite}}(y) + 0.1 \cdot R_{\text{search}}(y), \quad (5)$$

where R_{format} rewards adherence to the expected output structure (<think>, <tool_call>, <answer> tags), R_{cite} measures per-claim citation quality over cited sources, and R_{search} encourages sufficient use of retrieval tools during generation. The rubric reward receives the largest weight because it provides the main signal for evidence-grounded content synthesis. Full reward definitions and details are provided in Appendix A.

Optimization. We optimize the policy with Group Relative Policy Optimization (GRPO; Shao et al., 2024), sampling a group of candidate reports for each query and computing advantages relative to the group. Model initialization, training size, and hyperparameters are reported in Appendix B.

4 Experiments

We evaluate DEEPRUBRIC from three perspectives. First, we test whether evidence-tree supervision improves open-retrieval deep research performance (§4.2). Second, we measure how efficiently the model reaches competitive performance under RL (§4.3). Third, we analyze which construction choices affect supervision quality through ablations, and training-data analysis (§4.4–§4.6).

4.1 Experimental Settings

Datasets and Evaluation Metrics. We evaluate deep research agents on three long-form, open-retrieval benchmarks spanning scientific and general research domains: **AstaBench-ScholarQA-CS2** (SQAv2; Bragg et al., 2025), which assesses synthesis over up-to-date scientific literature; **ResearchQA** (Yifei et al., 2025), which covers academic research questions across diverse fields; and **DeepResearch Bench** (DRB; Du et al., 2025), which evaluates open-ended research tasks requiring comprehensive reports with citations. All benchmarks follow their official evaluation protocols. The details are provided in Appendix E.1.

Baselines. We organize baselines into three groups in Table 1: (1) non-rubric open systems, including Qwen3-8B/QwQ-32B RAG (Yang et al., 2025; Team, 2025), Search-R1 (Jin et al., 2025), ASearcher (Gao et al., 2025), WebExplorer (Liu et al., 2025c), WebThinker and its report-mode variants (Li et al., 2026b), Tongyi DeepResearch (Team et al., 2025), and Ai2 ScholarQA (Singh et al., 2025b); (2) matched rubric-RL systems, especially DR Tulu-8B (Shao et al., 2025), which shares our Qwen3-8B scale and GRPO-style training but uses query-conditioned rubrics; and (3) Proprietary reference systems, including Perplexity, Gemini, GPT-5 (Singh et al., 2025a), and OpenAI Deep Research. Open baselines use official code and recommended configurations; additional setup details are in Appendix E.2.

Implementation Details.

Data construction. We build evidence trees from two source corpora: Wikipedia [2] and OpenScholar (Asai et al., 2024)[3]. To control annotation cost, we use DeepSeek-V3.2 (Liu et al., 2025a) for the high-volume stages, including evidence-tree expansion from seed topics and query–rubric synthesis from each tree. This yields approximately 10K candidate pairs. We reserve GPT-5.1 (Singh et al., 2025a) for verification, which checks evidence support, query–rubric alignment, and criterion redundancy. After revision, 9,064 pairs are retained and used for SFT and RL training.

Training. We initialize from Qwen3-8B (Yang et al., 2025). To adapt the model to the expected tool-use and report format, we perform a lightweight cold-start SFT stage on our 200 generated queries. GPT-5.1 annotates reference trajectories for these queries, including reasoning steps, search actions, and final report writing, and we fine-tune for 2 epochs (∼3 GPU-hours). We then train the SFT checkpoint with GRPO on the remaining 8,886 query–rubric pairs for 140 steps, corresponding to approximately 1 epochs (∼750 GPU-hours), using Qwen3.5-35B-A3B as the rubric reward judge. All training uses 8×A100 80GB GPUs; hyperparameters are reported in Appendix B.

Tool environment. During training, the agent uses three local tools backed by the same Wikipedia and OpenScholar corpora used for data construction: `search` performs dense retrieval over the local Wikipedia corpus, `browse` fetches full article text from the local Wikipedia database, and `scholar` performs dense retrieval over OpenScholar papers. This local environment avoids external API calls during RL rollouts, reducing

[2] https://huggingface.co/datasets/inclusionAI/ASearcher-Local-Knowledge

[3] https://huggingface.co/datasets/OpenSciLM/OpenScholar-DataStore-V3

Model	Backbone	SQAv2					ResearchQA	DRB					Average
		Overall	Rubric	Answer	Cite-P	Cite-R		Overall	Comp.	Depth	Instr.	Read.	
Proprietary Deep Research													
Claude-Sonnet Search	Claude Sonnet	–	–	–	–	–	64.3*	34.5*	39.0	37.7	45.8	41.5	–
Perplexity-Sonar (High)	Proprietary	–	–	–	–	–	69.1*	40.7*	37.4	36.1	45.7	44.7	–
Perplexity Deep Research	Proprietary	67.3	91.6	92.7	47.3	37.6	75.3*	42.3*	40.7	39.3	46.4	44.3	61.6
Gemini Deep Research	Proprietary	–	–	–	–	–	68.5*	48.8*	48.5	48.5	49.2	49.4	–
Gemini 3 Pro + Search	Gemini 3 Pro	69.8	83.1	98.3	68.5	29.4	74.3	46.3	43.4	44.9	49.8	49.0	63.5
GPT-5 + Search	GPT-5	74.8	92.3	93.8	67.8	45.6	78.2†	50.7	49.7	51.5	51.6	48.5	67.9
OpenAI Deep Research	o3/o4	79.6	91.5	95.6	77.4	43.1	79.2†	46.9*	46.8	45.2	49.2	47.1	68.6
Naive RAG													
Qwen3-8B + RAG	Qwen3-8B	40.4	69.2	92.3	–	–	56.1	33.3	29.4	27.0	40.2	41.1	43.3
QwQ-32B + RAG	QwQ-32B	41.9	77.5	90.3	–	–	60.9	40.3	38.1	34.8	47.0	44.6	47.7
Open Deep Research Models													
Search-R1-7B	Qwen2.5-7B	22.2	9.7	79.0	–	–	27.9	9.5	5.2	2.1	18.6	16.8	19.9
ASearcher-Web-7B	Qwen2.5-7B	26.9	13.7	94.0	–	–	19.4	7.8	5.1	1.7	15.2	11.8	18.0
WebExplorer-8B	Qwen2.5-8B	42.5	78.6	91.4	–	–	64.8	36.7	33.7	28.5	45.7	42.2	48.0
WebThinker-32B-DPO	Qwen2.5-32B	32.9	36.7	94.9	–	–	48.6	23.3	19.7	12.3	36.8	26.3	34.9
Tongyi DeepResearch-30B-A3B	Qwen2.5-30B	46.5	89.5	**96.4**	–	–	66.7	40.6	39.1	34.3	46.8	**45.4**	51.3
Fixed Pipeline Deep Research													
WebThinker QwQ-32B (report)	QwQ-32B	45.2	86.4	94.3	–	–	72.8	37.9	36.2	32.6	43.2	42.9	52.0
WebThinker-32B-DPO (report)	Qwen2.5-32B	46.7	91.2	95.5	–	–	74.2	40.6	39.4	35.4	46.0	43.5	53.8
Ai2 ScholarQA	Claude Sonnet	87.7	88.1	89.1	92.4	81.2	75.0†	36.1	35.1	32.0	40.5	38.9	66.3
Open Deep Research													
Qwen3-8B + Search	Qwen3-8B	57.2	42.8	92.1	53.7	40.3	46.3	18.2	14.3	8.7	29.5	24.4	40.6
DR Tulu-8B (SFT)	Qwen3-8B	72.3	81.4	91.0	65.3	51.6	68.5	39.0	36.3	35.3	45.5	39.5	59.9
DR Tulu-8B (1000-step, ≈6000h)	Qwen3-8B	86.7	84.8	95.4	90.6	**76.1**	71.1	41.8	39.5	39.4	47.3	41.6	66.5
DR Tulu-8B (1900-step, ≈9700h)	Qwen3-8B	**86.8**	**89.6**	95.4	88.6	**73.7**	74.3	43.4	**41.7**	**41.8**	48.2	41.3	68.2
Ours (SFT)	Qwen3-8B	79.5	78.4	94.2	83.9	61.4	64.8	38.0	33.8	36.0	43.5	40.1	60.8
Ours (75-step, ≈402h)	Qwen3-8B	85.1	85.3	93.7	89.3	72.1	74.2	41.9	39.7	40.0	46.9	41.5	67.1
Ours (140-step, ≈750h)	Qwen3-8B	86.0	86.4	94.0	**91.5**	72.2	**75.2**	**43.6**	**41.7**	**41.8**	**48.4**	42.5	**68.3**

Table 1: **Main results with expanded SQAv2 and DRB columns.** The average is computed over the displayed overall scores (SQAv2, ResearchQA, and DRB) when all three are available. Rows with gray background use proprietary models. Green highlights our method. 🤗 indicates open model weights and training data; indicates open training code. **Bold numeric values** mark the best score among open models for that column. *Scores reported by original benchmark authors. †Evaluated on 100-sample subset. "–" = not reported.

latency, eliminating per-step search costs, and improving run-to-run stability. For evaluation, we follow prior work (Shao et al., 2025) and use the same live tools for fair comparison: `search` calls Google Search via the Serper API, `browse` reads web pages through Jina Reader, and `scholar` queries Google Scholar via Serper API.

4.2 Main Results

Table 1 reports performance across the three main open-retrieval benchmarks. Relative to Qwen3-8B with the same search setting, DEEPRUBRIC-8B raises the three-benchmark average from 40.6 to 68.3, with gains on SQAv2 (57.2→86.0), ResearchQA (46.3→75.2), and DRB (18.2→43.6). The SFT checkpoint reaches a 60.8 average, and the 75-step RL checkpoint reaches 67.1 after roughly 402 GPU-hours. The consistent improvement across benchmarks suggests that the gains are not confined to a single benchmark-specific metric.

Comparison with open models. DEEPRUBRIC-8B achieves the strongest average among open deep research systems in our evaluation. Relative to the Qwen3-8B search-only base agent, it raises the three-benchmark average from 40.6 to 68.3, with large gains on SQAv2 (57.2→86.0), ResearchQA (46.3→75.2), and DRB (18.2→43.6). The closest comparison is DR Tulu-8B (RL), which uses the same Qwen3-8B backbone family, GRPO-style training, and rubric-reward interface, but evolves search-grounded rubrics from policy rollouts. In contrast, DEEPRUBRIC uses the fixed rubrics generated and verified before RL, yet reaches a similar average (68.3 vs. 68.2) with 140 RL steps and 750 GPU-hours, compared with DR Tulu's 1,900-step, 9,700 GPU-hour run. DEEPRUBRIC is higher on ResearchQA (75.2 vs. 74.3) and DRB (43.6 vs. 43.4), but slightly lower on SQAv2 (86.0 vs. 86.8), where DR Tulu has higher rubric coverage and citation recall. It attains the best open-model scores on ResearchQA, DRB overall, and DRB instruction following, while also improving SQAv2 citation precision over DR Tulu (91.5 vs. 88.6).

Variant	SQAv2	ResearchQA	DRB	Avg.
Ours (full)	85.1	74.2	41.9	67.1
w/o revision	83.8	73.2	39.9	$65.6_{-2.2\%}$
Search-based rubrics	80.6	72.1	37.3	$63.3_{-5.7\%}$
Closed-book rubrics	83.2	70.4	40.7	$64.8_{-3.4\%}$

Table 2: **Ablation on rubric construction (75 steps).**

Comparison with Proprietary systems. DEEPRUBRIC-8B is competitive with proprietary systems on scientific QA, while the strongest proprietary systems remain ahead on broader report generation. On SQAv2, DEEPRUBRIC exceeds the proprietary deep-research references in the upper block, including GPT-5 + Search and OpenAI Deep Research; the only higher score in the table comes from Ai2 ScholarQA, a fixed pipeline built on Claude Sonnet. On ResearchQA and DRB, the best proprietary systems still lead, especially on DRB report-quality dimensions such as comprehensiveness, depth, and readability. This pattern suggests that evidence-tree supervision can make an 8B open model highly competitive on rubric- and citation-oriented scientific QA, while open-ended report generation still benefits from stronger base models and retrieval infrastructure. The expanded SQAv2 and DRB columns help localize these effects: rubric coverage, comprehensiveness, and depth are closest to our supervision signal, whereas citation precision/recall, instruction-following, and readability also depend on auxiliary rewards and inference-time formatting choices.

4.3 Training Efficiency Analysis

Step efficiency. Figure 3 compares checkpoint performance as a function of RL training steps. The figure includes three trajectories: DEEPRUBRIC (green, ours), DR Tulu (pink, full RLER with SFT cold-start), and No SFT (grey, DR Tulu-style RL from the base model without SFT). DEEPRUBRIC reaches strong performance rapidly: by 140 steps it achieves 86.0 on SQAv2 and 43.6 on DRB, matching or exceeding DR Tulu's 1,900-step endpoint (86.8 and 43.4 respectively) with 13.6× fewer RL updates. We continued training to 285 steps to test whether further updates help; DRB remains stable (43.1/43.6/43.8 at steps 190/235/285), while SQAv2 slightly decreases (86.3→86.0→84.6). This decline mirrors a trend also observed in DR Tulu: later checkpoints produce longer answers with more claims, but citation recall drops when some claims are not adequately

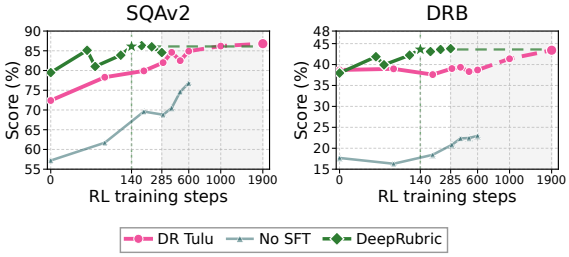

Figure 3: **Training-step efficiency.** DEEPRUBRIC reaches competitive performance with fewer RL steps.

Method	Data annotation	SFT	RL training	GPUs	Est. cost
DR Tulu-8B	GPT-5 (16K traj.)†	136 GPU-hrs	9,700 GPU-hrs	8–16×H100	≥$30K
Ours (8B)	$180 (API)	3 GPU-hrs	750 GPU-hrs	8×A100	∼$1.7K

Table 3: **Training cost comparison.**

grounded. Given the marginal returns beyond 140 steps, we stop at 285 steps and select the 140-step checkpoint for final evaluation. We attribute this early plateau to corpus saturation: within the fixed local retrieval environment, the policy largely learns to exploit the bounded evidence space early.

Training cost. Table 3 summarizes the end-to-end cost. Our total training budget is ∼$1.7K (including $180 in API calls, 3 GPU-hours for SFT, and 750 GPU-hours for RL on 8×A100), compared to ≥$30K for DR Tulu—a ∼17× reduction. GPU costs are estimated using rental server pricing. The reduction is driven by higher reward signal quality per rollout: evidence-tree rubrics provide denser, more structured feedback, enabling the policy to converge in 140 steps rather than 1,900, which directly translates to ∼13× fewer RL GPU-hours.

4.4 Ablation Study

To isolate the effect of evidence-tree construction, we conduct a fixed-query counterfactual ablation: the training queries are kept unchanged, while the rubrics are regenerated without access to the original evidence tree. This setup tests whether the tree structure provides supervision beyond what can be recovered from the final query alone. We compare three variants: **Closed-book rubrics**, which generate criteria directly from the query; **Search-based rubrics**, which first retrieve passages using the query and then generate criteria from the retrieved context; and **w/o revision**, which keeps the tree-based rubrics but removes the LLM-based verification pass. For cost efficiency, all variants are trained for 75 GRPO steps under the same recipe. The full pipeline outperforms both query-

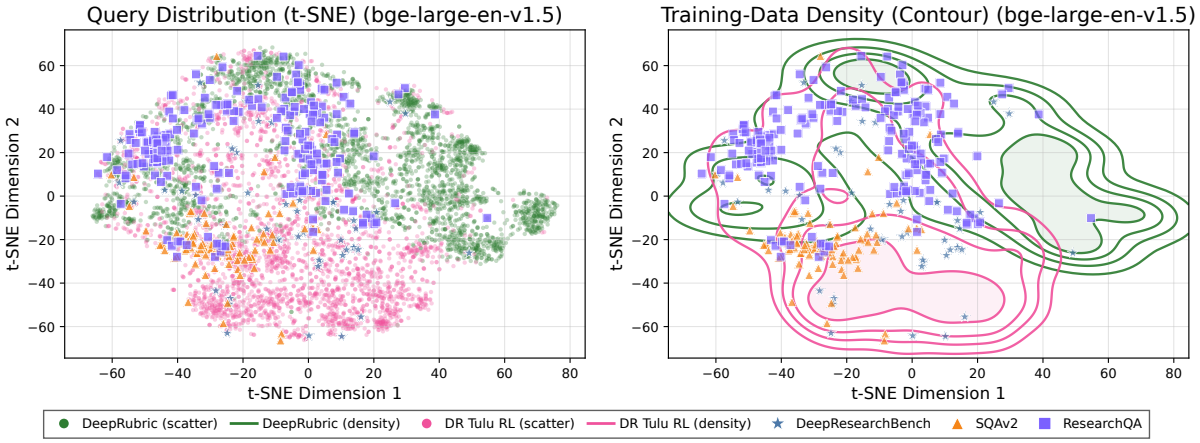

Figure 4: **Training query distribution.** BGE-large embeddings projected with t-SNE. Left: scatter plot of training queries and benchmark queries. Right: density contours of training data with benchmark queries overlaid.

first rubric baselines by 1.8–3.8 points on average, showing that adding retrieved passages to query-first rubric generation does not recover the benefit of tree-structured decomposition.

4.5 Training Data Analysis

Construction statistics. Starting from 9,838 evidence trees (depth 3, ∼55 nodes each), the LLM-based verification produces 9,064 retained query–rubric pairs (92.1% retention rate): among these, 91.5% are revised for improved query–rubric alignment and 0.6% are kept unchanged; the remaining 7.9% are dropped. Each retained pair contains on average 7.0 rubric criteria (58.4% factual, 41.6% logical) grounded in 5.0 selected evidence leaves with 2.9 passages per leaf. Full statistics are reported in Appendix C.3.

Semantic Distribution. Figure 4 projects BGE-large query embeddings for DEEPRUBRIC, DR Tulu RL, and three evaluation benchmarks via t-SNE. The two training sets occupy complementary regions despite targeting the same task, and their density shapes differ: DEEPRUBRIC covers a wider area with lower peak density, reflecting greater query diversity from multi-leaf evidence-tree synthesis; DR Tulu RL concentrates in a narrower region, reflecting more homogeneous single-document queries. The benchmark overlays connect this to downstream performance: SQAv2 queries fall within the DR Tulu RL density region (both methods perform comparably there), while ResearchQA and DRB queries have greater presence near the DEEPRUBRIC-dense region, consistent with our gains on these benchmarks. This suggests the advantage of evidence-tree co-generation is *structural*—it produces more diverse training queries that teach the model to decompose complex information needs—rather than merely topical.

4.6 Case Study

Table 8 presents a representative DRB case, including the prompt, abridged final answers, and DRB scores. The prompt requires a multi-faceted analysis of investments, products, client cases, strategy, talent programs, and future trends, rather than a flat enumeration of firms. DR Tulu-8B mainly organizes its response by first identifying a set of firms and then accumulating firm-specific evidence, yielding a broad inventory but making the cross-firm implications less explicit. In contrast, DEEPRUBRIC-8B begins with a higher-level pattern and concludes by assigning firms to comparative roles, such as platform builders, large-scale adopters, and strategy/governance specialists.

5 Conclusion

We introduced DEEPRUBRIC, an evidence-first framework for constructing query–rubric supervision for reinforcement learning of deep research agents. By deriving both training queries and reward criteria from the same evidence tree, DEEPRUBRIC grounds rubrics in retrieved documents and aligns rewards with the intended task scope. Experiments on three open-retrieval benchmarks show that an 8B model trained with DEEPRUBRIC approaches the strongest open rubric-RL baseline with substantially fewer RL steps and GPU-hours, while ablations and additional evaluations further validate the effectiveness of evidence-tree construction for reward supervision.

Limitations

DEEPRUBRIC is limited by the corpora used to construct supervision. Our current data is built from Wikipedia and OpenScholar, which provide broad factual and scholarly coverage but may not fully support specialized domains such as clinical, legal, or proprietary enterprise research. Moreover, because training queries are synthesized from corpus evidence rather than sampled from real user demand, the resulting distribution is corpus-shaped and may not cover all deployed deep-research scenarios equally well, especially those involving private data, domain-specific standards, or rapidly changing information.

References

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, and 1 others. 2024. Openscholar: Synthesizing scientific literature with retrieval-augmented lms. *arXiv preprint arXiv:2411.14199*.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D Hwang, Peter Jansen, Varsha Kishore, and 1 others. 2025. Astabench: Rigorous benchmarking of ai agents with a scientific research suite. *arXiv preprint arXiv:2510.21652*.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Jimmy Lin, Akari Asai, and Victor Zhong. 2026. Agentir: Reasoning-aware retrieval for deep research agents. *arXiv preprint arXiv:2603.04384*.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. 2025. Deepresearch bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*.

Shanghua Gao, Yuchang Su, Pengwei Sui, Curtis Ginder, and Marinka Zitnik. 2026. Qworld: Question-specific evaluation criteria for llms. *arXiv preprint arXiv:2603.23522*.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*.

Janghoon Han, Heegyu Kim, Changho Lee, Dahm Lee, Min Hyung Park, Hosung Song, Stanley Jungkyu Choi, Moontae Lee, and Honglak Lee. 2025. Deer: A benchmark for evaluating deep research agents on expert report generation. *arXiv preprint arXiv:2512.17776*.

Chen Hu, Haikuo Du, Heng Wang, Lin Lin, Mingrui Chen, Peng Liu, Ruihang Miao, Tianchi Yue, Wang You, Wei Ji, and 1 others. 2025. Step-deepresearch technical report. *arXiv preprint arXiv:2512.20491*.

Mengzhao Jia, Zhihan Zhang, Ignacio Cases, Zheyuan Liu, Meng Jiang, and Peng Qi. 2025. Autorubric-r1v: Rubric-based generative rewards for faithful multimodal reasoning. *arXiv preprint arXiv:2510.14738*.

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, and 1 others. 2025. Verltool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*.

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*.

Ruizhe Li, Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. 2026a. Deepresearch bench ii: Diagnosing deep research agents via rubrics from expert report. *arXiv preprint arXiv:2601.08536*.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. 2026b. Webthinker: Empowering large reasoning models with deep research capability. *Advances in Neural Information Processing Systems*, 38:120091–120131.

Yishan Li, Wentong Chen, Yukun Yan, Mingwei Li, Sen Mei, Xiaorong Wang, Kunpeng Liu, Xin Cong, Shuo Wang, Zhong Zhang, and 1 others. 2026c. Agentcpm-report: Interleaving drafting and deepening for open-ended deep research. *arXiv preprint arXiv:2602.06540*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, and 1 others. 2025b. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, and 1 others. 2025c. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*.

Linyue Ma, Yilong Xu, Xiang Long, and Zhi Zheng. 2025. An efficient rubric-based generative verifier for search-augmented llms. *arXiv preprint arXiv:2510.14660*.

OpenAI. 2025. Deep research system card.

MohammadHossein Rezaei, Robert Vacareanu, Zihao Wang, Clinton Wang, Bing Liu, Yunzhong He, and Afra Feyza Akyürek. 2025. Online rubrics elicitation from pairwise comparisons. *arXiv preprint arXiv:2510.07284*.

Jie Ruan, Inderjeet Nair, Shuyang Cao, Amy Liu, Sheza Munir, Micah Pollens-Dempsey, Tiffany Chiang, Lucy Kates, Nicholas David, Sihan Chen, and 1 others. 2025. Expertlongbench: Benchmarking language models on expert-level long-form generation tasks with structured checklists. *arXiv preprint arXiv:2506.01241*.

Rulin Shao, Akari Asai, Shannon Zejiang Shen, Hamish Ivison, Varsha Kishore, Jingming Zhuo, Xinran Zhao, Molly Park, Samuel G Finlayson, David Sontag, and 1 others. 2025. Dr tulu: Reinforcement learning with evolving rubrics for deep research. *arXiv preprint arXiv:2511.19399*.

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

Manasi Sharma, Chen Bo Calvin Zhang, Chaithanya Bandi, Clinton Wang, Ankit Aich, Huy Nghiem, Tahseen Rabbani, Ye Htet, Brian Jang, Sumana Basu, and 1 others. 2025. Researchrubrics: A benchmark of prompts and rubrics for evaluating deep research agents. *arXiv preprint arXiv:2511.07685*.

William F Shen, Xinchi Qiu, Chenxi Whitehouse, Lisa Alazraki, Shashwat Goel, Francesco Barbieri, Timon Willi, Akhil Mathur, and Ilias Leontiadis. 2026. Rethinking rubric generation for improving llm judge and reward modeling for open-ended tasks. *arXiv preprint arXiv:2602.05125*.

Zhengliang Shi, Yiqun Chen, Haitao Li, Weiwei Sun, Shiyu Ni, Yougang Lyu, Run-Ze Fan, Bowen Jin, Yixuan Weng, Minjun Zhu, and 1 others. 2025. Deep research: A systematic survey. *arXiv preprint arXiv:2512.02038*.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, and 1 others. 2025a. Openai gpt-5 system card. Technical report.

Amanpreet Singh, Joseph Chee Chang, Dany Haddad, Aakanksha Naik, Jena D Hwang, Rodney Kinney, Daniel S Weld, Doug Downey, and Sergey Feldman. 2025b. Ai2 scholar qa: Organized literature synthesis with attribution. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 513–523.

Shuang Sun, Huatong Song, Yuhao Wang, Ruiyang Ren, Jinhao Jiang, Junjie Zhang, Fei Bai, Jia Deng, Wayne Xin Zhao, Zheng Liu, and 1 others. 2025. Simpledeepsearcher: Deep information seeking via web-powered reasoning trajectory synthesis. *arXiv preprint arXiv:2505.16834*.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, and 1 others. 2025. Webshaper: Agentically data synthesizing via information-seeking formalization. *arXiv preprint arXiv:2507.15061*.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, and 1 others. 2025. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*.

Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Sherry Wu. 2026. Checklists are better than reward models for aligning language models. *Advances in Neural Information Processing Systems*, 38:114728–114754.

Manya Wadhwa, Zayne Sprague, Chaitanya Malaviya, Philippe Laban, Junyi Jessy Li, and Greg Durrett. 2025. Evalagent: Discovering implicit evaluation criteria from the web. *arXiv preprint arXiv:2504.15219*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhenglin Wang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Robert Tang, and 1 others. 2026. Webdancer: Towards autonomous information seeking agency. *Advances in Neural Information Processing Systems*, 38:120957–120985.

Mian Wu, Gavin Zhang, Sewon Min, Sergey Levine, and Aviral Kumar. 2025. Rlac: Reinforcement learning with adversarial critic for free-form generation tasks. *arXiv preprint arXiv:2511.01758*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhaorui Yang, Bo Pan, Han Wang, Yiyao Wang, Xingyu Liu, Luoxuan Weng, Yingchaojie Feng, Haozhe Feng, Minfeng Zhu, Bo Zhang, and 1 others. 2026. Multimodal deepresearcher: Generating text-chart interleaved reports from scratch with agentic framework. 40(40):34368–34377.

Li S Yifei, Allen Chang, Chaitanya Malaviya, and Mark Yatskar. 2025. Researchqa: Evaluating scholarly question answering at scale across 75 fields with

survey-mined questions and rubrics. *arXiv preprint arXiv:2509.00496*.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. 2025. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. pages 414–431.

Appendix

A Reward Function Details

We provide additional details of each reward component in the composite reward (Eq. 5).

A.1 Rubric Scoring

For each leaf criterion in the evidence tree, we use an LLM judge (Qwen3.5-35B-A3B) to evaluate whether the generated report adequately addresses the criterion. The judge receives the original query, the model's response, and a single criterion (including criterion ID, type, description, and supporting evidence for factual criteria), and returns a score on an integer scale from 0 to 4. The score is normalized to $[0, 1]$ by dividing by 4. We adopt a structured rubric scoring prompt adapted from DR Tulu (Shao et al., 2025), shown in Figure 5.

Rubric Scoring Judge Prompt

```
[System]
  You will be given a question (in <question>
  tags), an answer (in <response> tags), and
  a single criterion (in <criterion> tags).
  Judge only how well the answer satisfies
  that specific criterion.

  The criterion can be of different types:
  - Factual: judge whether the answer provides
  the specific, correct, and relevant factual
  content required.  Do not reward merely
  mentioning related topics.
  - Logical: judge whether the answer performs
  the specific reasoning required—comparison,
  distinction, synthesis, or conclusion.

  Return ONLY a JSON object {"score": x} where
  x is an integer from 0 to 4.

  Scoring guidelines:
  4:  Fully  satisfies.  All  essential
  components explicitly, correctly addressed.
  3: Mostly satisfies.  Minor omissions or
  ambiguities.
  2:  Partially  satisfies.  One  or  more
  essential parts missing.
  1: Minimally satisfies.  Mostly generic,
  weak, or poorly connected.
  0: Does not satisfy.  Required content
  missing or incorrect.

  [User]
  <question>{question}</question>
  <response>{answer}</response>
  <criterion>{criterion_text}</criterion>
```

Figure 5: **Rubric scoring judge prompt.** For factual criteria, the criterion text includes the criterion type, description, and supporting evidence from the evidence tree, enabling the judge to verify factual claims against specific sources.

The per-criterion scores are aggregated into a single rubric reward via a weighted average over all criteria:

$$R_{\text{rubric}}(y) = \frac{\sum_{r \in \mathcal{R}(T)} w_r \cdot s_r(y)}{\sum_{r \in \mathcal{R}(T)} w_r}, \qquad (6)$$

where $s_r(y) \in [0, 1]$ is the normalized judge score for criterion r and w_r is its importance weight.

A.2 Format Reward

The format reward $R_{\text{format}}(y) \in [0, 1]$ checks whether the model output follows the expected structured format. It is computed as a weighted sum of four binary indicators:

$$\begin{aligned} R_{\text{format}}(y) = 0.5 I_{\text{answer}}(y) + 0.2 I_{\text{cite}}(y) \\ + 0.1 I_{\text{tool}}(y) + 0.2 I_{\text{think}}(y), \end{aligned} \qquad (7)$$

where $I_{\text{answer}}(y)$ indicates the presence of properly formed `<answer>...</answer>` tags containing the final report, $I_{\text{cite}}(y)$ indicates the presence of inline citations, $I_{\text{tool}}(y)$ indicates at least one valid tool call was issued, and $I_{\text{think}}(y)$ indicates the model produced reasoning blocks. The largest weight is assigned to the answer tag, as it is the most critical structural requirement.

A.3 Search Turn Reward

The search turn reward $R_{\text{search}}(y) \in [0, 1]$ encourages the agent to issue a sufficient number of search queries during report generation:

$$R_{\text{search}}(y) = \min\left(\frac{N_{\text{search}}(y)}{U}, 1.0\right), \qquad (8)$$

where $N_{\text{search}}(y)$ is the number of valid tool calls (across `search`, `browse`, and `scholar` tools) extracted from `<tool_call>` JSON blocks, and $U = 6$ is the upper bound. The reward saturates at U calls to avoid rewarding excessive searching.

A.4 Citation Reward

The citation reward $R_{\text{cite}}(y) \in [0, 1]$ evaluates the quality of inline citations in the generated report, adapted from DR Tulu (Shao et al., 2025). Unlike simple citation-format checks, we employ a per-claim evaluation framework that assesses both the *content quality* of citations (do they support the claims?) and the *structural quality* (are they well-distributed and properly formatted?).

A.4.1 Claim Extraction

We first segment the generated answer into individual claims. Each `<cite id="ID">text</cite>`

tag defines a cited claim paired with its citation IDs; text outside cite tags is treated as an uncited claim. This yields a set of claims $\mathcal{C} = \{(c_i, \mathcal{S}_i)\}_{i=1}^{|\mathcal{C}|}$, where c_i is the claim text and $\mathcal{S}_i \subseteq \mathcal{D}$ is the set of cited source IDs (empty for uncited claims), and $\mathcal{D}$ is the citation store of all retrieved snippets.

A.4.2 Per-Claim F_1 Score

For each claim c_i we compute recall and precision scores using an LLM judge (same Qwen3.5-35B-A3B used for rubric scoring):

Citation recall. For cited claims ($\mathcal{S}_i \neq \varnothing$), we concatenate the source texts and ask the judge to assess support on a three-point scale:

$$\text{recall}(c_i) = \begin{cases} 1.0 & \text{if } \textit{Fully supported} \\ 0.5 & \text{if } \textit{Partially supported} \\ 0.0 & \text{if } \textit{No support} \end{cases} \quad (9)$$

For uncited claims ($\mathcal{S}_i = \varnothing$), we instead ask whether the claim *requires* a citation. If the judge determines the claim is a factual statement that needs grounding, the recall score is 0 (penalizing the missing citation); otherwise it is 1 (the claim is a transition, summary, or reasoning step that does not need citation).

Citation precision. For cited claims, the judge assesses whether the cited snippets contain key information relevant to the claim:

$$\text{precision}(c_i) = \begin{cases} 1.0 & \text{if } \textit{Relevant} \\ 0.0 & \text{if } \textit{Irrelevant} \end{cases} \quad (10)$$

Uncited claims receive a precision of 1 (no irrelevant citation was introduced).

Per-claim F_1. The per-claim score combines recall and precision via the harmonic mean:

$$F_1(c_i) = \frac{2 \cdot \text{recall}(c_i) \cdot \text{precision}(c_i)}{\text{recall}(c_i) + \text{precision}(c_i)}. \quad (11)$$

The average per-claim F_1 across all claims is:

$$\bar{F}_1 = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} F_1(c_i). \quad (12)$$

A.4.3 Citation Format Reward

Beyond content quality, we reward well-structured citation usage via the citation format reward R_{fmt}. This score captures three aspects:

Citation ID validity. Let $\mathcal{A}$ be the set of all citation IDs referenced in the answer. The ID validity ratio is $v = |\{a \in \mathcal{A} : a \in \mathcal{D}\}|/|\mathcal{A}|$, measuring the fraction of cited IDs that resolve to actual retrieved snippets.

Meaningful claim ratio. Among cited claims, we filter out trivially short or symbol-dominated spans. The meaningful claim ratio m is the fraction of cited claim texts that pass lexical quality checks (at least 4 lexical units and 18 characters). The base quality is $Q = v \cdot m$.

Citation distribution. We compute the normalized position of each unique valid citation within the answer text and score the distribution on three sub-metrics: (i) *span*: the range covered by citations (normalized by 0.6); (ii) *uniformity*: deviation from ideal even spacing; and (iii) *center*: proximity of mean citation position to 0.5. The distribution score is $d = 0.4 \cdot \text{span} + 0.4 \cdot \text{uniformity} + 0.2 \cdot \text{center}$.

Citation count. The count score rewards using a sufficient number of distinct valid citations: $n = \min(|\{a \in \mathcal{A} : a \in \mathcal{D}\}|/6,\ 1.0)$.

The final citation format reward is:

$$R_{\text{fmt}} = Q \cdot (0.7 + 0.1 \cdot d + 0.2 \cdot n). \quad (13)$$

A.4.4 Final Citation Reward

The overall citation reward combines content quality and structural quality:

$$R_{\text{cite}}(y) = 0.6 \cdot \bar{F}_1 + 0.4 \cdot R_{\text{fmt}}. \quad (14)$$

The citation judge prompts are shown in Figures 6–8.

Citation Recall Prompt (Cited Claims)

```
You are an expert in evaluating text quality.
  You will receive a user's question, a
  factual statement from an AI assistant's
  response, and a snippet from the document.
  Assess whether this statement is supported
  by the snippet.
  - [[Fully supported]]: Most information in
  the statement is supported by or extracted
  from the snippet. Statement and snippet
  parts are almost identical.
  - [[Partially supported]]: More than half
  of the content is supported, but a small
  portion is not mentioned or contradicts the
  snippet.
  - [[No support]]: The statement is largely
  unrelated, or most key points do not align.
  Provide: "Rating: [[...]] Analysis: ..."
  <question>{question}</question>
  <statement>{statement}</statement>
```

```
<snippet>{cited_snippets}</snippet>
```

Figure 6: **Citation recall judge prompt** for cited claims with source snippets.

Citation Recall Prompt (Uncited Claims)

```
You are an expert in evaluating citation coverage for source-grounded QA.

You will receive: (1) the user's question, (2) the assistant's final answer, (3) one sentence from the final answer.

Determine whether the sentence is a factual claim that should be grounded with an explicit citation.

Answer [[Yes]] if the sentence introduces a concrete factual claim, empirical result, dataset/model detail, number, comparison, or attribution.
Answer [[No]] if the sentence is only an introductory phrase, transition, restatement of the question, or summary following already cited claims.

Output: "Need Citation: [[Yes/No]] Analysis: ..."

<question>{question}</question>
<final_answer>{full_response}</final_answer>
<statement>{statement}</statement>
```

Figure 7: **Citation recall judge prompt** for uncited claims. Claims judged as needing citation but lacking one receive a recall score of 0.

Citation Precision Prompt

```
You are an expert in evaluating text quality. Assess whether the snippet contains key information of the statement.

- [[Relevant]]: Some key points of the statement are supported by or extracted from the snippet.
- [[Irrelevant]]: The statement is almost unrelated to the snippet.

Provide: "Rating: [[...]] Analysis: ..."

<question>{question}</question>
<statement>{statement}</statement>
<snippet>{cited_snippets}</snippet>
```

Figure 8: **Citation precision judge prompt.** Evaluates whether each cited snippet is relevant to the claim it supports.

B Training Details

Our training infrastructure is built on `verl-tool` (Jiang et al., 2025), an open-source framework for training tool-augmented language models with reinforcement learning. The reward scoring pipeline is adapted from DR Tulu (Shao et al., 2025). All training runs use 8×A100 80GB GPUs on a single node.

Hyperparameter	Value
Base model	Qwen3-8B
SFT data size	200 examples
Annotation model	GPT-5.1
Number of training epochs	2
Learning rate	4×10^{-5}
Learning rate scheduler	cosine
Warmup ratio	0.1
Per-device batch size	1
Gradient accumulation steps	16
Max sequence length	16,384
Data type	BF16
Weight decay	0.0

Table 4: **Hyperparameters for SFT training.**

B.1 SFT Hyperparameters

We provide the hyperparameters used during the SFT cold-start stage in Table 4.

B.2 RL Hyperparameters

We provide the hyperparameters used during GRPO training in Table 5.

C Data Construction Details

We provide additional details of the evidence-tree data construction pipeline described in §3.1.

C.1 Source Corpora

We build evidence trees from two source corpora:

- **Wikipedia (2018 dump):** We use the December 2018 Wikipedia dump as a general-knowledge retrieval corpus. Seed topics are sampled from a diverse set of Wikipedia categories spanning science, history, technology, and social sciences.
- **OpenScholar** (Asai et al., 2024): We use the OpenScholar corpus as a scientific-literature retrieval source. Seed topics are sampled from academic research areas to ensure coverage of scientific deep research tasks.

Both corpora are indexed and exposed to the agent during data construction and RL training via three local retrieval tools: `search` (keyword search over the corpus), `browse` (full-text reading of a retrieved document), and `scholar` (semantic search over scientific papers). During benchmark evaluation, the agent switches to the live search, browsing, and scholar tools described in §4.1.

Hyperparameter	Value
RL algorithm	GRPO
Base checkpoint	SFT checkpoint
Training data	9,064 query-rubric pairs
Unique prompts per batch	8
Rollouts per prompt (group size)	8
Batch size	64
Mini-batch size	4
Max prompt length	2,096 tokens
Max response length	28,288 tokens
Max action length per tool call	2,048 tokens
Max observation length per tool call	4,096 tokens
Max tool call turns	20
Temperature	1.0
Top-p	1.0
Learning rate	1×10^{-6}
Learning rate schedule	constant
KL penalty coefficient	0.0
Entropy coefficient	0.0
Total GRPO steps	140
Checkpoint save frequency	every 5 steps
Data type	BF16
Weight decay	0.0
Parallelism strategy	FSDP (param + optimizer offload)
vLLM GPU memory utilization	0.6
Rubric judge model	Qwen3.5-35B-A3B
Reward weights (Eq. 5)	
Rubric weight (α_{rubric})	0.5
Format weight (α_{format})	0.2
Citation weight (α_{cite})	0.2
Search weight (α_{search})	0.1

Table 5: **Hyperparameters for GRPO training.** Training is performed on 8×A100 80GB GPUs with FSDP and full parameter/optimizer offloading. The reward weights follow the composite reward in Eq. 5.

C.2 Evidence Tree Generation Pipeline

The evidence tree generation proceeds in the following steps:

1. **Seed topic sampling.** We sample diverse seed topics from both corpora.
2. **Recursive decomposition.** Starting from a seed topic, we use DeepSeek-V3.2 (Liu et al., 2025a) to recursively decompose the topic into sub-queries. At each internal node, the model generates 2–6 child sub-queries that collectively cover distinct parallel facets of the parent (e.g., mechanisms, stakeholders, risks, quantitative evidence, alternatives). The branching factor is depth-aware: 6 children at the root, 4 at intermediate levels, and 3 at deeper levels, up to the configured maximum depth. The decomposition prompt (Figure 9) explicitly requires coverage of comparison/alternatives, risks/failures, and quantitative evidence.
3. **Gating.** At each level, a separate LLM call determines whether further branching would add meaningful evidence. Expansion continues only if the proposed children open substantively different facets.
4. **Evidence retrieval and extraction.** At each node, we retrieve supporting passages from the source corpus using the node's sub-query. An evidence extractor (Figure 10) produces 1–3 concise factual statements per passage, grounding each node in specific retrieved content.
5. **Termination.** Expansion terminates when a sub-query is judged atomic (i.e., further branching is unlikely to add novel, decision-relevant evidence) or when it reaches the configured maximum depth. The resulting trees average 54.48 nodes and 38.66 leaves, providing a broad evidence set for downstream synthesis.
6. **Bottom-up merge and rubric generation.** The complete tree (nodes + leaf evidence) is serialized into a compact JSON representation and fed to DeepSeek-V3.2 with the bottom-up merge prompt (Figure 11). The model (a) selects a small set of leaf nodes covering distinct facets, (b) iteratively merges them bottom-up through intermediate queries into a single final research question, and (c) generates grounded rubric criteria, each with a type (factual/logical), description, importance weight in $[0, 1]$, source leaf IDs, and supporting evidence for factual rubrics.
7. **Verification and revision.** A separate verification pass using GPT-5.1 (Singh et al., 2025a) audits each sample on six dimensions: question–rubric alignment, deep-research suitability, rubric quality (atomic, non-redundant), evidence sufficiency, merge faithfulness, and scoring validity. The auditor returns a KEEP-/REVISE/DROP decision; REVISE samples receive repaired questions and rubrics. This stage conservatively revises 91.50% of generated tuples and drops 7.87%.

We retain 9,064 KEEP or REVISE examples and use all retained pairs for training.

Tree-depth upper bound. We set the maximum node depth to 3 to control construction cost and avoid excessive expansion. The purpose of the tree is to provide enough complementary evidence facets for query–rubric co-generation, not to exhaustively enumerate every possible subtopic. Under this upper bound, the generated trees already contain four evidence levels, 54.48 nodes, 38.66 leaves, and 2.54 evidence passages per leaf on average; 91.40% of terminal leaves reach depth 3. These statistics indicate that coverage is already high for our intended use: each tree provides dozens of candidate evidence leaves, from which the bottom-up merge selects the subset most rel-

Algorithm 1: Evidence-Tree Query–Rubric Co-Generation

Require: Corpus $\mathcal{C}$, retriever ρ, LLM $\mathcal{M}$, maximum depth $D_{\max}$
Ensure: Dataset $\mathcal{D} = \{(x_i, T_i, \mathcal{R}(T_i))\}_{i=1}^{N}$
for each sampled topic $t \sim \mathcal{C}$ **do**
Stage 1: top-down evidence-tree construction
  Initialize root v_0 with $q_{v_0} \leftarrow t$ and $\mathcal{P}_{v_0} \leftarrow \rho(t)$
  QUEUE $\leftarrow \{v_0\}$
  while QUEUE $\neq \emptyset$ **do**
    $v \leftarrow$ QUEUE.pop()
    if $\mathrm{depth}(v) = D_{\max}$ or $\mathcal{M}$ judges q_v as atomic **then**
      continue ▷ v becomes a leaf
    end if
    $\mathcal{Q}_v \leftarrow \mathcal{M}(q_v, \mathcal{P}_v, \mathrm{Anc}(v))$
    for each child query $\tilde{q} \in \mathcal{Q}_v$ **do**
      $\tilde{\mathcal{P}} \leftarrow \rho(\tilde{q})$
      if $\tilde{\mathcal{P}}$ is sufficient and $\tilde{q}$ is non-redundant **then**
        Create child u with $q_u \leftarrow \tilde{q}$ and $\mathcal{P}_u \leftarrow \tilde{\mathcal{P}}$
        Add u to $\mathrm{Ch}(v)$ and push u to QUEUE
      end if
    end for
  end while
Stage 2: bottom-up co-generation and revision
  $(x, \mathcal{R}(T)) \leftarrow \mathcal{M}_{\text{synth}}(T)$
  $(x, \mathcal{R}(T)) \leftarrow \mathcal{M}_{\text{verify}}(x, T, \mathcal{R}(T))$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x, T, \mathcal{R}(T))\}$
end for

evant to a coherent research question and rubric. Increasing the upper bound would also increase retrieval calls, serialized context length, duplicate or weakly related leaves, and topic drift, which can make the final question and rubrics harder to verify. We therefore use depth 3 as a cost-controlled upper bound that provides sufficient multi-facet coverage while keeping synthesis, audit, and revision reliable.

Recursive Decomposition Prompt

```
You are expanding a research search tree.
Parent query: "{parent_query}"
Current depth: {depth}, max depth: {max_depth}.
Current evidence (snippets + extracted statements):
{snippets}
Statements (sample, deduped top 8): {statements}
Goal: maximize coverage AND depth potential.
Propose up to {limit} sibling queries that:
- Each covers a different parallel facet of the parent: stakeholders, timeline, mechanisms/causality, outcomes, risks/failures, alternatives, quantitative data, geography, policy, controversy.
- Ensure at least one query targets comparison/alternatives, one targets risks/failures, and one targets quantitative evidence.
- Keep at the same abstraction level as the parent.
- Avoid overlap; prioritize facets not yet evidenced.
- Keep queries concise (<12 tokens), factual, non-redundant.
Return JSON: {"queries": ["q1", "q2", "q3"]}
```

Figure 9: **Recursive decomposition prompt.** Branching factor is depth-aware: 6 children at the root, 4 at intermediate levels, 3 at deeper levels.

Evidence Extraction Prompt

```
You are an evidence extractor. Given search queries, an action note (goal), and a webpage summary, return 1-3 concise factual statements for each query from the summary that are relevant to the queries and help achieve the action note.
Queries: {queries}
Action note: {note}
Webpage summary: {summary}
Return JSON: {"statements": ["...", "..."]}
```

Figure 10: **Evidence extraction prompt.** Applied at every tree node after retrieval to extract atomic factual statements from passages.

C.3 Data Statistics

Table 6 summarizes the statistics of the generated training data.

Statistic	Value
Generated examples	9,838
Final retained (KEEP + REVISE)	9,064 (92.1%)
Final training pairs	9,064
Tree depth (fixed)	3
Nodes per tree	54.5
Leaves per tree	38.7
Selected leaves per example	5.0
Evidence passages per selected leaf	2.9
Rubric criteria per example	7.0
Factual / Logical	58.4% / 41.6%
Revision rate	91.5%
Drop rate	7.9%
Tree generation model	DeepSeek-V3.2
Quality audit model	GPT-5.1
SFT subset size	200
SFT annotation model	GPT-5.1

Table 6: **Training data statistics.** Revision indicates auditor-guided repair rather than rejection; retained examples include both KEEP and REVISE cases.

C.4 Verification Statistics and Strong-Model Influence

A natural concern is that the 91.5% revision rate might indicate that the final supervision is generated primarily by the strong verifier rather than by the evidence tree. We clarify that the verifier is not used as an open-ended content generator. As shown in Table 7, the dominant revision categories are structural alignment errors—question–rubric mismatch, evidence sufficiency, scope or merge overreach, invalid weights, and rubric granularity—rather than the addition of new factual targets.

The verification prompt (Figure 12) further enforces tree-grounded revision: revised factual rubrics must cite `source_leaf_ids` that are a subset of the already selected tree leaves, and their supporting evidence must be verbatim or tightly faithful paraphrases of selected-leaf statements. Thus, the verifier cannot introduce external evidence or new factual support; it can only repair how tree-derived information needs are expressed as a query and evaluation criteria.

We therefore view the verifier as improving the *quality and alignment* of the reward specification, rather than changing the factual *source* of the supervision. This distinction is also supported by the no-revision ablation: removing the verifier lowers performance, but the unrevised tree-based rubrics still outperform query-first rubric baselines, indicating that the evidence tree itself provides substantial supervision signal.

Primary Revision Reason	Rate
Question–rubric mismatch	56.4%
Evidence sufficiency / grounding	18.2%
Question scope / merge overreach	8.3%
Negative-weight rubric removal	8.2%
Rubric design / granularity	5.3%
Entity / domain / time mismatch	2.5%
Other	1.0%

Table 7: **Primary revision reasons.** All categories are structural alignment corrections; none involves generating new factual content.

D Tool and Inference Details

During both training and evaluation, the agent uses the same three retrieval-tool interfaces, but the backends differ between local RL rollouts and live benchmark inference:

- **`search`**: Issues a keyword query and returns ranked snippets (dense retrieval over the local Wikipedia/OpenScholar corpora during training; Google Search via Serper API during evaluation).
- **`browse`**: Given a URL or document identifier, retrieves and returns the full text of the page (direct corpus lookup during training; Jina Reader during evaluation).
- **`scholar`**: Issues a scientific-literature query and returns paper snippets (dense retrieval over OpenScholar during training; Google Scholar via Serper API during evaluation).

Tool calls are issued via structured `<tool_call>` JSON blocks within the generation. Each tool call includes a `name` field and an `arguments` field (containing the query or URL). Observations are returned in `<tool_response>` blocks and masked during gradient computation to avoid training on retrieval artifacts. We cap the number of tool call turns at 20 during training and 10 during evaluation to balance efficiency and performance.

D.1 System Prompt

We use a unified system prompt for both training and evaluation. The prompt instructs the model to act as a deep research assistant that performs iterative, multi-source investigations. Key elements include: (1) mandatory multi-round search (at least 2–4 rounds of tool calls), (2) structured output with `<think>`, `<tool_call>`, and `<answer>` tags, (3) inline citation requirements using `<cite id="...">...</cite>` grounded only in returned snippets, and (4) a strict workflow alternating reasoning → tool call / answer. The full system

prompt is shown in Figure 13.

E Evaluation Details

E.1 Benchmark Descriptions

We provide additional details on the three evaluation benchmarks:

- **AstaBench-ScholarQA-CS2 (SQAv2)** (Bragg et al., 2025): We use the official evaluation code to evaluate all methods on the 100 test-set questions. We compute rubric score, answer precision, citation precision, and citation recall using Gemini-2.5-Flash as the judge, following Bragg et al. (2025).
- **ResearchQA** (Yifei et al., 2025): We evaluate using the official ResearchQA evaluation suite with GPT-4.1-mini as the judge. We compute averaged rubric scores on a 200-question subset.
- **DeepResearch Bench (DRB)** (Du et al., 2025): We use the official evaluation code on 50 English open-ended deep research questions. We compute Comprehensiveness, Insight/Depth, Instruction-Following, and Readability, and report Overall as the macro average of component metrics. We use Gemini-2.5-Pro as the judge following Du et al. (2025).

E.2 Details of Baselines

For all open deep research baselines, we run their officially released code with default or recommended configurations:

- **Search-R1-7B** (Jin et al., 2025): We use the official checkpoint and inference code.
- **ASearcher-Web-7B** (Gao et al., 2025): We use the official checkpoint and inference code.
- **WebExplorer-8B** (Liu et al., 2025c): We use the official checkpoint. Note that WebExplorer does not release training code or data.
- **WebThinker-32B-DPO** (Li et al., 2026b): We use the official DPO checkpoint. For the fixed-pipeline variant (report mode), we additionally enable the iterative report generation module.
- **Tongyi-DR-30B-A3B** (Team et al., 2025): We use the official checkpoint.
- **DR Tulu-8B** (Shao et al., 2025): We use both the SFT and RL checkpoints with the official `dr-agent-lib` inference pipeline.
- **Ai2 ScholarQA** (Singh et al., 2025b): We use the official code with Claude Sonnet as the backbone LM.

For closed deep research systems (Perplexity Deep Research, Gemini 3 Pro + Search, GPT-5 + Search, and OpenAI Deep Research), we use their public APIs with default settings. Results marked with † in Table 1 are evaluated on a 100-sample subset due to cost constraints. Results marked with * are reported by the original benchmark authors.

F Case Study Response Comparison

Table 8 shows the prompt, abridged final answers, and DRB scores for the case study discussed in §4.6. Ellipses omit intermediate details and citation markup.

Item	Content					
Question	Summarize global AI investments, key initiatives, and outputs by major international consulting firms, including the Big Four, Accenture, MBB, IBM, and Capgemini. The report should cover products/services, client cases, strategy, talent programs, and future trends.					
DR Tulu answer	*"For this summary, 'major international consulting firms' includes the Big Four . . . MBB firms . . . IBM . . . Capgemini . . . Accenture . . . The focus is on their global investments, AI-driven products and services, client-facing initiatives . . . strategic thrusts . . . and talent development programs."* *"Several high-impact items . . . were not retrievable in this session and therefore are not asserted beyond cautious, sourced generalizations. A next-step due-diligence plan would capture . . . annual reports . . . firm newsrooms . . . learning and development pages . . . venture and research lab updates."*					
DEEPRUBRIC answer	*"The available evidence points to a clear pattern: major international consulting firms are investing heavily in AI, but their approaches diverge by firm type and by stage of maturity."* *"In conclusion, the major international consulting firms are pursuing AI through a mix of platform development, client-facing solutions, strategic advisory, and talent upskilling. Accenture and IBM are particularly prominent as platform builders . . . Big Four are scaling AI at massive scale . . . MBB firms are emphasizing strategy, governance, and human-capital transformation . . . For all firms, the central challenge is translating AI ambition into sustained business value without losing sight of the human and organizational dimensions of transformation."*					
		Overall	Comp.	Depth/Insight	Instr. Follow.	Readability
DRB score	DR Tulu-8B	32.6	24.7	33.8	37.5	37.2
	DEEPRUBRIC-8B	47.6	47.1	47.6	54.2	38.5

Table 8: **Abridged final-answer comparison for one DRB case study.** Ellipses mark omitted details.

Bottom-Up Merge & Rubric Generation Prompt

You are an autonomous agent for proposing **deep-research questions** from a search tree.

Input is a compact JSON representation of a search tree with: nodes (id, parent, depth, query, children) and leaves (leaf id, path root→leaf, evidence statements).

Your task is to define a **complete, well-formed research problem** that ultimately requires a long-form, structured answer. You are expected to make the question self-contained and answerable in scope (but you are NOT expected to solve it).

Important: The tree may contain **noise** or even harmful/irrelevant information not related to the root topic. You must identify and avoid such nodes/claims when forming the question and rubrics. If a leaf is misleading but must be addressed, reflect this with **negative-weight rubrics** that require debunking or exclusion.

You must:
1) Select a **small set of leaf nodes** that best cover distinct facets of the root topic.
2) Perform a **bottom-up merge**: combine selected leaf queries into intermediate queries, then merge those into the final research question.
3) Create rubrics **grounded only in statements from the selected leaves**.
4) Provide a **visualizable reasoning trace**: which leaves were selected, why, and how they were merged step-by-step.

Bottom-up merge requirements:
- Start from selected leaves (last-layer nodes). Group them into **multiple** intermediate queries.
- Merge intermediate queries **iteratively** into higher-level queries until one final question.
- The merge can take many steps; include it in merge_trace with explicit inputs, outputs, and short rationale. Each merge step should explain why those inputs were grouped and what new query they form.

Rubric Types (each rubric MUST belong to exactly one):
- **Logical**: structure/organization/reasoning checks for a long-form answer.
- **Factual**: specific facts/claims that must be addressed, each supported by evidence from selected leaves only.

Weighting:
- Importance score between 0 and 1; higher = more critical.
- Negative weight = misleading/harmful content that should be rejected or debunked.
- Weights do not need to sum to 1.

Rubric quality (applies to every rubric):
- Make each rubric precise, atomic, and binary-checkable; avoid vague verbs.
- Anchor factual rubrics to concrete supporting statements from selected leaf evidence; do not invent new facts.
- Cover evidence breadth & attribution, reasoning soundness, completeness, organization/clarity, and handling of ambiguity or counterpoints.
- Keep rubrics concise and avoid duplicate checks.
- Each rubric must be evaluated independently (no dependencies between rubrics).
- Every rubric MUST include source_leaf_ids ⊆ selected_leaf_ids.

Question Construction:
- Motivated by selected rubrics and evidence; expose meaningful gaps or unresolved relationships.
- Must NOT be fully answerable from provided material alone.
- Do NOT list rubrics or solution steps in the question.
- Brief, precise, research-heavy. Do NOT invent facts beyond the material.

- Minimum 5 rubrics (numbered R1, R2, R3, ...).

Return JSON only: {question, selected_leaf_ids, selection_trace, merge_trace, rubrics}.

Figure 11: **Bottom-up merge and rubric generation prompt.** This prompt is the core of the evidence-tree pipeline, converting raw tree evidence into a research question paired with grounded rubric criteria. The `source_leaf_ids` ⊆ `selected_leaf_ids` constraint ensures all rubrics remain grounded in retrieved evidence.

Quality Verification System Prompt

You are an expert auditor for tree-rubric data used in long-form answer evaluation and deep research QA. You must audit one JSON sample.

Goal: Decide if the sample should be KEEP / REVISE / DROP. If REVISE, provide a repaired question and/or repaired rubrics. Keep the original topic unless structurally unsalvageable.

Deep research standards: Requires synthesis across multiple evidence sources; multi-step reasoning, not shallow fact lookup; connects mechanisms, evidence, limitations, uncertainty, and conclusions; rewards structured, evidence-grounded comparative reasoning.

Audit dimensions (apply all, do not skip):

1) **Question-rubric alignment**: Rubrics must stay centered on the main question. Granularity should match. Do not elevate a narrow technical detail into a mandatory rubric unless truly central.

2) **Deep research suitability**: Rubrics should evaluate deep reasoning—comparison, synthesis, causal/mechanistic explanation, evidence qualification, structured conclusions—not reduce to a checklist of isolated facts.

3) **Rubric quality**: Each rubric atomic, specific, independently judgeable, non-redundant. Avoid vague or non-judgeable wording.

4) **Evidence sufficiency**: source_leaf_ids must sufficiently support each rubric. A detail in one leaf does not automatically justify a required rubric. Flag claims stronger than cited support. Note: leaf nodes may be truncated; judge based on visible information.

5) **Merge faithfulness**: Final question should reflect semantic union of selected leaves and intermediate merges. Flag unjustified broadening, strengthening, or narrowing.

6) **Scoring validity**: Weights reasonable; core factual rubrics not dominated by vague logical ones; negative-weight rubrics treated as problematic unless explicitly justified.

Decision policy: KEEP (usable as-is); REVISE (valuable but needs targeted repair); DROP (severe structural or semantic failure). Prefer REVISE over DROP when topic remains valuable. Do not invent unsupported claims.

If REVISE, follow hard rewrite template:

- revised_question: Keep topic, fix overreach/under-specification. Must stay faithful to selected leaves + merge trace. Must remain a deep research question.
- revised_rubrics: Output 5-8 rubric objects. Every rubric MUST contain: id, type (logical/factual), description, weight, source_leaf_ids. source_leaf_ids must be $\subseteq$ selected_leaf_ids. Logical rubrics do NOT need evidence. Factual rubrics MUST include non-empty evidence grounded in selected-leaf statements (verbatim or tightly faithful short paraphrase). Avoid semantic duplicates. Prefer rubrics evaluating synthesis, evidence-based comparison, causal reasoning, limitations, and conclusion quality.

Return valid JSON only: {decision, error_tags, major_issues, justification, fix_plan, revised_question, revised_rubrics, implementation_guidance}.

Figure 12: **Quality verification system prompt.** The key constraint is that all revised rubrics must ground source_leaf_ids within the tree's selected_leaf_ids and provide evidence from leaf statements only, preventing the verifier from injecting parametric knowledge.

Deep Research Agent System Prompt (Part 1: Task & Answer Requirements)

```
You are a deep research assistant.  Your core function is to conduct thorough, multi-source
  investigations into any topic.  You must handle both broad, open-domain inquiries and queries
  within specialized academic fields.

  For every request, you MUST perform iterative, multi-step information gathering.  You should
  search as broadly and deeply as possible, gather as much relevant information as is reasonably
  available, and synthesize evidence from credible, diverse sources to deliver a comprehensive,
  accurate, and objective response.

  You MUST NOT rely on a single search. Instead:
  - Always perform multiple rounds of search using different queries, perspectives, or keyword
  variations.
  - After each search, analyze gaps, uncertainties, or missing aspects, and issue follow-up searches.
  - Continue searching until the key aspects of the question are sufficiently covered.
  - You should typically perform at least 2-4 rounds of tool calls before producing the final
  answer, unless the question is extremely simple.

  A response is NOT sufficient if: only one source or one perspective is used; key concepts in the
  question are not individually investigated; there is no cross-source verification.

  The final answer MUST be comprehensive, detailed, and in-depth.  It should fully address all
  aspects of the question, explain underlying mechanisms, compare different perspectives, and
  provide clear reasoning supported by evidence.

  When you have gathered sufficient information and are ready to provide the definitive response,
  you must enclose the entire final answer within <answer></answer> tags.

  You should ground every nontrivial claim in retrieved snippets.  Cite using <cite
  id="...">...</cite> drawn only from returned snippets.  Prefer authoritative sources
  (peer-reviewed papers, reputable benchmarks/docs) and prioritize recent work for fast-moving
  areas. Acknowledge uncertainty and conflicts; if evidence is thin or sources disagree, state it
  and explain what additional evidence would resolve it.

  Structure the answer with clear markdown headers and a coherent flow. In each section, write 5-8
  sentence paragraphs with clear topic sentences and transitions.

  The answer MUST: be comprehensive and cover all key aspects; provide detailed explanations rather
  than brief summaries; compare different models, assumptions, or perspectives when relevant;
  explain causal mechanisms and not just describe phenomena; synthesize information across sources
  into a coherent narrative.

  Use lists sparingly only when they improve clarity.  Synthesize rather than enumerate content:
  group findings across papers, explain relationships, and build a coherent narrative supported by
  citations.

  DO NOT invent snippets or citations and never fabricate content.

  You can reason and use tools iteratively, but all reasoning text must appear only inside a
  standalone <think>...</think> block, and all tool calls must appear only inside a standalone
  <tool_call>...</tool_call> block. Tool calls must never appear inside a <think> block.
```

Figure 13: **System prompt for the deep research agent (Part 1).** Task definition, search requirements, citation policy, and answer formatting rules.

Deep Research Agent System Prompt (Part 2: Tools & Workflow)

Tool Usage Constraints

- The browse tool can ONLY be used on URLs returned by a previous search call.
- You MUST NOT fabricate, guess, or manually construct URLs for browsing.
- If additional webpages are needed, first use search to retrieve them, then select from returned results.
- Prioritize browsing multiple distinct URLs from different sources to ensure diversity of evidence.

Available Tools

{"name":"search","description":"Perform web searches and return top results.","parameters":{"query":[list of strings]}}

{"name":"browse","description":"Open a specific URL and return readable content.","parameters":{"url":"string"}}

{"name":"scholar","description":"Retrieve information from scientific papers.","parameters":{"query":[list of strings]}}

Tool Call Format

<tool_call>{"name":"tool_name","arguments":{...}}</tool_call>

Tool Output Format

For search/scholar: <tool_response><snippet id=ID>content</snippet>...</tool_response>
For browse: <tool_response><webpage id=ID>content</webpage></tool_response>

Support every non-trivial claim with retrieved evidence. Wrap the exact claim span in <cite id="ID1,ID2">...</cite>, where IDs are snippet IDs from returned results (comma-separated if multiple). Use only returned snippets; never invent IDs.

Workflow Rules

After every </tool_response>, the assistant must first output a standalone <think>...</think> block. Then it must output either: (1) exactly one standalone <tool_call> block if more evidence is needed, or (2) exactly one standalone <answer></answer> block if evidence is sufficient.

Final Answer Rules

- Generate a comprehensive, detailed, and in-depth final answer marked with <answer></answer>.
- Avoid short or shallow responses; prioritize depth, clarity, and completeness.
- Wrap supported text in <cite id="SNIPPET_ID">...</cite> using exact IDs from returned results.
- If multiple sources support a passage, use multiple <cite> tags around relevant clauses.

Figure 14: **System prompt for the deep research agent (Part 2).** Tool definitions, call/output format, workflow rules, and final answer requirements. The same prompt is used for both training and evaluation; during training the tools connect to local corpora, during evaluation they connect to live APIs.