REINFORCEMENT LEARNING FOUNDATIONS FOR DEEP RESEARCH SYSTEMS: A SURVEY

Wenjun Li, Zhi Chen, Jingru Lin, Hannan Cao, Wei Han, Sheng Liang, Zhi Zhang, Kuicai Dong, Dexun Li, Chen Zhang, Yong Liu
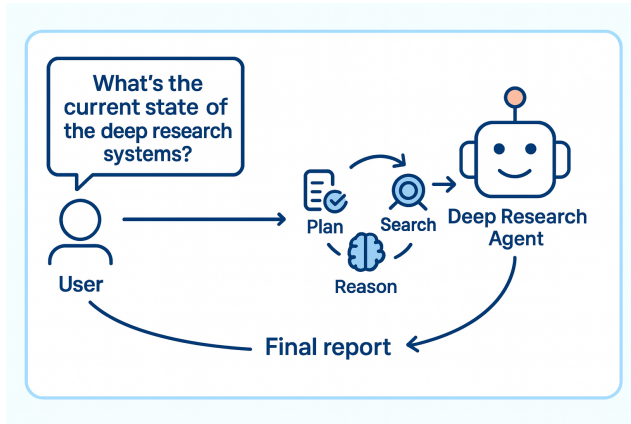
Huawei Technologies Co., Ltd

ABSTRACT

Deep research systems, agentic AI that solve complex, multi-step tasks by coordinating reasoning, search across the open web and user files, and tool use, are moving toward hierarchical deployments with a Planner, Coordinator, and Executors. In practice, training entire stacks end-to-end remains impractical, so most work trains a single planner connected to core tools such as search, browsing, and code. While SFT imparts protocol fidelity, it suffers from imitation and exposure biases and underuses environment feedback. Preference alignment methods such as DPO are schema and proxy-dependent, off-policy, and weak for long-horizon credit assignment and multi-objective trade-offs. A further limitation of SFT and DPO is their reliance on human defined decision points and subskills through schema design and labeled comparisons. Reinforcement learning aligns with closed-loop, tool-interaction research by optimizing trajectory-level policies, enabling exploration, recovery behaviors, and principled credit assignment, and it reduces dependence on such human priors and rater biases.

This survey is, to our knowledge, the first dedicated to the RL foundations of deep research systems. It systematizes recent work along three axes: (i) data synthesis and curation; (ii) RL methods for agentic research covering stability, sample efficiency, long context handling, reward and credit design, multi-objective optimization, and multimodal integration; and (iii) agentic RL training systems and frameworks. We also cover agent architecture and coordination, as well as evaluation and benchmarks, including recent QA, VQA, long-form synthesis, and domain-grounded, tool-interaction tasks. We distill recurring patterns, surface infrastructure bottlenecks, and offer practical guidance for training robust, transparent deep research agents with RL.

A curated paper list is available at github.com/wenjunli-0/deepresearch-survey.

CONTENTS

1 INTRODUCTION

The rapid emergence of deep research systems (e.g., OpenAI (2025); Google (2025); Perplexity Team (2025); Team et al. (2025)), large language models (LLMs) capable of tackling complex, multi-step information-seeking tasks, marks a significant shift in how AI approaches reasoning, execution, and synthesis. In this survey, we focus on information-seeking use cases, as most existing research and products center on this application. We define deep research systems as AI researchers that autonomously plan and carry out multi-step investigations across the open web and user-provided files, iteratively searching, reading, and reasoning as new evidence appears, and ultimately producing either a concise answer for an objective question or a well-structured, citation-backed report for a subjective open question.

A trend in both academia (Li et al., 2025f; Jin et al., 2025b; Wan et al., 2025) and industry (ByteDance & contributors, 2025; LangChain & contributors, 2025; MiroMindAI & contributors, 2025) is to move from monolithic agents to hierarchical agent architectures for deep research. Figure 1 mirrors this architecture: a Planner performs step-by-step decomposition and reflection; a Coordinator handles assignment, delegation, aggregation, and verification; and a pool of Executors (i.e., specialized agents and tools) executes grounded actions over the web and files. This separation of concerns decouples strategic planning from execution details, enabling parallelization, plug-and-play expertise (e.g., swapping in better searchers or code runners and scaling to additional tools), and tighter instrumentation for process logging, credit assignment, and auditability. It also keeps the Planner's global state clean and coherent over long horizons while the Coordinator and Executors handle delegation and grounded actions.

While the hierarchical architecture is attractive for deployment, it is not yet practical to train the entire workflow end-to-end. As a result, most research targets a single model (typically the Planner) wired directly to a small set of fundamental tools (search/browse/code), which collapses rollout length and variance, fits existing RL/SFT/DPO infrastructure, and yields cleaner signals. The training objective is to strengthen long-horizon capabilities in one place (i.e., reasoning, decomposition, tool use, reflection, and synthesis) in an end-to-end manner so that the resulting Planner can later slot into the full hierarchy as a stronger "brain," while coordination and execution remain modular and swappable. Hence, in this survey, we primarily focus on the training of the planner model and will cover the agent architecture and coordination design in Section 5.

Supervised fine-tuning (SFT; Ouyang et al. (2022); Wei et al. (2022)) is an effective way to initialize deep research agents: it is stable, data-efficient, and good at teaching protocol fidelity (e.g., tool-call schemas, response formats), and basic stepwise reasoning patterns. Because SFT optimizes against gold (x, y) pairs, it excels at imparting local behaviors like query rewriting templates, citation style, argument wrapping, and reduces variance early on. The same properties, however, limit performance on multi-turn research tasks. Reference traces are long, composite, and human-authored; imitation induces imitation bias (copying a particular decomposition) and exposure bias (teacher-forced steps hide compounding errors at inference). SFT also leaves environment feedback underused: it cannot directly learn from tool failures, stochastic retrieval, or non-stationary states (e.g., prices, availability). In short, SFT is valuable scaffolding for competencies and interfaces, but not a vehicle for optimizing end-to-end decision quality.

Preference-based methods (e.g., DPO; Rafailov et al. (2023)) can be pushed beyond single-turn outputs by decomposing agent workflows into labeled steps (e.g., query generation, retrieval selection, synthesis) and learning local preferences at each stage. However, although there are several research works exploring training deep research agents via DPO-based methodologies (Zhang et al., 2025c; Zhao et al., 2025a; Asai et al., 2023), we think several structural mismatches remain in such approahces. First, DPO optimizes textual alternatives rather than state–action returns: pairwise losses are applied to strings conditioned on prior text, without explicit grounding in environment state (tool results, cache, budgets) or action semantics. This makes credit assignment inherently myopic—it judges which snippet is preferable at that step but cannot attribute downstream success/failure to earlier retrieval or tool-use decisions, nor can it trade off depth of search against cost/latency under partial observability. Second, stepwise DPO inherits schema and proxy dependence: one must hand-design the process decomposition and generate preferences (often with heuristics), which introduces label noise, and brittleness when the unseen task requires a different decomposition. Third, DPO is largely off-policy and offline: it improves on fixed comparisons but does not explore the closed-
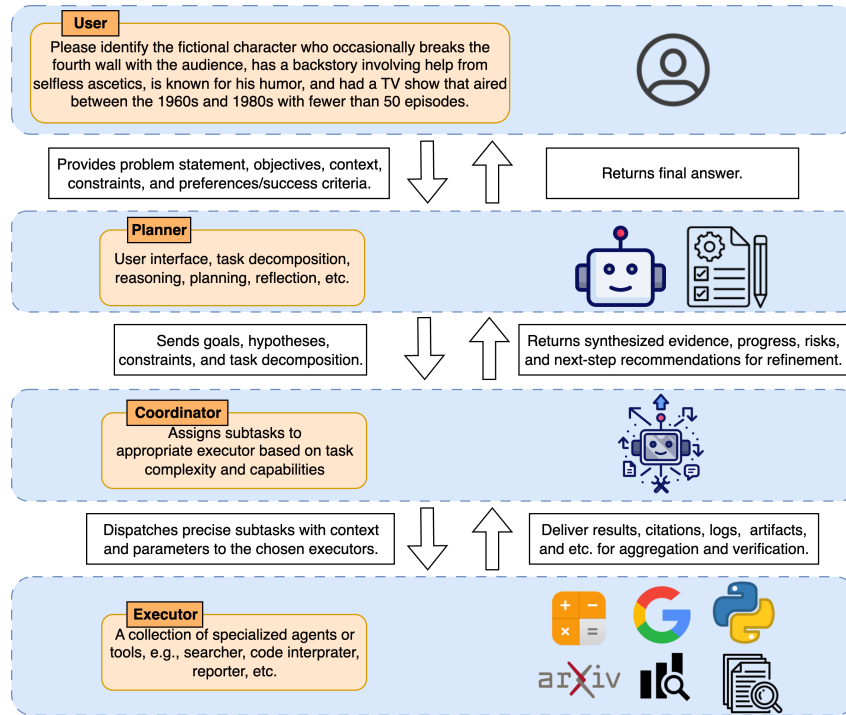
Figure 1: Illustration of the hierarchical deep research system architecture.

loop space of actions and tool outcomes, so it struggles to learn recovery behaviors (e.g., when a query returns junk, a site blocks access, or prices shift) and to adapt to non-stationary environments. Finally, multi-objective desiderata (accuracy, calibration, cost, safety) enter only implicitly through rater preferences; DPO provides no principled mechanism to aggregate rewards over long horizons.

Given the limitations of SFT/DPO-based approaches, both the industry and academia regard reinforcement learning (RL) as a promising pathway towards training deep research agents in an end-to-end manner. Deep research ultimately demands trajectory-level learning in a closed-loop, tool-rich environment: deciding how to decompose problems, when and how to invoke tools, which evidence to trust, when to stop, and how to trade off accuracy, cost, and latency as the state evolves. RL treats the system as a policy over states and actions, enabling end-to-end improvement from environment signals, credit assignment across multi-step traces, and exploration of alternative strategies for search, tool orchestration, recovery, and synthesis.

Motivated by the shift toward RL for training deep research agents, and the rapid acceleration of progress in this area, we present, to our knowledge, the first survey dedicated to the RL foundations of deep research systems. This survey aims to help researchers systematically understand recent developments, underlying challenges, and key methodological advances. We organize the literature along three primary axes:

- **Data Synthesis & Curation:** Methods for creating and curating complex, high-quality training data, often through synthetic generation, to support multi-step reasoning and tool usage.

- n **RL Methods for Agentic Research:** Works that (i) extend baseline pipelines (e.g., DeepSeek-R1-style Guo et al. (2025)) to improve stability, sample efficiency, long-context handling; (ii) design rewards and credit assignment that propagate credit across multi-step traces (outcome- vs. step-level, composite judges, return decomposition); and (iii) integrate multimodality via backbone VLMs that run iterative perception–reasoning cycles.

- **Agentic RL Training Frameworks**: Training deep research agents that interact with tools over long horizons requires a scalable, stable training system. We survey recent open source

infrastructures to surface bottlenecks, distill recurring design patterns, and offer practical guidance for composing scalable and reproducible training stacks.

Beyond RL training foundations, we highlight two cross-cutting areas that are strategically important:

- **Agent Architecture & Coordination:** Hierarchical, modular, and multi-agent designs that enhance compositional reasoning and division of labor.
- **Evaluations & Benchmarks:** Datasets for assessing deep research systems in holistic, task-rich, tool-interactive settings.

Together, these axes provide a cohesive view of the RL-enhanced deep research ecosystem. By tracing advances along each axis, the survey offers a conceptual roadmap for newcomers and a technical reference for researchers aiming to push agentic AI toward robust, real-world problem solving. Figure 2 presents the taxonomy and the key papers we survey.
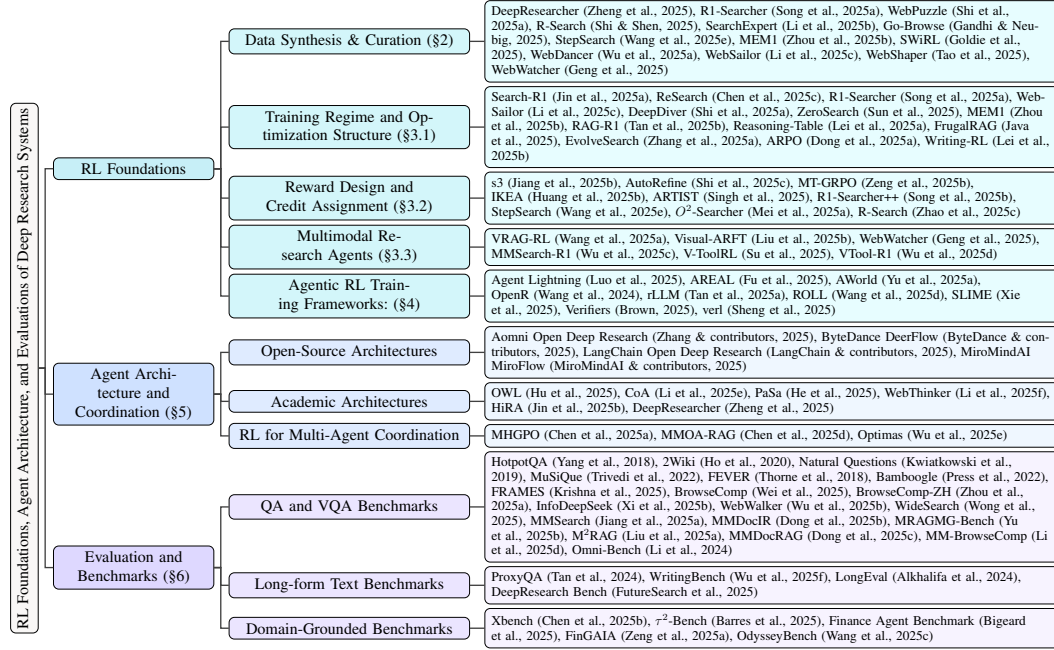

Figure 2: The organizational structure of the survey and representative papers under each branch.

Positioning and contributions   Unlike concurrent surveys (Huang et al., 2025a; Li et al., 2025a; Xi et al., 2025a; Xu & Peng, 2025; Li et al., 2025g; Zhang et al., 2025d;b), we adopt a training-first, RL-centric perspective. We (i) explain why SFT/DPO misalign with closed-loop, tool-interactive research and motivate end-to-end RL at the planner; (ii) present, to our knowledge, the first taxonomy dedicated to RL foundations for deep research, spanning data synthesis & curation, RL methods for agentic research, and training frameworks; (iii) treat agentic RL training as a systems problem, surfacing infrastructure bottlenecks and scalable training; (iv) connect training to deployment via a planner-centric training vs. hierarchical execution decoupling, and provide a comprehensive, in-depth synthesis of evaluations and benchmarks. Compared with contemporaneous overviews, our survey is more narrowly focused on RL and delivers deeper data, algorithmic, and infrastructure insights. Together, our survey offers a unified blueprint and actionable guidance for training robust deep research agents via RL.

Timeframe and inclusion criteria   We survey RL-based training for deep research agents released after February 2025 (post–DeepSeek R1) through September 2025 (manuscript cut-off), covering RL training pillars and agent architecture and coordination designs. For benchmarks papers, we cite canonical QA/VQA and long-form text benchmarks developed in recent years, while domain-grounded, tool-interactive benchmarks are restricted to 2025 vintages.

2 DATA SYNTHESIS & CURATION

Data synthesis has become an increasingly critical component of data-driven training paradigms, particularly in the development of modern AI systems. With the rise of generative models, synthetic data generation has become more accessible and cost-effective. However, curating high-quality synthetic data for training deep research agents remains a challenging task, especially when aiming to support complex reasoning, tool use, and multi-step decision-making.

In this section, we first discuss how RL training data differs from that of SFT/DPO training, and examine the essential properties that synthetic data should exhibit to effectively support the training of deep research agents. We then provide an overview of recent works that explore various techniques for constructing complex queries and curating training data. We introduce a taxonomy of data complexity to better characterize the difficulty and structure of synthetic tasks. Finally, we outline the insights and open challenges in this rapidly evolving area.

Table 1: Summary of papers in this section. New Dataset(s) indicates whether a named dataset was newly released (as opposed to only curating or augmenting existing data).

Paper	New Dataset(s)
Construction methods with new datasets	
DeepDiver (Shi et al., 2025a)	Yes — *WebPuzzle*.
WebDancer (Wu et al., 2025a)	Yes — *CrawlQA, E2HQA*.
WebSailor (Li et al., 2025c)	Yes — *SailorFog-QA*.
WebShaper (Tao et al., 2025)	Yes — *WebShaper*.
WebWatcher (Geng et al., 2025)	Yes — *BrowseComp-VL*.
InfoSeek (Xia et al., 2025)	Yes — *InfoSeek*.
Systems / pipelines (no new dataset)	
R-Search (Shi & Shen, 2025)	No — uses fresh corpora.
SearchExpert (Li et al., 2025b)	No — constructed tasks.
SWiRL (Goldie et al., 2025)	No — rollout prefixes.
Go-Browse (Gandhi & Neubig, 2025)	No — curated existing datasets.
StepSearch (Wang et al., 2025e)	No — augments *MuSiQue*.
Search-R1 (Jin et al., 2025a)	No — uses *NQ + HotpotQA*.
R1-Searcher (Song et al., 2025a)	No — difficulty labels on existing data.
MEM1 (Zhou et al., 2025b)	No — synthesized from existing data.
DeepResearcher (Zheng et al., 2025)	No — curated existing datasets.

2.1 FORMAT OF RL TRAINING DATA

We begin by clarifying the purpose, format, and requirements of RL training data for deep research agents. This background frames the discussion that follows and clarifies the terminology we use throughout.

RL optimizes *goal achievement in an environment*. Formally, the deep research agent learns a policy π_θ that maximizes expected return under tool-use interactions. The RL training setup comprises *environment, tasks, and reward*. Let the task set be

$$\mathcal{Q} = \{q_j\}_{j=1}^M,$$

often multi-step questions/tasks. The interactive environment is

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, P, r),$$

where $\mathcal{S}$ are states, $\mathcal{A}$ are actions (e.g., tool calls like search, browse, code interpreter), $\mathcal{O}$ are observations (pages/snippets/images), P defines state-transition dynamics, and r yields rewards. A trajectory

$$\tau = (o_0, a_0, o_1, a_1, \ldots, o_T, a_T)$$

consists of observations and actions during task execution. Outcome-only rewards take the form

$$R(q, \tau) = g\big(\hat{y}(\tau), y_q^\star\big) \quad (g(\cdot) \text{ can be e.g., EM/F1}),$$

4

and can be augmented with stepwise/process signals $r_t(q, \tau)$ (e.g., feasibility checks). Crucially, RL does *not* require expert trajectories; $(q, y_q^\star)$ plus a reliable $g(\cdot)$ suffice.

Most RL pipelines for deep research adopt an DeepSeek-R1-style regime and use a small "SFT" set to cold-start the policy (see Section 3). Unlike pure SFT, this data's role is interface compliance and scaffolded rollouts, not end-to-end imitation. Each example is a full trajectory that (i) plans and decomposes the query (`<think>...</think>`); (ii) invokes tools with validated argument schemas (`<tool_name>...</tool_name>`); (iii) parses and interprets intermediate outputs (`<result>...</result>`); and (iv) synthesizes the final response (`<answer>...</answer>`). The emphasis is on correct calling conventions, step ordering, and smooth transitions between reasoning, tool interaction, and answer synthesis; subsequent training then replaces imitation with end-to-end training from the initial query, through tool-mediated interactions, to final rewards.

RL training data for deep research agents requires: (i) tasks that resist parametric recall, e.g., a cross-document/recency constraint $\chi(q) \geq 2$ and a contamination predicate $C(q) = 0$ (not answerable from parametric memory alone); (ii) *cheap, stable, objective* rewards R (e.g., exact answers, checklists, functional tests). For fair comparison, RL pipelines also fix retrieval settings and enforce step/time caps on rollouts.

RL performance hinges on *which tasks* are posed and *how* success is verified and selected. We therefore separate:
$$\mathrm{Construct}(\mathcal{C}) \ \to \ \tilde{\mathcal{Q}},$$
which maps corpora/web sources $\mathcal{C}$ into *candidate tasks* that demand multi-step reasoning and tool use; and
$$\mathrm{Curate}(\tilde{\mathcal{Q}}) = \{ q \in \tilde{\mathcal{Q}} : \bigwedge_{f \in \mathcal{F}} f(q) = 1 \},$$
a filtering/scheduling pipeline with optional curriculum $\mu(q)$ implementing contamination/novelty gates, outcome/process verification, and difficulty staging.

2.2 Constructing Complex Queries and Curating Data

We analyze data preparation for deep research systems along two complementary axes—how complex queries are constructed and how the resulting data are curated. We first cover construction strategies (what goes into $\tilde{\mathcal{Q}}$), then turn to curation (what remains and how it is scheduled).

Construction Strategies To elicit and cultivate long-horizon capabilities—robust reasoning, iterative tool interaction, reflection, and synthesis—we require genuinely complex queries that compel the model to perform multiple rounds of planning, evidence gathering, and verification. To avoid "shortcut" solutions (e.g., answers obtainable via a single lookup or memorized fact), a line of work focuses on strategies that deliberately increase task difficulty while preserving verifiability. We group these strategies into three categories, outlined below:

1. **Cross-Source Composition (often recency-aware).** These methods author questions that require integrating evidence across multiple sources—typically drawn from recent news/papers/webpages to push beyond the model's parametric memory. *R-Search* clusters fresh documents (news + arXiv) and trains a single LLM to plan, execute multi-source search, and synthesize an answer in one pass; the learning objective explicitly couples reasoning with structured, stepwise search. *WebPuzzle* generates cross-page inverted QA from multiple open-web pages and further assigns pass@k difficulty tags to winnow easy items, producing a curated subset for RL. *SearchExpert* also starts from fresh crawls but is plan-centric: it produces code-level search DAGs and converts them into compact natural-language DAGs for SFT; harder cases are used for RL with a search-feedback reward tied to retrieved evidence quality.

2. **Structure-Driven Path Growth (graph/set navigation).** Here, solution length grows by expanding over hyperlink graphs or by composing formal set operations. *CrawlQA* grows paths by recursive link traversal from authoritative roots (arXiv/GitHub/Wikipedia) to emulate human browsing, then synthesizes typed questions (e.g., COUNT, MULTI-HOP, INTERSECTION) from the visited page set—lengthening solution paths via browsing rather

than obfuscation. *WebSailor* builds dense site graphs via random walks from rare entities, then samples subgraphs to create SailorFog-QA—questions that demand multi-hop browsing and non-linear synthesis. *WebWatcher* constructs a Wikipedia hyperlink graph and turns selected items into image-grounded VQA to require cross-modal retrieval. *WebShaper* replaces literal graph traversal with a set-theoretic formalism—Knowledge Projections—and a layer-wise Expander that controls reasoning depth while avoiding shortcut paths. *Go-Browse* treats data collection as structured exploration over a reusable URL graph (NavExplorer, PageExplorer), keeping only tasks that pass a feasibility check judged by a VLM.

3. **Difficulty Staging by Transformation/Evolution.** These create easy-to-hard progressions via rewriting or step-level supervision. *E2HQA* iteratively rewrites a simple seed by replacing entities with constraints mined from the web while preserving the final answer, so each iteration adds hops. *StepSearch* augments MuSiQue with sub-question trajectories and trains with information-gain rewards and redundancy penalties per step, encouraging deeper, better-targeted queries. *MEM1* increases workload by bundling multiple independent questions into a single multi-objective prompt (harder via breadth rather than tighter cross-step dependency). *SWiRL* rolls out multi-step tool-use trajectories with an open-source model, then splits each k-step trace into k prefixes to form an implicit prefix curriculum.

Beyond construction and curation, some work introduces a cross-cutting modifier: **obfuscation**. *DeepDiver*, *WebSailor*, and *WebWatcher* deliberately mask entities or fuzz attributes (e.g., vague dates, indirect descriptors) so direct lookup fails, forcing multi-step discovery. This is an orthogonal knob to raise difficulty and can be combined with any of the construction and curation choices above.

Curation Strategies   Orthogonal to constructing new complex queries, other work designs selection procedures that decide what stays for training. We survey how examples are filtered/selected as follows:

1. **Contamination and Novelty Gates.** These filters remove samples solvable from parametric memory or by a single document so that training focuses on truly complex tasks. *DeepResearcher* drops any question that the base model answers within pass@10, ensuring novelty beyond internal knowledge. *SearchExpert* retains only items that require its crawled context, explicitly rejecting cases answerable without retrieval from the constructed corpus.

2. **Outcome-Verified Selection.** This keeps items whose rollouts are either demonstrably solvable or demonstrably nontrivial under a verifiable success test. *Search-R1* optimizes and reports final answer EM on a unified NQ+HotpotQA pool, using the same objective to filter useful rollouts for continued training. *WebPuzzle* tags each item with pass@k difficulty and filters out easy cases to yield a harder RL subset. *Go-Browse* requires at least one verified success judged by a VLM before an item is admitted, which both proves feasibility and guards against degenerate tasks that waste budget.

3. **Process-Quality Filtering.** Rather than judging only final answers, these methods validate intermediate steps so credit aligns with sound action and reasoning. *SWiRL* introduces an LLM-as-judge process filter to check coherence and rationale alignment across steps. *StepSearch* augments MuSiQue with sub question trajectories and uses information gain rewards and redundancy penalties at each step, effectively filtering for processes that add evidence rather than repeat it. *Go-Browse* also applies a feasibility judge during exploration, discarding paths that fail stepwise checks even if a final answer could be guessed.

4. **Difficulty-Aware Curricula and Scheduling.** After filtering, examples are staged so training time concentrates on the band that moves learning the most. *R1-Searcher* labels items by observed rollout counts on Hotpot and 2Wiki (easy, medium, hard) and upweights medium and hard during training. *WebPuzzle* samples a hard mix after tagging to keep batches informative.

2.3   CLASSIFICATION OF QUERY COMPLEXITY

After surveying recent advancements in constructing and curating RL training data for deep research agents, we propose the following classification of QA tasks based on their complexity. This taxon-
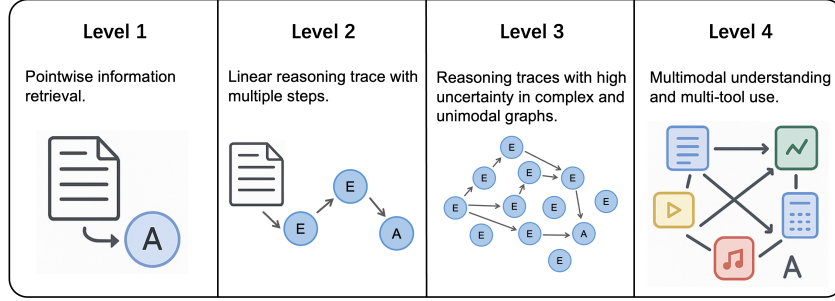
Figure 3: Illustration of QA Task Complexity Levels.

omy provides a common language to (i) stratify datasets for curriculum design, (ii) report results by difficulty band, and (iii) diagnose failure modes in long-horizon training:

Level 1: Questions with low uncertainty that can be answered directly using the model's internal knowledge or a single, straightforward web search. Examples include factual queries about natural phenomena or simple weather-checking questions. Example datasets: SimpleQA (Wei et al., 2024), TriviaQA (Joshi et al., 2017).

Level 2: Multi-hop questions that require multiple searches but follow a clear and well-defined reasoning path. These questions can be solved through a structured sequence of logical steps. For instance, questions generated via easy-to-hard reverse construction techniques fall into this category. Example datasets: HotpotQA (Yang et al., 2018), Bamboogle (Press et al., 2022).

Level 3: Questions characterized by both high uncertainty and high difficulty in reducing that uncertainty. The involved entities are connected in complex and often emergent ways, with no pre-defined reasoning path. Solving these tasks requires extensive exploration and rigorous cross-validation. Importantly, these questions remain *unimodal*, meaning that the entire reasoning chain and all necessary clues can be resolved through textual search alone. Example datasets: SailorFog-QA, WebShaper.

Level 4: Questions that require both *multimodal* understanding and multi-tool orchestration, extending Level 3 in complexity. The input question itself may be presented across different modalities (e.g., text, images, audio), requiring the agent to first interpret the query correctly. Solving the task demands coordinating multiple tools (e.g., image recognition, audio transcription, code execution) and integrating evidence from diverse modalities to produce a final answer. Example datasets: WebWatcher, FactualVQA (Wu et al., 2025c).

2.4 DISCUSSION

We suggest using the levels as a practical curriculum: warm-start on Levels 1–2 to establish interface compliance and basic planning, advance to Level 3 for exploration and cross-validation skills, and add Level 4 last for multimodal coordination. Report learning curves and sample efficiency by level to make these stages visible. Treat "construction" and "curation" as complementary levers observed in the literature: construction increases structural difficulty (hops, branching, recency, modality), while curation improves signal quality (decontamination, process checks, difficulty mixing).

We present the following questions for future research: (i) How can we generate large-scale, complex, multimodal queries at low cost while ensuring feasibility and verifiability? (ii) Active, difficulty-aware task generation during training. Can we close the loop so the agent (and its value/uncertainty estimates) drives on-the-fly construction and curation—selecting recency windows, obfuscation knobs, and curricula to maximize sample efficiency?

3 RL METHODS FOR AGENTIC RESEARCH

With high-quality synthetic data in place, the next step is designing effective RL training pipelines for deep research agents. As established in Section 1, this chapter focuses on recent advances in

Table 2: Summary of papers in Section 3.1.

Work	Policy Model	Cold Start?	Reward Type	RL Optimizer
Search-R1 (Jin et al., 2025a)	Qwen2.5-7B-Base/Instruct	✗	Outcome	PPO, GRPO
ReSearch (Chen et al., 2025c)	Qwen2.5-32B-Base/Instruct	✗	Outcome+Format	GRPO
R1-Searcher (Song et al., 2025a)	Qwen2.5-7B-Base, Llama-3.1-8B-Instruct	✗	Outcome+Format	Reinforce++, GRPO
WebSailor (Li et al., 2025c)	Qwen2.5-72B-Base	✓	Outcome+Format	DUPO (GRPO-variant)
DeepDiver (Shi et al., 2025a)	Qwen2.5-7B-Instruct, Pangu-7B-Reasoner	✓	Outcome+Format +Retrieval_Necessity	GRPO
ZeroSearch (Sun et al., 2025)	Qwen2.5-7B-Base/Instruct, LLama-3.2-3B-Base/Instruct	✗	Outcome	REINFORCE, GRPO, PPO
ASearcher (Gao et al., 2025)	Qwen2.5-14B-Base, QwQ-32B	✗	Outcome+Format	GRPO
SSRL (Fan et al., 2025)	Llama-3.1-8B-Base/Instruct, Qwen2.5-7B-Instruct	✗	Outcome+Format	GRPO
MEM1 (Zhou et al., 2025b)	Qwen2.5-7B-Base	✗	Outcome	PPO
RAG-R1 (Tan et al., 2025b)	Qwen2.5-7B-Instruct	✓	Outcome	PPO
Reasoning-Table (Lei et al., 2025a)	Qwen2.5-7B-Instruct, Qwen2.5-Coder	✓	Outcome+Format +Position	GRPO
FrugalRAG (Java et al., 2025)	Qwen2.5-7B-Instruct, Llama-3.1-8B-Instruct	✓	Outcome +Retrieval_Necessity	GRPO
EvolveSearch (Zhang et al., 2025a)	Qwen2.5-7B-Instruct	✓	Outcome+Format	GRPO
ARPO (Dong et al., 2025a)	Qwen3-14B-Base, Llama-3.1-8B-Instruct	✓	Outcome	ARPO (GRPO-variant)
Writing-RL (Lei et al., 2025b)	Qwen2.5-7B-finetuned, Llama-3.1-8B-finetuned	✓	Outcome	PPO

RL-based training (rather than SFT/DPO), emphasizing end-to-end learning that strengthens long-horizon reasoning, planning, tool use, reflection, and synthesis.

We organize the literature into three threads: (i) Training Regime and Optimization Structure: works that go beyond the baseline pipeline (i.e., DeepSeek-R1-style Guo et al. (2025) and Search-R1-style Jin et al. (2025a)) to improve stability, sample efficiency, long-context handling, etc; (ii) Reward Design and Credit Assignment: approaches that determine what behaviors are reinforced and how credit is propagated across multi-step traces, including outcome-level vs. step-level rewards, novel reward designs, and return decomposition; (iii) Multimodal Research Agents: end-to-end multimodal agents that use a multimodal backbone (e.g., vision language models (VLMs)) to perform iterative perception–reasoning cycles; we prioritize works that internalize multimodal competence rather than delegating it primarily to external tools (e.g., pure OCR calls).

For quick reference, Table 2, 3, 4 summarize the papers covered in Section 3, reporting their backbone policy models, whether a cold start is used before RL (e.g., SFT/RSFT/none), and the reward types employed (outcome/format/etc). For brevity, we report only the largest backbone size from each model family when multiple scales are used in the same paper (e.g., if a paper trains on both Qwen2.5-3B/7B and Llama-3-8B/70B, we list only Qwen2.5-7B and Llama-3-70B). Cold start refers to the SFT/RSFT to learn reasoning skeletons, tool invocation, and answer formats. Outcome

reward evaluates the correctness of the final answer (e.g., EM/F1), while format reward checks compliance with reasoning skeletons and parsing requirements (e.g., final answer and tool invocations formats). Additional reward designs are discussed in Section 3.2.

3.1 Training Regime and Optimization Structure

In this section, we focus on regimes that go beyond the widely adopted DeepSeek-R1- and Search-R1-style baseline: a simple, effective two-stage process with an optional cold start followed by reinforcement learning to align the model with long-horizon objectives. On top of this baseline, recent work introduces significant innovations to handle increasing task complexity and tool-augmented environments.

The Standard Agentic RL Pipeline Before diving into per-paper innovations, we briefly outline the common RL training pipeline used in deep research agents, using Search-R1 as the canonical reference. First, some works apply a cold start (e.g., supervised fine-tuning or rejection sampling fine-tuning (SFT/RSFT)) to teach interface compliance (e.g., tool schemas) and stabilize early rollouts; others skip this step. During RL training, the policy is given a complex query and generates a trajectory τ with interleaved reasoning and tool use. A prompt template enforces a parseable, ReAct-style (Yao et al., 2023) structure with explicit tags, i.e., `<think>...</think>` for reasoning, `<search>...</search>` to trigger retrieval, `<information>...</information>` for injected results, and `<answer>...</answer>` for the final response. This supports multi-turn search-and-reason loops until the model answers or an action budget is exhausted.

> **[Search-R1 Template]**
> Answer the given question. You must conduct reasoning inside `<think>` and `</think>` first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by `<search>` query `</search>`, and it will return the top searched results between `<information>` and `</information>`. You can search as many times as you want. If you find no further external knowledge needed, you can directly provide the answer inside `<answer>` answer `</answer>` without detailed illustrations. For example, `<think>` xxx `</think>`. Question: question.

Many pipelines combine an outcome reward (i.e., final answer correctness) with a small format reward to encourage well-formed traces. While most works adopt a composite reward of outcome and format, some work (including Search-R1) uses outcome-only, rule-based rewards (e.g., EM on extracted answers) and shows that this suffices under their setup. Optimization is typically PPO or GRPO with KL regularization to a reference policy. Crucially, in tool-augmented optimization, tokens generated by tools are masked out so that gradients (and KL) are computed only on model-generated tokens, thereby stabilizing training with interleaved tool use. Advantages come from GAE in PPO or group baselines in GRPO.

Putting this together, the baseline loop is: (i) optional cold start; (ii) templated rollouts with explicit tags and an action budget; (iii) tool returns injected into the context; (iv) reward computation (at least outcome; optional format); (v) policy update via PPO/GRPO with KL to a reference and masking of non-policy tokens. This yields a policy that learns the reasoning skeleton and when/how to invoke tools, with stability anchored by the reference-KL and token masking.

3.1.1 Cold-Start Choices

While half of the papers in Table 2, 3 and 4 skip a cold start for simplicity, the rest adopt a two-stage pipeline (cold start with SFT/RSFT followed by RL training), and report clear benefits: improved early-stage stability (e.g., avoiding reward collapse) and faster, more sample-efficient convergence by quickly mastering formats (Li et al., 2025c; Tao et al., 2025; Dong et al., 2025a; Tan et al., 2025b). For example, *RAG-R1* (Tan et al., 2025b) applies SFT before RL and argues that SFT is critical for leveraging both internal and external knowledge; *WebSailor* (Li et al., 2025c) shows that a modest RSFT cold start is indispensable for complex web tasks because early RL signals are extremely sparse and learning is slow due to multi-turn, heavy tool use; Their ablation comparing direct RL vs. RFT → RL finds that the cold-started model converges to much higher final performance. In the same spirit, *ARPO* (Dong et al., 2025a) adopts a cold start before RL training explicitly to mitigate

reward collapse during the initial RL phases. Benefits also appear in other modalities, e.g., video (*Ego-R1*), visual (*WebWatcher*), and table reasoning (*Reasoning-Table*). Despite these trends, the optimal extent and duration of any cold start prior to RL remain open questions (Shi et al., 2025a).

3.1.2 CURRICULUM OVER THE PIPELINE

Another line of research strengthens the training regime by incorporating curriculum over the standard pipeline. *EVO-RAG* (Ji et al., 2025) introduces a two-stage curriculum: *discovery* to encourage broad, diverse querying, followed by *refinement* that steers the agent toward concise, targeted queries for evidence-grounded answers. *Writing-RL* (Lei et al., 2025b) generalizes this idea to multi-stage curricula and adds *margin-aware data selection*, which estimates learning headroom (the gap between the policy's output and the strongest reference) to prioritize high-potential samples; it further reports consistent gains of curriculum over non-curriculum training. Building on these ideas, *EvolveSearch* (Zhang et al., 2025a) applied curriculum learning iteratively to both SFT and RL training, achieving further performance improvements. In all cases, the optimizer (e.g., PPO/GRPO) remains standard; the innovation lies in how data are staged, scheduled, and escalated in difficulty throughout training.

3.1.3 OPTIMIZERS IN PRACTICE

In terms of RL optimizers, most recent papers adopt GRPO as the primary algorithm (e.g., Chen et al. (2025c); Shi et al. (2025a); Singh et al. (2025); Java et al. (2025); Zhang et al. (2025a); Gao et al. (2025); Fan et al. (2025)), reporting strong benchmark results. A second line of work uses PPO as the main trainer (e.g., Jin et al. (2025a); Zhou et al. (2025b); Tan et al. (2025b); Jiang et al. (2025b)). Studies that compare both (e.g., Jin et al. (2025a); Sun et al. (2025); Zhao et al. (2025c)) generally find that GRPO has simpler mechanics and converges with fewer updates, whereas PPO tends to provide greater training stability under noisy, long-horizon returns; final reward quality is often comparable. Beyond the GRPO/PPO choice, several alternatives appear. *R1-Searcher* (Song et al., 2025a) employs REINFORCE++, observing that GRPO generalizes better out-of-domain while REINFORCE++ achieves higher data efficiency and stronger in-domain scores. *ZeroSearch* (Sun et al., 2025) uses vanilla REINFORCE, reporting gains over PPO/GRPO within its pipeline. *WebSailor* (Li et al., 2025c) introduces Duplicating Sampling Policy Optimization (DUPO), an improvement over DAPO (Yu et al., 2025c), that adds two dynamic sampling strategies: (i) a pre-training filter to drop trivially easy queries (all rollouts correct), and (ii) in-training duplication of queries with non-zero return variance. DUPO keeps batches informative without extra sequential rollouts and reports a ~ 2–3× speed-up over DAPO's dynamic sampling. Orthogonal to the optimizer itself, *ARPO* (Dong et al., 2025a) augments trajectory-level RL with entropy-triggered partial rollouts at tool steps—branching only when post-tool token entropy spikes—together with segment-aware advantage attribution so shared prefixes and branched continuations receive different credits; this targets exploration where tool feedback raises uncertainty and improves tool efficiency without inflating rollout cost.

3.1.4 CONTEXT CONTROL

Existing research also targets limitations of current agentic training pipelines, such as high memory consumption and limited adaptability (Zhou et al., 2025b; Shi et al., 2025c; Li et al., 2025c). *MEM1* (Xu et al., 2025) observes that prompt length grows across multi-turn search and addresses this by replacing accumulated history with a compact internal state that is rewritten at every step; earlier turns are pruned, so the working context remains near-constant while still carrying forward the necessary reasoning and tool observations. *AutoRefine* (Shi et al., 2025c) tackles the issue with a search-and-refine loop that interleaves retrieval with explicit evidence distillation: after each search, long documents are compressed into short "refine notes" that preserve only the crucial evidence to be used in subsequent steps, preventing prompt growth without sacrificing fidelity. *WebSailor* (Li et al., 2025c) tackles the context issue from another angle: it employs a lightweight rejection-sampling fine-tuning step to reconstruct verbose reasoning and tool-use traces into concise, consistent, action-oriented sequences, thereby improving trace quality and reducing prompt bloat without sacrificing fidelity. Research on agent architecture and coordination designs also recognizes this issue (see Section 5). In brief, these systems decouple a domain-agnostic planner from domain-specific executors,

and aggregate/distill intermediate results before returning them to the planner, keeping the planner's working context short, clean, and stable over long horizons.

3.1.5 LEARNING SEARCH NECESSITY

Recent work makes "when to search" a learned decision so agents lean on parametric knowledge and retrieve only when needed. *R1-Searcher++* (Song et al., 2025b) separates internal vs. external traces in SFT (with masked external text), then uses outcome-based RL plus a group penalty and a memorization module that rewrites retrieved facts into internal traces—reducing future lookups. *Frugal-RAG* (Java et al., 2025) runs a lightweight SFT to learn high-recall exploration from ReAct rollouts, then applies GRPO to learn an explicit STOP action that trades additional queries against confidence, adapting test-time compute per question. *IKEA* (Huang et al., 2025b) builds a knowledge-boundary-aware policy: prompts bias toward internal recall, GRPO uses rewards that favor correct answers with fewer retrievals, and a balanced mix of internal-answerable vs. external-required examples teaches the agent to search only when its own knowledge is insufficient. Complementarily, SSRL trains a self-search behavior (internal retrieval within the trajectory) via RL and then swaps to real search at inference, effectively learning to defer external queries until necessary (Fan et al., 2025).

3.1.6 COST & LATENCY-AWARE TRAINING

To rein in the cost of live retrieval and keep inference responsive, two complementary strategies have emerged. Simulation-based training removes web/API calls during RL: *ZeroSearch* (Sun et al., 2025) replaces the real engine with an LLM-based search simulator, enabling cheap, unlimited rollouts while simulating real-web dynamics by employing a noise curriculum, cutting API spend by orders of magnitude without hurting QA quality. *SSRL* (Fan et al., 2025) pushes this idea inside the model, structuring a self-search loop within the trajectory and optimizing it with GRPO; training remains fully in-simulation yet transfers well to real search, yielding substantial speedups. Orthogonally, *RAG-R1* (Tan et al., 2025b) reduces per-question latency by issuing multi-query searches in parallel and fuses evidence, teaching the agent when and how to search efficiently, thus reducing retrieval rounds and end-to-end time without sacrificing EM.

3.1.7 ASYNCHRONOUS ROLLOUT AND TRAINING

ASearcher (Gao et al., 2025) implements a fully asynchronous actor–learner design that decouples rollout generation from policy updates and tolerates stragglers, enabling very long-horizon trajectories with heavy tool use (search+browse) without idling the learner. Summarization and evidence aggregation are learned end-to-end within the RL loop (GRPO), while a simple dynamic filter removes zero-signal prompts. Although the contribution is primarily systems-level, these choices alter the feasible training regime—unlocking stable, sample-efficient long-context training—so we place it here; for engineering details and broader agentic RL training frameworks, see Section 4.

3.1.8 DISCUSSION

Across recent work, the baseline pipeline has solidified: optional cold start to teach interfaces, templated rollouts with explicit tool tags and action budgets, outcome(-plus-format) rewards, and PPO/GRPO with KL-to-reference while masking tool-return tokens. On top of this, three themes stand out. First, *stability and efficiency*: cold starts and curricula speed convergence and prevent early reward collapse; token masking and reference-KL anchor learning despite interleaved tool text. Second, *data/compute shaping*: curricula, dynamic sampling (e.g., duplication/filters), and entropy-triggered branching focus updates where uncertainty and learning headroom are highest. Third, *cost/latency control and search necessity*: context-control mechanisms (state rewriting, evidence distillation) and policies that learn when to search (including STOP actions and knowledge-boundary awareness) reduce retrieval rounds, while simulators and parallel query strategies cut training/inference costs without sacrificing answer quality. Optimizer choice (GRPO vs. PPO vs. REINFORCE-family) largely trades simplicity and speed for robustness under long-horizon, noisy returns; final quality is often comparable when the surrounding regime (masking, KL, curricula) is well-tuned.

We list three open questions in this area: (i) Cold start and curriculum scheduling: How should we automatically decide *when* to stop SFT/RSFT, advance curriculum phases, or switch difficulty to

maximize sample efficiency without overfitting? (ii) Optimizer–tool interaction: What principled criteria pick PPO/GRPO/REINFORCE++ under partial, delayed, and segment-wise credit with tool boundaries—and can segment-aware advantage attribution be unified with KL control for stronger stability? (iii) Cost-aware objectives: How do we train truly multi-objective policies that optimize accuracy and explicit budgets (latency, queries, tokens), with guarantees on test-time compute allocation (reasoning depth, retrieval rounds)?

3.2 Reward Design and Credit Assignment

Reward design and credit assignment are central to RL training. While training regime and optimization structure dictate how learning unfolds, reward strategies determine which behaviors are reinforced and how credit is allocated across complex trajectories.

Table 3: Summary of papers in Section 3.2.

Work	Policy Model	Cold Start?	Reward Type	RL Optimizer
s3 (Jiang et al., 2025b)	Qwen2.5-7B-Instruct	✗	Gain_Beyond_RAG	PPO
AutoRefine (Shi et al., 2025c)	Qwen2.5-3B-Base/Instruct	✗	Outcome +Retrieval_Quality	GRPO
MT-GRPO (Zeng et al., 2025b)	Qwen2.5-7B-Base	✗	Outcome+Format +Retrieval_Quality	MT-GRPO (GRPO-variant)
IKEA (Huang et al., 2025b)	Qwen2.5-7B-Base/Instruct	✗	Outcome+Format +Retrieval_Necessity	GRPO
ARTIST (Singh et al., 2025)	Qwen2.5-14B-Instruct	✗	Outcome+Format +Tool_Execution	GRPO
R1-Searcher++ (Song et al., 2025b)	Qwen2.5-7B-Instruct	✓	Outcome+Format +Retrieval_Necessity	Reinforce++
StepSearch (Wang et al., 2025e)	Qwen2.5-7B-Base/Instruct	✗	Outcome+Format +(InfoGain-Redundancy)	StePPO (PPO-variant)
O^2-*Searcher* (Mei et al., 2025a)	Qwen2.5-3B-Instruct	✓	Outcome+Format +LongText_Metrics	GRPO
R-Search (Zhao et al., 2025c)	Qwen2.5-7B-Instruct	✗	Outcome+Format +Evidence	PPO, GRPO

3.2.1 Outcome-level Rewards

Most studies frame multi-turn interaction as a bandit problem (Zeng et al., 2025c), evaluating performance at the terminal step (e.g., answer correctness and formatting). We organize outcome-level rewards into: (i) classical metrics, (ii) composition strategies, and (iii) novel outcome-level signals that target search/evidence/efficiency beyond correctness and format.

Classical metrics Terminal rewards traditionally score the final response with (i) exact match (EM) / F1 for correctness, (ii) LLM-as-a-judge graders for holistic quality, and (iii) task metrics such as NDCG in retrieval-style tasks. Representative examples include rule-based paper-query alignment in PaSa (He et al., 2025), LLM-graded writing quality in Writing-RL (Lei et al., 2025b), loose/strict LLM graders in DeepDiver (Shi et al., 2025a), and task-specific metrics (e.g., NDCG) in SAGE (Wang et al., 2025b) and VRAG-RL (Wang et al., 2025a).

Composition strategies Outcome signals are typically composed via: (i) additive (weighted sums) that combine format and correctness (e.g., Mei et al., 2025b; Song et al., 2025b; Zhao et al., 2025c); (ii) hierarchical/conditional schemes that gate the final reward on pass/fail conditions to avoid overpenalizing partially correct reasoning (Ren et al., 2025; Shi et al., 2025b; Huang et al., 2025b;

Lei et al., 2025a); and (iii) group-relative bonuses that compare rollouts within a sampled group (e.g., Song et al., 2025b). These choices control stability and what behaviors are emphasized at the terminal step.

Novel outcome-level signals Beyond standard answer/format scoring, several works introduce outcome-level rewards that target search, evidence, and tool economy directly:

- **Gain-Beyond-RAG (GBR; Jiang et al. (2025b)).** Rewards the *delta* in generation accuracy when using the agent's retrieved context versus a naive top-k RAG baseline, attributing improvement specifically to the searcher.
- **Cross-model evidence utility.** *R-Search* (Zhao et al., 2025b) treats agent-written evidence as a self-contained bundle: a *frozen external LLM* must answer correctly from this evidence; the reward reflects the downstream recoverability of the gold answer.
- **Knowledge-boundary shaping.** *IKEA* (Huang et al., 2025b) designs a hierarchical terminal signal that discourages redundant external calls when the model already knows the answer, while lightly encouraging retrieval when it does not, aligning usage with actual need.
- **Group-relative efficiency.** *R1-Searcher++* (Song et al., 2025b) grants a bonus to trajectories that use the *fewest* external calls among correct rollouts in a sampled group, thereby incentivizing internalization and thriftiness.
- **Query diversity.** O^2-*Searcher* (Mei et al., 2025a) adds a diversity-aware outcome term that promotes non-duplicative queries under a controlled budget, mitigating mode collapse in query generation.
- **Refinement/coverage.** *AutoRefine* (Shi et al., 2025c) gives credit when the final refined knowledge state explicitly contains the required gold-answer components, rewarding evidence consolidation beyond mere format/correctness.

In practice, these novel signals are used alongside classical outcome terms, e.g., O^2-*Searcher*, *R1-Searcher++*, and *R-Search* retain format/answer components while augmenting them with diversity, group-relative, or evidence-utility rewards.

3.2.2 STEP-LEVEL REWARDS

Outcome-level rewards treat the entire decision trajectory as a single decision step, thereby overlooking the multi-turn structure of the task. This limitation hinders the model's ability to learn robust reasoning chains. To address this, researchers (Zeng et al., 2025c; Wang et al., 2025e; Liu et al., 2025b) have explored incorporating step-level rewards alongside outcome-level rewards, enabling more fine-grained credit assignment.

Tool/execution & presence checks To expose fine-grained supervision within a trajectory, several works attach per-turn rewards to how a tool is used and what it returns. *MT-GRPO* (Zeng et al., 2025c) grants a small reward when a tool call is well-formed and successfully executed, and another when the retrieved snippets contain any gold answer span, thereby reinforcing correct invocation and retrieval sufficiency at the moment they occur. *ARTIST* (Singh et al., 2025) further instantiates this pattern with a Function Reward for issuing the correct sequence of function calls and a State Reward for correct state tracking during multi-turn tool use, i.e., explicit step-level execution checks.

Information gain vs. redundancy Instead of merely checking presence, *StepSearch* (Wang et al., 2025e) shapes each turn by rewarding novel, useful evidence and penalizing repeats. Concretely, a step reward adds information gain, i.e., the marginal similarity improvement between the round's retrieved documents and a reference set of gold evidence, minus a redundancy penalty that increases with overlap against earlier rounds, steering the agent toward diversified, high-yield hops.

Query-intent alignment Complementing evidence-level shaping, *StepSearch* (Wang et al., 2025e) adds a query-intent signal: overlap between the model's generated query and reference sub-task keywords for that turn. This per-turn reward keeps search on-task (e.g., targeting the correct sub-question in a decomposition) and reduces drift, without waiting for the final answer to provide feedback.

Multimodal turn checks For multimodal agents, *Visual-ARFT* (Liu et al., 2025b) extends turn-level shaping beyond text by scoring format quality, search/tool usage, and code quality at each step, providing dense feedback that trains the full perception–tool–reason loop rather than only the terminal output.

In practice, these step-level signals are combined with outcome-level rewards to provide dense supervision for multi-turn tasks; reward placement typically attaches step rewards at round boundaries and the terminal reward at the end of the trajectory.

3.2.3 CREDIT ASSIGNMENT

Effective credit assignment is pivotal in multi-turn RL because the agent's useful behaviors (e.g., deciding *when* to search, *what* to retrieve, and *how* to answer) occur at different times; without careful credit flow, terminal rewards either over-credit the last step or under-credit key intermediate decisions.

Trajectory-level estimators Many work use terminal-only signals with standard policy-gradient optimizers—REINFORCE, GRPO, or PPO—plus a KL penalty to a frozen reference model to stabilize updates. In tool-augmented settings, implementations typically mask injected tokens (e.g., pasted documents) and compute loss only on action tokens (tool invocation, reasoning/answer tokens), so gradients do not spuriously pass through the evidence text (Song et al., 2025b; Zhao et al., 2025c; Huang et al., 2025b). This setup assigns a single trajectory-level advantage to the full sequence and is simple and robust, but can be slow to propagate credit to early, causally important steps.

Turn-level estimators To better align credit with the process structure, *MT-GRPO* (Zeng et al., 2025b) computes per-turn advantages that blend immediate turn feedback with final-outcome advantages, improving learning for multi-turn tool use agents. Concretely, early turns receive credit from both their own turn rewards (e.g., search quality) and a portion of the outcome signal, while later turns are outcome-focused, yielding finer attribution than trajectory-level estimators without changing the underlying RL optimizer.

Token/round placement and masking Even with standard optimizers like PPO/GRPO, *where* rewards are attached matters. Process-aware implementations such as *StepSearch* Wang et al. (2025e) attach the terminal reward to the final tokens and attach step rewards at the end of each search round, allowing GAE to propagate credit locally through each round while still reflecting downstream success. Step-level execution rewards in *ARTIST* (Singh et al., 2025) are similarly attached at the tool-call boundary (function and state checks), providing immediate feedback on action quality within the turn. As above, action-token-only loss and observation masking are maintained to avoid leaking gradients through retrieved content.

3.2.4 DISCUSSION

Overall, verifiable terminal rewards remain the anchor, but how they are composed (additive vs. conditional vs. group-relative) and where credit flows (trajectory vs. turn vs. token/round) affect stability and sample efficiency. Novel outcome signals target search, evidence quality, and tool economy; step-level shaping helps, but dense process rewards can trigger reward hacking or tool avoidance, making outcome-first designs with tight action budgets more stable.

Key open questions include: (i) how to compose and schedule multi-objective rewards (e.g., correctness, GBR, diversity, evidence utility) in a principled, anti-hacking way; (ii) how to build robust, low-cost verifiers that replace expensive or easily gamed LLM judges without sacrificing alignment; (iii) how to deliver causal, low-variance credit to search/tool choices via turn/token-level methods; and (iv) how to learn budget- and risk-aware policies that trade off accuracy, latency, and tool cost without inducing tool avoidance.

3.3 MULTIMODAL RESEARCH AGENTS

As deep research agents expand beyond text, a central question is how to build agents whose policy natively perceives and reasons over multiple modalities (e.g., vision and language). This section sur-

veys end-to-end multimodal models (typically VLMs) that perform iterative perception–reasoning cycles. We explicitly exclude works that outsource multimodality to off-the-shelf OCR/vision modules, code interpreters, or retrieval heuristics. Instead, we focus on models that ingest visual evidence directly and produce grounded reasoning traces and answers within a unified multimodal token space. Despite recent progress, multimodal deep research agents remain comparatively early-stage relative to text-only LLM agents.

Table 4: Summary of papers in Section 3.3.

Work	Policy Model	Cold Start?	Reward Type	RL Optimizer
Visual-ARFT (Liu et al., 2025b)	Qwen2.5-VL-7B-Instruct	✗	Outcome+Format +Retrieval_Quality	GRPO
VRAG-RL (Wang et al., 2025a)	Qwen2.5-VL-7B-Instruct	✓	Outcome+Format +Retrieval_Quality	GRPO
WebWatcher (Geng et al., 2025)	Qwen2.5-VL-32B-Instruct	✓	Outcome+Format	GRPO
MMSearch-R1 (Wu et al., 2025c)	Qwen2.5-VL-7B-Instruct	✗	Outcome+Format	GRPO
V-ToolRL (Su et al., 2025)	Qwen2-VL-2B-Instruct	✓	Outcome	GRPO
VTool-R1 (Wu et al., 2025d)	Qwen2.5-VL-32B-Instruct	✗	Outcome	GRPO

3.3.1 MULTIMODAL ACTION–OBSERVATION INTERFACE

Across recent multimodal deep research agents, the RL optimizer is essentially unchanged (GRPO/PPO with a KL reference, masking tool-generated tokens, and optimizing only final-answer tokens); the real novelty lies in the *state/action/observation* contract (Liu et al., 2025b; Wang et al., 2025a; Geng et al., 2025; Wu et al., 2025c; Su et al., 2025; Wu et al., 2025d), as shown in Table 4. In multimodal settings, *perception becomes an action*: the policy issues visual operations-(i) region crop/zoom reprojected to raw pixels for true DPI on dense charts/tables, (ii) edit-then-reason steps (highlight/mask/box) with dual-image conditioning ($I \oplus I'$), and (iii) lightweight image code (rotate/denoise/brighten)-and then *re-conditions* on their consequences, turning one-shot perception into a controllable evidence-update loop (Wang et al., 2025a; Wu et al., 2025d; Liu et al., 2025b). This works only with observation engineering for long contexts and noise control (e.g., raw-pixel crops over thumbnails; reader→summary transforms for webpages) and with unified, tagged action schemas (visual acts and web/text search under `<action>...</action>`) coupled to tight action budgets for stability (Wu et al., 2025c; Su et al., 2025; Geng et al., 2025).

Observation Engineering & Grounding Because images are high-entropy, agents must see *usable* evidence. Effective choices include (a) re-encoding raw-pixel crops (not encoder-space thumbnails) to recover small text and chart detail (Wang et al., 2025a), (b) dual-image reinsertion after edits to externalize attention (Wu et al., 2025d), (c) strict, typed tool-return schemas with controller-level normalization/caching (Su et al., 2025), and (d) reader→summary webpage processing to suppress boilerplate (Wu et al., 2025c). Trajectory quality gates—schema checks, step-consistency filters, and *entity obfuscation paired with retrieved images* to force genuine visual grounding—prevent "answer-from-prior" shortcuts (Geng et al., 2025).

Learning Evidence Necessity In multimodal agents, the classic "when to search" problem generalizes to *evidence necessity*: choosing which modality to query first (image vs. text), whether to perform perception actions (crop/zoom, highlight/mask, lightweight image code) before or instead of retrieval, when to hand off from image search to text reading, and when to stop. Recent agents implement this via a unified action space (including NO-OP/FINISH) and outcome-centric rewards augmented with light, modality-aware efficiency signals—e.g., a *search penalty* for thrift in web calls (Wu et al., 2025c), trajectory-level *image-retrieval ranking* (NDCG) to surface the right visual

earlier (Wang et al., 2025a), and a binary *executable-image-code* check to encourage safe preprocessing exploration (Liu et al., 2025b). Datasets are balanced across *search-/vision-required* vs. -free items and filtered with strict schema/consistency gates to teach selectivity rather than reflexive tool use; the optimizer remains standard while evaluation couples quality with thrift (search ratio, perception-action counts, image NDCG, executable-code rate) (Wu et al., 2025c; Wang et al., 2025a; Liu et al., 2025b).

3.3.2 DISCUSSION

Recent progress in multimodal agents is driven less by new optimizers and more by perception-as-action, engineered observations, and a lean RL recipe (GRPO/PPO+KL with tool-token masking and outcome-first rewards). Together, these techniques transform brittle one-shot reads into controllable evidence updates. They improve stability through design choices such as raw-pixel crops, dual-image reconditioning, and reader-to-summary transforms, while also instilling evidence necessity—deciding which modality to use and whether to perceive before retrieving—in order to curb over-search and strengthen grounding.

We identify three promising directions for multimodal deep research agents: (i) tracing performance gains back to specific perception steps or image regions; (ii) multi-image/multi-page reasoning at scale (PDFs, dashboards) without exploding context; (iii) developing standardized *process+thrift* benchmarks and reporting (action budgets, masking policy, cache/rate limits) for apples-to-apples comparison.

4 AGENTIC RL TRAINING FRAMEWORKS

Deep research agents learn through long, tool-using interactions; making them trainable with RL is therefore a systems problem. This section surveys open-source infrastructure for agentic RL released in 2025 and is organized around three questions: (1) what bottlenecks currently limit training, (2) what design patterns and system mechanisms recent frameworks contribute to address them, and (3) how to choose and compose frameworks in practice.

4.1 BOTTLENECKS & CHALLENGES IN AGENTIC RL TRAINING

We identify five recurring bottlenecks and challenges in current agentic RL training:

- **Rollout throughput and latency.** Long, multi-turn, tool-using episodes stall GPUs (e.g., longest-sample tails, synchronization barriers), so experience collection often dominates end-to-end cost (Fu et al., 2025).

- **Policy staleness and unstable credit assignment.** Asynchronous or mixed-policy batches violate standard PPO-style assumptions, while sparse/delayed rewards over long trajectories increase variance and hinder stable improvement; staleness-aware objectives and stepwise credit assignment are typically required (Fu et al., 2025).

- **Large-scale orchestration.** Switching the same model between training and generation, mapping TP/PP/DP (and MoE/EP) parallelism, and moving tensors without redundancy are non-trivial at cluster scale; efficient, zero-redundancy train↔gen transitions remain a core systems challenge (Sheng et al., 2025).

- **Heterogeneous agent runtimes.** Production agents (e.g., LangChain/AutoGen/custom stacks) are tightly coupled to tools and execution logic, making it difficult to "drop in" RL without refactoring; clean trainer–agent disaggregation and unified transition interfaces are needed (Luo et al., 2025).

- **Outcome-only supervision is weak for multi-step reasoning.** Pure outcome reward signals under-specify long-horizon behavior; process-aware supervision (e.g., PRMs), verifiable/tool-aware rewards, and structured protocols are needed to shape trajectories (Wang et al., 2024).

Table 5: Open-source agentic RL training frameworks for LLMs (listed alphabetically).

Framework	Key features	GitHub
Agent Lightning (Luo et al., 2025)	Trainer–agent disaggregation (server–client); unified transition/MDP interface; LightningRL credit assignment for multi-turn traces; telemetry & automatic intermediate rewards (AIR)	⭘ repo
AReaL (Fu et al., 2025)	Fully asynchronous actor-learner with high MFU; interruptible/cancelable decoding; staleness-aware PPO for mixed-policy batches; dynamic packing & parallel reward service	⭘ repo
OpenR (Wang et al., 2024)	PRM-centric process supervision; PRM-guided decoding/test-time compute; unified data + online/offline RL stack; gym-style MDP with PPO	⭘ repo
rLLM (Tan et al., 2025a)	Async, OpenAI-compatible rollout engine; stepwise GRPO variants for long horizons; observation masking (action-only loss); verl-backed distributed PPO/GRPO	⭘ repo
ROLL (Wang et al., 2025d)	Single controller + workerized Actor/Critic/Env/Reward; sample-level scheduler & async reward workers; AutoDeviceMapping for heterogeneous clusters; 5D parallelism with vLLM/SGLang backends	⭘ repo
SLIME (Xie et al., 2025)	SGLang-native serving (router, PD/EP); Megatron-native trainer, large-MoE ready; async/decoupled rollout–train via Ray; abort in-flight & frequent weight sync	⭘ repo
Verifiers (Brown, 2025)	Protocol-first MultiTurn/Tool environments; OpenAI-compatible with vLLM rollout; built-in GRPO trainer (Accelerate/DeepSpeed/LoRA); rubric/judge-based, tool-aware rewards	⭘ repo
verl (Sheng et al., 2025)	HybridFlow single-/multi-controller APIs; 3D-HybridEngine for zero-redundancy resharding across train ↔ gen; scalable 3D/ZeRO/FSDP/Megatron parallelism with auto device mapping; high-throughput RLHF/RLAIF pipelines.	⭘ repo

4.2 WHAT THE FRAMEWORKS CONTRIBUTE (METHODS & FEATURES)

To address the aforementioned challenges, the mainstream open-source training frameworks propose the following methods and features.

Addressing rollout throughput and latency. To make rollouts fast and elastic, recent systems raise sampling throughput by adopting fully asynchronous actor–learner designs that remove batch-wide waits. *AReaL* (Fu et al., 2025) proposes interruptible/cancelable decoding together with staleness-aware PPO to stabilize learning while boosting Model Flops Utilization (MFU). Serving-native RL sampling further helps: *SLIME* (Xie et al., 2025) binds SGLang (Zheng et al., 2024) (prefill–decode disaggregation and expert parallelism) to Megatron training, exposes an OpenAI-compatible router, and adds sampler features such as abort-in-flight tailored to dynamic or oversampling recipes. Complementarily, *ROLL* (Wang et al., 2025d) contributes sample-level schedulers and dedicated environment/reward workers to coordinate per-sample lifecycles, asynchronous rewards, and tool sandboxes across large clusters.

Bridging heterogeneous agent runtimes. *Agent Lightning* (Luo et al., 2025) reduces refactoring costs for production agents by introducing trainer–agent disaggregation with a unified transition interface, allowing existing stacks to be wrapped and treated as MDPs without code rewrites. In parallel, *Verifiers* (Brown, 2025) adopts a protocol-first design with OpenAI-compatible endpoints, en-

abling the same module to support evaluation, data generation, and RL, and providing a lightweight GRPO path for multi-turn tool use.

Stabilizing credit assignment & process supervision. For long-horizon behavior, *rLLM* introduces hierarchical/stepwise GRPO variants that propagate or group advantages across steps, and *Agent Lightning*'s LightningRL adds explicit credit assignment over multi-turn traces (Tan et al., 2025a; Luo et al., 2025). *OpenR* (Wang et al., 2024) integrates Process Reward Models (PRMs) and guided decoding to inject step-level signals into both training and test-time search, while *Verifiers* (Brown, 2025) supplies rubric/judge-based, tool-aware rewards with parsers that make multi-criteria, process-aware supervision practical.

Systems co-design for scale *verl* (Sheng et al., 2025) provides a hybrid single-/multi-controller programming model and a 3D-HybridEngine for zero-redundancy resharding across train↔gen, yielding substantial throughput gains across RLHF-style recipes. *ROLL* (Wang et al., 2025d) standardizes a single controller with parallel Actor/Critic/Reward/Env workers and AutoDeviceMapping, integrating Megatron-Core/FSDP for training and vLLM/SGLang for serving. *SLIME* (Xie et al., 2025) packages large-MoE Megatron examples and server-managed SGLang pools behind one endpoint, easing frontier-scale agentic runs and simplifying deployment pathways.

Ecosystem-level conveniences Beyond core bottlenecks, frameworks add observability and shape signals to improve diagnosability and training efficacy. For example, *Agent Lightning* (Luo et al., 2025) integrates telemetry and automatic intermediate rewards into the runtime. Meanwhile, unified, end-to-end stacks are becoming common: *rLLM* (Tan et al., 2025a) combines agents, environments, and stepwise GRPO; *OpenR* (Wang et al., 2024) and releases code, models, and data for PRM-centric workflows.

4.3 How to choose (pragmatic guidance)

When the goal is a training back end that "just runs" PPO/GRPO with minimal glue, verl is a pragmatic starting point: it is production-ready, flexible, and its HybridFlow APIs simplify scalable train/gen switching and parallelism. It fits teams that already have a sampler and primarily need a robust engine for training and generation orchestration. If raw rollout throughput on long, tool-using episodes is the primary constraint, AReaL and SLIME are purpose-built for high MFU and dynamic sampling: AReaL provides a research-grade asynchronous blueprint with staleness-aware updates, while SLIME's SGLang-native sampling (with aborts and frequent weight refresh) targets serving-side speed under Megatron training (Fu et al., 2025; Xie et al., 2025). ROLL is a good fit when per-sample scheduling, environment/reward workers, and heterogeneous cluster management need to live in one library (Wang et al., 2025d).

If you already operate a production agent and you prefer not to rewrite, Agent Lightning's server–client split and unified transition interface lets you keep existing agent logic and tools while retrofitting RL (Luo et al., 2025). Domains that demand process-aware rewards or verifier-style signals benefit from OpenR's PRMs and guided decoding for reasoning-centric research and from Verifiers' protocol/rubric tooling (judge- and tool-aware rewards) coupled with a lightweight GRPO trainer (Wang et al., 2024; Brown, 2025). For frontier-scale parallelism or MoE, verl offers zero-redundancy transitions across train ↔ gen with flexible device mapping, while ROLL or SLIME become attractive when you want, respectively, baked-in orchestration or serving-native scaling (Sheng et al., 2025; Wang et al., 2025d; Xie et al., 2025).

A blended path many teams use is to adopt verl (or ROLL) as the training spine, choose SGLang or vLLM for sampling, and bring Verifiers (or a custom PRM/judge) for rewards; if a mature agent stack already exists, Agent Lightning can attach RL without refactors, and if throughput remains the blocker, asynchronous sampling such as AReaL or SLIME can be swapped in to raise utilization.

4.4 Discussion

This section frames agentic RL training as a systems problem and surveys recent frameworks through three lenses: (i) core bottlenecks; (ii) architectural responses—asynchrony, distributed

orchestration, serving-native sampling, trainer–agent disaggregation, process-aware supervision hooks, and training–generation co-design; and (iii) pragmatic composition choices for training.

Looking ahead, we highlight three open questions for framework development: (i) Safety and robustness in online rollouts: What sandboxing, fault isolation, and guardrails are needed for browser/code/tool interactions under continual learning? (ii) Orchestration, elasticity, and fault tolerance at scale. How should schedulers handle preemption, multi-tenant QoS, partial-trajectory reclaim/cancel, and weight-sync semantics? (iii) Reproducibility, observability, and reporting. What trace/telemetry standards enable deterministic replays under non-deterministic tools/web, and which common "process+thrift" metrics (e.g., MFU, cost-per-success, staleness histograms, tool budgets) should frameworks report by default for standardized evaluation?

5 AGENT ARCHITECTURE & COORDINATION

While our survey centers on RL foundations for training, the way agents are architected and coordinated is equally critical in practice. End-to-end RL over a full hierarchical stack remains impractical today because rollouts are long and high-latency, credit assignment spans many interacting components, and current infrastructure struggles with deterministic retrieval, tool sandboxes, and large-scale judging. As a result, most deployments compose pre-trained (often RL-enhanced) models into hierarchical or multi-agent systems rather than training the entire workflow jointly.

This chapter surveys those deployment-oriented designs: hierarchical planners (Planner–Coordinator–Executors), multi-agent teams, expert routing/gating, and modular sub-agents with shared state. We focus on how coordination mechanisms (e.g., task decomposition, scheduling, message passing, etc) turn individually trained models into scalable, robust deep research systems. Our aim is to outline the patterns and trade-offs that make these architectures effective today, and to clarify how stronger RL-trained planners can later slot into them to extend capability without incurring full end-to-end training cost.

5.1 OPEN-SOURCE ARCHITECTURES & COORDINATION

We analyze four well-maintained open-source deployment frameworks for deep-research systems listed in Table 6, focusing on how they coordinate planning, delegation, tool use, and report assembly. These projects are not agentic RL training stacks: they emphasize multi-agent architecture design and production orchestration rather than on-policy RL training. For each framework, we summarize along four axes: (i) *Planning & Roles*, (ii) *Tools & Interfaces* (search/crawl/exec/MCP), (iii) *Human Oversight & Observability*, and (iv) *Evaluation & Reporting*.

These frameworks differ primarily in where planning resides (implicit recursive loop vs. explicit planner), how specialization is expressed (single agent with internal phases vs. multi-agent roles), and production affordances (e.g., MCP connectors, structured logging). These design choices drive concrete trade-offs in cost and latency, achievable depth, reproducibility, and extensibility for future tools.

Table 6: Open-source deep research frameworks (sorted by published date). Note: GitHub stars are as observed on 06 Sept 2025.

Open-Source Frameworks	Organization	Stars	Date
Open Deep Research (Zhang & contributors)	Aomni	~17.6k	4 Feb 2025
DeerFlow (ByteDance & contributors)	ByteDance	~16.8k	9 May 2025
Open Deep Research (LangChain & contributors)	LangChain	~8.5k	16 Jul 2025
MiroFlow (MiroMindAI & contributors)	MiroMind AI	~441	8 Aug 2025

System Snapshots

- **Aomni Open Deep Research.** *Planning & Roles:* Single-agent recursive loop: generate queries → fetch & extract → summarize/reflect into learnings and next directions → recurse

19

until a depth cap, then render a Markdown report. SERP concurrency is configurable. No explicit planner/coordinator roles, i.e., planning is implicit in the next-direction proposal.
Tools & Interfaces: Firecrawl (search + extraction); model backends via Fireworks (e.g., OpenAI, DeepSeek R1).
Human Oversight & Observability: None; no dedicated planner review UI.
Evaluation & Reporting: Minimal; intended as a clean baseline rather than a full product stack.

- **ByteDance DeerFlow.** *Planning & Roles:* Explicit planner decomposes tasks; a coordinator manages lifecycle; a research team of specialists (e.g., researcher, coder) executes; a reporter aggregates/formats outputs. Orchestrated with LangGraph (stateful graph).
  Tools & Interfaces: Tavily/Brave for search, Jina for crawling, optional private KB (RAGFlow), Python execution; broad MCP support.
  Human Oversight & Observability: Optional plan review/auto-accept; rich report post-editing and TTS. LangGraph Studio & LangSmith tracing; Docker/Compose for backend + frontend.
  Evaluation & Reporting: Tests/examples provided; no standardized leaderboard.

- **LangChain Open Deep Research.** *Planning & Roles:* A LangGraph agent with configurable subtasks (summarization, research, compression, final report). The repo also ships two *legacy* topologies for comparison: (1) plan-and-execute (human-reviewable section plan) and (2) supervisor–multi-researcher (parallelism for speed).
  Tools & Interfaces: MCP-compatible; switchable across multiple model/search providers.
  Human Oversight & Observability: Supported via the plan-and-execute variant (human review of the section plan). Runs in LangGraph Studio / OAP UI.
  Evaluation & Reporting: First-class Deep Research Bench harness.

- **MiroMind MiroFlow.** *Planning & Roles:* A pipeline boots an orchestrator that manages multi-turn tool calls, delegates to sub-agents (e.g., browsing) with their own toolsets/prompts, and aligns outputs via a dedicated summarizer/formatter.
  Tools & Interfaces: Unified LLM client (OpenRouter/Anthropic/OpenAI/etc), MCP Tool Manager with FastMCP servers (search, vision, code, audio, reading, reasoning), optional E2B sandbox.
  Human Oversight & Observability: Built-in web UI for review/edit; built-in logging.
  Evaluation & Reporting: GAIA evaluation scripts with reproducible metrics (e.g., pass@1 / avg@3).

Design Patterns & Trade-offs   We synthesize differences using the same four axes to surface key trade-offs in reproducibility, throughput, and auditability.

- **Planning & Roles.** *Aomni* uses *implicit* planning inside a recursive loop; *DeerFlow* adopts an *explicit* planner with a coordinator; *LangChain* supports both a single-agent graph and legacy plan-and-execute / supervisor–multi-researcher variants; *MiroFlow* centralizes planning in an orchestrator with hierarchical sub-agents. This choice sets reproducibility, human-in-the-loop (HITL) insertion points, debugging granularity, and the simplicity–specialization balance (*Aomni* is simple and quick to adapt, while *DeerFlow*/*MiroFlow* gain throughput and tool coverage from specialist teams at the cost of coordination overhead; *LangChain* provides a controlled setting to compare these topologies under one roof).

- **Tools & Interfaces.** *DeerFlow*, *LangChain*, and *MiroFlow* expose broad MCP connectors; Aomni intentionally keeps a narrow interface (Firecrawl + LLM). Wider tool surfaces improve enterprise readiness but enlarge failure modes and complicate evaluation.

- **Human Oversight & Observability.** *DeerFlow* emphasizes plan review and report editing; *DeerFlow* and *LangChain* leverage LangGraph Studio; *MiroFlow* ships logging and a visual UI; *Aomni* is deliberately bare-bones. These choices impact auditability and developer velocity.

- **Evaluation & Reporting.** *LangChain* integrates *Deep Research Bench*; *MiroFlow* provides *GAIA* scripts; *DeerFlow* includes tests/examples but no standardized leaderboard; Aomni serves as a minimal baseline. Built-in harnesses enable apples-to-apples comparisons.

Actionable Insights

- **Use Aomni as a loop baseline.** Ideal for isolating the effects of recursion depth/breadth and for ablating query-generation and reflection strategies without multi-agent confounds.
- **Prototype explicit planners on DeerFlow.** The planner-coordinator-team-reporter split is a natural substrate for studying hierarchical credit assignment (planner-level vs. worker-level), coordinator scheduling, and HITL gating policies.
- **Run controlled architecture comparisons on LangChain.** Hold the environment and tool surface constant, then toggle among single-agent, plan-and-execute, and supervisor–multi-researcher graphs to quantify quality/latency/cost trade-offs.
- **Stress-test production affordances on MiroFlow.** The tool manager, sub-agents, and observability suite support measuring failure recovery, tool-latency masking, backpressure handling, and reproducibility under load (e.g., GAIA).

5.2 ACADEMIC ARCHITECTURES & COORDINATION

We review six representative deployment frameworks for deep research systems from academia that integrate search and other tools into agentic workflows, emphasizing multi-agent or multi-component designs and the recurring patterns behind them.

Table 7: Representative academic systems for deep research.

Work	Domain	Agent Architecture
OWL (Hu et al., 2025)	Multi-agent assistant	Planner-Coordinator-Executors
CoA (Li et al., 2025e)	Multi-tool (web + code)	Single AFM with role-activated agents
PaSa (He et al., 2025)	Academic search	Crawler & Selector agents
WebThinker (Li et al., 2025f)	Open-web research	Single large reasoning model loop with explore/draft actions
HiRA (Jin et al., 2025b)	Deep search	Planner-Coordinator-Executors
DeepResearcher (Zheng et al., 2025)	Real-world web	Single planner with a browsing helper

System Snapshots

- **OWL (Optimized Workforce Learning).** Modular "Workforce" with a *domain-agnostic Planner*, a *Coordinator*, and tool-equipped *Executors* (including web/search), designed to be plug-and-play across domains; only the Planner is optimized post-SFT via real-web (online DPO), boosting cross-domain generalization and achieving open-source SOTA on GAIA.
- **CoA (Chain-of-Agents).** Collapses a multi-agent workflow into a single Agent Foundation Model (AFM), a role-conditioned backbone that dynamically activates tool- and role-agents. Training is two-stage: (1) multi-agent distillation to produce chain-of-agents trajectories for SFT; (2) agentic RL on verifiable tasks (web + code) to refine end-to-end problem solving. Delivers strong results while avoiding joint multi-policy optimization; code, data, and model weights are open-sourced.
- **PaSa (Paper Search Agent).** Two-agent loop for scholarly discovery: a *Crawler* expands a paper queue via search + citation chasing and a *Selector* prioritizes candidates; training uses Imitation Learning (IL) → RL (session-level PPO within AGILE) on *AutoScholarQuery*, with evaluation on *RealScholarQuery*, yielding large recall gains over Google/Scholar baselines.
- **WebThinker.** Embeds a *Web Explorer* directly into the reasoning language model's chain so the model can *think-search-navigate-extract-draft* in one continuous loop; adds drafting/check/edit tools for live report writing; improves tool use via online DPO on trajectories, lifting performance on GAIA/HLE/WebWalker benchmarks.

- **HiRA: Decoupled Planning & Execution.** Three-tier hierarchy, i.e., *Planner - Coordinator - Executors*, that routes subtasks to domain-specialized executors and feeds back distilled results (not raw tool dumps); supports plug-and-play executors and dual-channel memory, improving answer quality and efficiency on complex, cross-modal deep search.

- **DeepResearcher.** End-to-end RL on the open web: a single *planner* policy trained with GRPO learns when/how to search, browse, and synthesize over a compact JSON tool interface. Rewards are automatic (word-level F1 with format penalties), and tool outputs are treated strictly as observations (no gradient on retrieved text). A lightweight *browsing helper* segments long pages and returns distilled snippets, keeping the policy focused on high-level sequencing rather than DOM-level actions. This planner+helper design—rather than a deep hierarchy like HiRA—makes end-to-end RL feasible and delivers strong real-web performance.

Actionable Insights

- **Hierarchical orchestration.** A planner (often via a coordinator) delegates to specialized executors and aggregates *distilled* results to keep long-horizon reasoning clean and scalable (Li et al., 2025f; Jin et al., 2025b; Hu et al., 2025; Li et al., 2025e).

- **Memory-guided iterative retrieval.** Agents refine queries across hops while maintaining scratchpads/knowledge caches and provenance, steering subsequent evidence selection and avoiding search myopia (Zheng et al., 2025; He et al., 2025; Li et al., 2025f).

- **Structured tool interfaces with stateful execution.** Actions are issued via compact JSON/code primitives; browsing/search helpers return structured snippets, enabling batching, retries, and reproducibility (Zheng et al., 2025; Li et al., 2025f;e).

- **Beyond SFT: preference/RL optimization.** Targeted tuning (DPO/GRPO/PPO) of the planner or full loop improves tool use, coordination, and end quality (Hu et al., 2025; Zheng et al., 2025; Li et al., 2025f; He et al., 2025; Li et al., 2025e).

5.3 RL FOR MULTI-AGENT COORDINATION

This section surveys RL formulations that learn two or more decision-making policies within one deep research system, optimizing from a system-level objective. Unlike the common "single controller + fixed tools" setup, the works here either (i) perform joint cooperative multi-agent reinforcement learning (MARL) with explicit cross-agent credit assignment, for example Multi-Agent Proximal Policy Optimization (MAPPO; Yu et al. (2022)) under Centralized Training with Decentralized Execution (CTDE) as in MMOA-RAG or critic-free, group-relative advantages (MHGPO), often with role-conditioned parameter sharing and decentralized execution at test time; or (ii) pursue coordinate-wise component updates via globally aligned local rewards (Optimas), which is not joint MARL but is practical when full end-to-end multi-agent training is brittle or costly. For orientation, we reference MAPPO as a standard CTDE baseline; the surveyed methods address nonstationarity, long-horizon/sparse rewards, and compute limits through centralized critics or critic-free advantages, warm-starts and constrained action spaces, conservative/trust-region updates, and shared backbones.

MHGPO This paper studies a three-agent multi-hop search pipeline (Query Rewriter → Reranker → Answerer) and introduces MHGPO, a critic-free MARL method for LLM agents. It replaces value critics with group-relative advantages computed over heterogeneous rollout groupings—Independent Sampling, Fork-on-First, and Round-Robin—to balance exploration cost and signal quality. Final-answer rewards are propagated upstream; per-agent losses are then combined to update a single, role-conditioned LLM backbone, yielding joint end-to-end optimization without a critic. Empirically, it outperforms MAPPO on multi-hop QA, requires no SFT warm-up, and lowers compute/memory by removing critics. The trade-off is that all agents share one parameter set rather than maintaining distinct policy networks.

MMOA-RAG This work casts a realistic RAG pipeline as cooperative MARL and jointly trains the Query Rewriter, Document Selector, and Answer Generator with MAPPO under a shared global reward (F1/EM). It uses CTDE with a centralized critic, SFT warm starts for each role, constrained

Table 8: Multi-agent training strategies for deep research systems.

Work	Trainable Roles	Joint end-to-end RL? (How)	Credit assignment / params (notes)
MHGPO (Chen et al., 2025a)	Query Rewriter; Reranker; Answerer	✓ Critic-free (group-relative)	Global terminal reward; shared LLM backbone; no critic
MMOA-RAG (Chen et al., 2025d)	Query Rewriter; Document Selector; Answer Generator (retriever fixed)	✓ MAPPO (CTDE)	Shared global reward; shared backbone; SFT warm-start
Optimas (Wu et al., 2025e)	Prompts / routers / model params / hyperparams (heterogeneous)	✗ Coordinate-wise (per-component)	Learned LRF per component aligned to global metric; optional RL within modules

action spaces for the selector (document ID tokens), and light format penalties to stabilize exploration. All three agents share one role-conditioned LLM backbone; the retriever is fixed. Across HotpotQA, 2WikiMultihopQA, and AmbigQA, joint training of the modules outperforms single-module tuning and shows promising out-of-distribution transfer. This is bona fide multi-agent RL across the trainable modules, though parameter sharing and a fixed retriever stop short of full end-to-end optimization of the entire pipeline.

Optimas Optimas does not perform joint multi-agent RL. Instead, it learns a local reward function (LRF) for each component (prompt, router, model, hyperparameters) and adapts these LRFs online so they stay aligned with the system's global metric; components are then improved coordinate-wise via trust-region-style updates (prompt/search/model selection, or RL when appropriate). This delivers consistent gains across compound systems while avoiding simultaneous multi-agent credit assignment. In practice, Optimas is a strong alternative when full end-to-end multi-agent RL is too brittle or costly: it can include RL within selected modules yet sidesteps the complexity of joint training.

5.4 DISCUSSION

The surveyed frameworks and papers point to a converging recipe for practical deep research systems. First, separating planning from execution keeps the planner's state clean while specialization and routing increase throughput and depth; this holds across simple single-loop designs (Aomni), explicit planner–coordinator splits (DeerFlow), configurable graphs (LangChain), and orchestrator-led hierarchies (MiroFlow). Second, structured tool interfaces and narrow, well-typed actions (search, browse, code; MCP) reduce failure modes and make recovery, caching, and retries feasible. Third, human oversight and observability are not optional: review points, trace capture, and replay make systems auditable. Finally, stronger RL-trained planners slot naturally into these stacks, improving global behavior without requiring full joint training of all components.

We highlight three open questions for this area: (i) Learning to coordinate. How do we train coordination itself, including credit assignment across planner, coordinator, and executors, learned communication protocols, and role discovery, while preserving stability? (ii) Adaptive topology and scheduling. When should a system expand or contract its team, route to new specialists, or change its plan given budget, latency, and risk constraints, and can these decisions be learned rather than scripted? (iii) Standards for portability and replay. What common schemas for actions, traces, judges, and tool results will enable reliable replay under a non-deterministic web and allow results to transfer across stacks with minimal glue?

6 EVALUATIONS

Reliable evaluation is central to measuring progress in deep research systems but remains difficult because these agents operate as multi-step, tool-using workflows with both objective and open-ended outputs. In this chapter, we map the evaluation space and focus on what current practice actually

measures and how. We group the landscape into three families: question answering (QA) and vision question answering (VQA), which stress final answer accuracy under retrieval and browsing; long form synthesis, which assesses the quality of extended reports; and domain grounded agent benchmarks, which test end-to-end task execution with tools. Although evaluation is not itself an RL objective, it is essential for validating claims about training regimes, reward designs, and architectures, and for setting shared standards.

The first part of this section outlines key QA/VQA benchmarks developed to evaluate the reasoning and information retrieval capabilities of agentic systems. These benchmarks include traditional single- and multi-hop QA/VQA datasets to more advanced, LLM-driven dynamic web browsing settings, where the model must plan and adapt its search in real time. Together, they provide a comprehensive and diverse evaluation suite, enabling systematic assessment of how effectively LLM-based agents can search for relevant information, retrieve and synthesize evidence from multiple sources, reason across disparate facts, and generate accurate, contextually grounded answers for complex information-seeking scenarios.

Another key component of evaluating deep research systems involves assessing their ability to generate high-quality long-form texts, including summaries, explanations, reports, and academic-style outputs. Unlike short-form tasks with single correct answers, long-form generation is inherently open-ended and must be judged across multiple dimensions such as coherence, factuality, relevance, structure, and style. This makes evaluation particularly challenging. While human evaluation remains the gold standard, recent years have seen the development of automated metrics and domain-specific benchmarks that better capture these subjective and multi-faceted qualities. The second part of this section will survey representative benchmarks and evaluation protocols for long-form text generation.

Beyond task-specific evaluations like complex QA or long-form generation, another important strand of benchmarking targets domain-specific agents. These benchmarks aim to measure how well AI agents perform in realistic workflows, specialized domains, and applied settings, offering a more faithful assessment of their practical utility. In this section, we highlight representative benchmarks in this emerging space, which are critical for evaluating end-to-end capabilities of deep research agents under realistic operating conditions.

6.1 QA AND VQA BENCHMARKS

Text-based QA benchmarks. Earlier multi-hop QA benchmarks are generally in the static corpus setting where retrieval is performed over a fixed, pre-defined corpus. For example, HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020) require reasoning over multiple supporting documents from Wikipedia to answer; Natural Questions (Kwiatkowski et al., 2019) is derived from real Google queries and requires both short and long answer extraction; MuSiQue (Trivedi et al., 2022) is similar to HotpotQA and 2WikiMultiHopQA, but it also tests multi-hop reasoning robustness by adding distractor paragraphs to the context. FEVER (Thorne et al., 2018) and QASC (Khot et al., 2019) both emphasize fact checking and verification. The former consists of claims and evidence from Wikipedia, while the latter are from multiple text corpora. These datasets form the foundation for evaluating the core reasoning capabilities of LLMs in a more stable, noise-controlled setting. Later QA benchmarks began simulating a web-like environment by integrating retrieval settings that mimic search engine use. Representative benchmarks like Bamboogle (Press et al., 2022) and FRAMES (Krishna et al., 2025) demonstrate increased question complexity through retrieval over a static corpus to emulate real-world search behaviors. With the emergence of deep research systems capable of navigating dynamic and noisy web content, these QA benchmarks no longer provide sufficient challenge for meaningful evaluation. Recent QA benchmarks have shifted toward open-web evaluation. BrowseComp (Wei et al., 2025) and its Chinese counterpart BrowseComp-ZH (Zhou et al., 2025a) measure the ability of AI agents to locate hard-to-find information in live web environments. InfoDeepSeek (Xi et al., 2025b) include questions with false premise, testing whether a system can detect and handle misleading or unanswerable queries rather than producing hallucinated answers. Webwalker (Wu et al., 2025b) measures an agent's ability to perform multi-step information gathering by traversing real websites, testing both search-query formulation and page navigation skills. While these benchmarks requires deep search of information, WideSeach (Wong et al., 2025) target to evaluates AI agents on large-scale, exhaustive information gathering tasks that are not requiring deep reasoning, but rather breadth, consistency and fidelity.

Table 9: Benchmarks surveyed in this chapter and their key features.

Benchmark	Key features
QA/VQA Benchmarks	
HotpotQA	Multi-step reasoning over multiple Wikipedia documents.
2WikiMultiHopQA	Multi-hop questions across two Wikipedia articles.
Natural Questions (NQ)	Real Google queries; short and long answers.
MuSiQue	Multi-hop with distractor paragraphs to test robustness.
FEVER	Claim verification with Wikipedia evidence.
QASC	Science questions; multi-sentence composition across corpora.
Bamboogle	Search like retrieval over a static corpus.
FRAMES	Higher complexity with fixed corpus retrieval.
BrowseComp	Open web browsing; hard to find information.
BrowseComp-ZH	Chinese counterpart to BrowseComp.
InfoDeepSeek	False premise questions; detect misleading or unanswerable queries.
Webwalker	Multi-step gathering on real sites; query formulation and navigation.
WideSearch	Large-scale exhaustive gathering; breadth, consistency, fidelity.
MMSearch	Early multimodal search; small scale.
MMDocIR	Large multimodal suite (long docs, images, QA, evidence chains)
MRAMG-Bench	Large multimodal suite (docs, images, QA); text and visual answers.
M^2RAG	Multimodal RAG: captioning, QA, fact verification, image reranking.
MMDocRAG	Multimodal RAG: multimodal retrieval, reranking, answer generation.
MM-BrowseComp	Multimodal BrowseComp; image or video signals on webpages.
Omni Bench	Beyond vision and language; adds audio, video, structured data.
Long-Form Text Benchmarks	
HelloBench	Long text queries in five categories; HelloEval (LLM judge plus checklist).
ProxyQA	Human-authored proxy questions capturing key points.
WritingBench	Query dependent criteria generated per instance.
LongEval	LLM as judge vs human references (arXiv, Wikipedia, blogs).
DeepResearch Bench	Deep research report generation; 100 PhD level tasks; RACE and FACT metrics.
Domain-Grounded Benchmarks	
Xbench	Real-world productivity; recruitment and marketing workflows.
τ^2 Bench	Dual control telecom; user and agent both act with tools.
Finance Agent Benchmark	Financial research workflows.
FinGAIA	Finance benchmark in Chinese.
OdysseyBench	Office productivity across Word, Excel, PDF, email, calendar; long horizons and tool coordination.

Collectively, these datasets move beyond controlled, noise-free corpora to test the end-to-end skills needed for real-world, open-web information seeking, and complex problem solving.

Multimodal VQA benchmarks. While above-mentioned benchmarks concentrate on text-only QA pairs, benchmarks that combine visual understanding with web search/browsing are rapidly maturing. MMSearch (Jiang et al., 2025a) is one of the pioneer efforts designed to evaluate the multimodal search capability of the Large Multimodal Models (LMMs). MMSearch consists of 300 curated samples, which is really small. MMDocIR (Dong et al., 2025b) consists 1,685 questions for complex VQA on long multimodal document. It comes with not only answer for evaluating end-to-end results, but providing evidence (page and layout level) labels to evaluate intermediate deep research efficacy. MRAMG-Bench (Yu et al., 2025b) provides a more comprehensive multimodal benchmark. It includes 4,346 documents, 14,190 images, and 4,800 QA pairs from diverse domains, with tasks requiring both textual and visual answers. MMDocRAG (Dong et al., 2025c) enables evaluation on multimodal retrieval, reranking, and generation on long multimodal document. It provides 4,055 expert-annotated QA pairs with multi-page, cross-modal evidence chains, and answer in multimodal form. M^2RAG (Liu et al., 2025a) further advances this direction by introducing a multimodal retrieval-augmented generation benchmark that spans four open-domain tasks: image captioning, multimodal question answering, fact verification, and image reranking. MM-BrowseComp (Li et al., 2025d) is a multimodal version of BrowseComp. The questions are often prompts with images, and crucial information encountered during the search and reasoning process may also be embedded within images or videos on webpages.

Despite claiming to be multimodal benchmarks, these benchmarks involve only visual data. Omni-Bench (Li et al., 2024) extends beyond vision–language tasks by integrating more modalities, including audio, video, and structured data. It is designed to assess how well agents can coordinate across heterogeneous input sources and reasoning contexts, reflecting the kinds of multi-sensory information humans process in real-world scenarios.

6.2 Long-form Text Benchmarks

Many user queries to deep research systems demand not just factual accuracy but well-structured, long-form text that synthesizes information across sources. Previous long-form text benchmark can serve to evaluate the text generation capabilities of the systems. For example, HelloBench (Que et al., 2024) collects diverse user queries that require long text generation from different sources and divide them into five categories: open-ended QA, summarization, chat, text completion, and heuristic text generation. To evaluate the outputs, the authors developed HelloEval, a two-stage, human-aligned evaluation method, which combines LLM-as-a-judge and a checklist-based scheme which pairs each sample with 4–6 binary (yes/no) questions. ProxyQA (Tan et al., 2024) include manually created proxy-questions, which are short, targeted questions probing the key points that a high-quality answer should include, for each query. Both works rely on human annotators to create the evaluation questions. WritingBench (Wu et al., 2025f) takes a different approach by proposing a query-dependent evaluation framework that empowers LLMs to dynamically generate instance-specific assessment criteria. While these benchmarks use evaluation questions defined in advance, LongEval (Alkhalifa et al., 2024) uses LLM-as-a-judge to compare LLM-generated text with original human-written text from arxiv, wikipedia ang blog.

While these benchmarks provide solid foundations for evaluating long-form text, they are not specifically designed for deep research systems. The rise of deep research systems require benchmarks that test multi-step reasoning, iterative information gathering, synthesis across diverse sources, and the ability to generate accurate, well-supported insights rather than just surface-level answers. To bridge the gap, DeepResearch Bench (FutureSearch et al., 2025) is the first specialized benchmark for evaluating deep research systems, with a specific focus on report generation. It covers 100 PhD-level tasks, which requires the agents to plan research steps, gather and filter evidence from diverse web sources, and synthesize it into analyst-grade, citation-rich reports. This benchmark also offers two evaluation frameworks: RACE (Reference-based Adaptive Criteria-driven Evaluation), which uses LLM-as-a-judge to score dimensions such as comprehensiveness, depth, and instruction-following against high-quality reference reports; and FACT (Factual Abundance and Citation Trustworthiness), which measures the proportion of claims backed by correct citations and the overall accuracy of these citations.

6.3 Domain-Grounded Benchmarks

Beyond iterative reasoning to solve complex multi-hop questions and generating high-quality long text outputs, the current deep research systems are capable of a broader range of tasks. With the integration of multiple tools, the agents are now more reliable and capable of executing domain-specific and professional-level tasks. Therefore, many researchers start to benchmarks with domain-grounded and profession-aligned evaluations, moving to more faithfully measure the utility of AI systems in practice. Xbench (Chen et al., 2025b) is designed to assess the real-world productivity of AI agents, with special focus on recruitment and marketing. The evaluation tasks are shaped by professional headhunters and marketing practitioners, ensuring they reflect authentic business workflows. τ^2-Bench (Barres et al., 2025) is a benchmark for assessing conversational AI in dual-control environments in Telecom domain. In the dual-control environments, both the AI agent and the user have agency and can use tools to affect a shared world. This setup is reflective of real-world scenarios such as technical support, where the user isn't merely passive but also takes actions—like toggling airplane mode or restarting a device—based on the agent's guidance. The Finance Agent Benchmark (Bigeard et al., 2025) is a purpose-built evaluation framework designed to test LLM agents on real-world financial research challenges. FinGAIA (Zeng et al., 2025a) also targets financial domains but it focuses on Chinese. OdysseyBench (Wang et al., 2025c) targets at real-world office productivity—spanning applications like Word, Excel, PDF, Email, and Calendar. Rather than isolated tasks, it challenges agents to coordinate across multiple tools and contexts over extended time horizons.

6.4 DISCUSSION

The evaluation of deep research systems has rapidly matured, moving far beyond traditional metrics of accuracy and precision. This chapter charted a clear evolutionary path for benchmarks, beginning with static, multi-hop QA datasets like HotpotQA and progressing to dynamic, open-web challenges such as BrowseComp that test real-time information seeking. The frontier has further expanded to encompass multimodal reasoning (Omni-Bench), the synthesis of high-quality long-form text (DeepResearch Bench), and most recently, performance in realistic, domain-grounded workflows (OdysseyBench, Xbench). This progression reflects a fundamental shift in the research community: from evaluating isolated skills like retrieval and reasoning to assessing holistic, system-level capabilities that mirror the complexity of real-world professional tasks. Ultimately, these sophisticated evaluation frameworks are not just measurement tools; they are crucial drivers shaping the development of more capable, reliable, and practically useful AI agents.

Though numerous benchmarks for comprehensive capability evaluation have been established as per our investigation, there still remain many potential and challenging topics to explore:

- **Scalability and cost of high-fidelity evaluation.** As benchmarks become more grounded in real-world domains, cost and complexity rise sharply. Sourcing expert tasks and qualified evaluators is a major bottleneck. Future work should explore automated scenario generation and more reliable, calibrated LLM-as-a-judge systems to reduce dependence on human annotation without losing quality.

- **Evaluating Long-Term, Interactive, and Adaptive Agents:** Most current benchmarks test discrete single-turn tasks, while real research and professional work often involve long-term projects where agents must maintain context, learn from user feedback, and adapt their strategy over multiple sessions. We need frameworks that assess these longitudinal capabilities, including memory, continual learning, and collaborative interaction.

- **Assessing robustness, safety, and trustworthiness.** Beyond false premise checks, benchmarks should incorporate adversarial attacks, misinformation, and ethical dilemmas. Key questions include how agents handle conflicting sources, avoid harmful or biased content, and provide transparent reasoning. Standardized tests for these dimensions are essential for high stakes deployment.

- **Beyond vision–language to true cross-modal synthesis.** Existing multimodal evaluation has focused mainly on text and images. The next step is to test reasoning across audio, video, and structured databases, with tasks that require seamless integration of heterogeneous sources to solve a single complex problem.

7 CONCLUSION

This survey focuses on RL foundations for deep research systems, covering how agents are trained end-to-end to plan, tool use, reason, and synthesize through long-horizon, tool-using interactions. Our primary scope spans three areas: data synthesis and curation for rewardable tasks, RL methods that shape decision quality over full trajectories, and systems and frameworks that make agentic RL practical and reproducible at scale. As secondary foci, we review agent architecture and coordination patterns for deployment, and evaluations and benchmarks that measure both final answers and process quality. To our knowledge, this is the first survey centered on RL for deep research, offering a unified taxonomy, aligned axes for comparing systems, and consolidated tables for quick reference. Across chapters we distill practical guidance on task construction, reward and judge design, optimizer and regime choices, and system instrumentation. We also compile discussions and open questions that matter to the community. Our aim is to give researchers and builders a compact map of the space and actionable suggestions that accelerate progress toward capable, reliable, and comparable deep research agents.

References

Rabab Alkhalifa, Hsuvas Borkakoty, Romain Deveaud, Alaa El-Ebshihy, Luis Espinosa-Anke, Tobias Fink, Gabriela Gonzalez-Saez, Petra Galuščáková, Lorraine Goeuriot, David Iommi, et al. Longeval: longitudinal evaluation of model performance at clef 2024. In *European Conference on Information Retrieval*, pp. 60–66. Springer, 2024.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection, 2023. URL https://arxiv.org/abs/2310.11511.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Antoine Bigeard, Langston Nashold, Rayan Krishnan, and Shirley Wu. Finance agent benchmark: Benchmarking llms on real-world financial research tasks, 2025. URL https://arxiv.org/abs/2508.00828.

William Brown. Verifiers: Reinforcement learning with llms in verifiable environments. https://github.com/willccbb/verifiers, 2025.

ByteDance and contributors. Deerflow: Community-driven deep research framework. https://github.com/bytedance/deer-flow, 2025. Accessed 2025-08-13.

Guanzhong Chen, Shaoxiong Yang, Chao Li, Wei Liu, Jian Luan, and Zenglin Xu. Heterogeneous group-based reinforcement learning for llm-based multi-agent systems. *arXiv*, 2025a. URL https://arxiv.org/abs/2506.02718.

Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, et al. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*, 2025b.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for llms via reinforcement learning, 2025c. URL https://arxiv.org/abs/2503.19470.

Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, and Jiaxin Mao. Improving retrieval-augmented generation through multi-agent reinforcement learning. *arXiv*, 2025d. URL https://arxiv.org/abs/2501.15228.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025a.

Kuicai Dong, Yujing Chang, Xin Deik Goh, Dexun Li, Ruiming Tang, and Yong Liu. Mmdocir: Benchmarking multi-modal retrieval for long documents, 2025b. URL https://arxiv.org/abs/2501.08828.

Kuicai Dong, Yujing Chang, Shijie Huang, Yasheng Wang, Ruiming Tang, and Yong Liu. Benchmarking retrieval-augmented multimomal generation for document question answering, 2025c. URL https://arxiv.org/abs/2505.16470.

Yuchen Fan, Kaiyan Zhang, Heng Zhou, Yuxin Zuo, Yanxu Chen, Yu Fu, Xinwei Long, Xuekai Zhu, Che Jiang, Yuchen Zhang, Li Kang, Gang Chen, Cheng Huang, Zhizhou He, Bingning Wang, Lei Bai, Ning Ding, and Bowen Zhou. Ssrl: Self-search reinforcement learning, 2025. URL https://arxiv.org/abs/2508.10874.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. Areal: A large-scale asynchronous reinforcement learning system for language reasoning, 2025. URL https://arxiv.org/abs/2505.24298.

FutureSearch, :, Nikos I. Bosse, Jon Evans, Robert G. Gambee, Daniel Hnyk, Peter Mühlbacher, Lawrence Phillips, Dan Schwarz, and Jack Wildman. Deep research bench: Evaluating ai web research agents, 2025. URL https://arxiv.org/abs/2506.06287.

Apurva Gandhi and Graham Neubig. Go-browse: Training web agents with structured exploration. *arXiv preprint arXiv:2506.03533*, 2025.

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

Xinyu Geng, Peng Xia, Zhen Zhang, Xinyu Wang, Qiuchen Wang, Ruixue Ding, Chenxi Wang, Jialong Wu, Yida Zhao, Kuan Li, et al. Webwatcher: Breaking new frontiers of vision-language deep research agent. *arXiv preprint arXiv:2508.05748*, 2025.

Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D Manning. Synthetic data generation & multi-step rl for reasoning & tool use. *arXiv preprint arXiv:2504.04736*, 2025.

Google. Gemini deep research — your personal research assistant. https://gemini.google/overview/deep-research/, 2025. Accessed: 2025-08-18.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yichen He, Guanhua Huang, Peiyuan Feng, Yuan Lin, Yuchen Zhang, Hang Li, and Weinan E. PaSa: An LLM agent for comprehensive academic paper search. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11663–11679, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.572. URL https://aclanthology.org/2025.acl-long.572/.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. URL https://www.aclweb.org/anthology/2020.coling-main.580.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv*, 2025. URL https://arxiv.org/abs/2505.23885.

Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, Jianye Hao, et al. Deep research agents: A systematic examination and roadmap. *arXiv preprint arXiv:2506.18096*, 2025a.

Ziyang Huang, Xiaowei Yuan, Yiming Ju, Jun Zhao, and Kang Liu. Reinforced internal-external knowledge synergistic reasoning for efficient adaptive search agent, 2025b. URL https://arxiv.org/abs/2505.07596.

Abhinav Java, Srivathsan Koundinyan, Nagarajan Natarajan, and Amit Sharma. Frugalrag: Learning to retrieve and reason for multi-hop qa. *arXiv preprint arXiv:2507.07634*, 2025.

Yuelyu Ji, Rui Meng, Zhuochun Li, and Daqing He. Curriculum guided reinforcement learning for efficient multi hop retrieval augmented generation. *arXiv preprint arXiv:2505.17391*, 2025.

Dongzhi Jiang, Renrui Zhang, Ziyu Guo, Yanmin Wu, jiayi lei, Pengshuo Qiu, Pan Lu, Zehui Chen, Guanglu Song, Peng Gao, Yu Liu, Chunyuan Li, and Hongsheng Li. MMSearch: Unveiling the potential of large models as multi-modal search engines. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=J2Jyp1SZ0n.

Pengcheng Jiang, Xueqiang Xu, Jiacheng Lin, Jinfeng Xiao, Zifeng Wang, Jimeng Sun, and Ji-awei Han. s3: You don't need that much data to train a search agent via rl. *arXiv preprint arXiv:2505.14146*, 2025b.

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

Jiajie Jin et al. Decoupled planning and execution: A hierarchical reasoning framework for deep search. *arXiv*, 2025b. URL `https://arxiv.org/abs/2507.02652`.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Alexander Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *AAAI*, 2019.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *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)*, pp. 4745–4759, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.243. URL `https://aclanthology.org/2025.naacl-long.243/`.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

LangChain and contributors. Open deep research (langchain). `https://github.com/langchain-ai/open_deep_research`, 2025. Accessed 2025-08-13.

Fangyu Lei, Jinxiang Meng, Yiming Huang, Tinghong Chen, Yun Zhang, Shizhu He, Jun Zhao, and Kang Liu. Reasoning-table: Exploring reinforcement learning for table reasoning, 2025a. URL `https://arxiv.org/abs/2506.01710`.

Xuanyu Lei, Chenliang Li, Yuning Wu, Kaiming Liu, Weizhou Shen, Peng Li, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. Writing-rl: Advancing long-form writing via adaptive curriculum reinforcement learning. *arXiv preprint arXiv:2506.05760*, 2025b.

Jian Li, Xiaoxi Li, Yan Zheng, Yizhang Jin, Shuo Wang, Jiafu Wu, Yabiao Wang, Chengjie Wang, and Xiaotong Yuan. A survey on ai search with large language models. *arXiv preprint arXiv:2506.xxxxx*, 2025a.

Jinzheng Li, Sibo Ju, Yanzhou Su, Hongguang Li, and Yiqing Shen. Enhancing llms' reasoning-intensive multimedia search capabilities through fine-tuning and reinforcement learning. *arXiv preprint arXiv:2505.18831*, 2025b.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025c.

Shilong Li, Xingyuan Bu, Wenjie Wang, Jiaheng Liu, Jun Dong, Haoyang He, Hao Lu, Haozhe Zhang, Chenchen Jing, Zhen Li, Chuanhao Li, Jiayi Tian, Chenchen Zhang, Tianhao Peng, Yancheng He, Jihao Gu, Yuanxing Zhang, Jian Yang, Ge Zhang, Wenhao Huang, Wangchunshu Zhou, Zhaoxiang Zhang, Ruize Ding, and Shilei Wen. Mm-browsecomp: A comprehensive benchmark for multimodal browsing agents, 2025d. URL `https://arxiv.org/abs/2508.13186`.

30

Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, Hongxuan Lu, Tianrui Qin, Chenghao Zhu, Yi Yao, Shuying Fan, Xiaowan Li, Tiannan Wang, Pai Liu, King Zhu, He Zhu, Dingfeng Shi, Piaohong Wang, Yeyi Guan, Xiangru Tang, Minghao Liu, Yuchen Eleanor Jiang, Jian Yang, Jiaheng Liu, Ge Zhang, and Wangchunshu Zhou. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic rl, 2025e. URL `https://arxiv.org/abs/2508.13167`.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *arXiv*, 2025f. URL `https://arxiv.org/abs/2504.21776`.

Yangning Li, Weizhi Zhang, Yuyao Yang, Wei-Chieh Huang, Yaozu Wu, Junyu Luo, Yuanchen Bei, Henry Peng Zou, Xiao Luo, Yusheng Zhao, et al. Towards agentic rag with deep reasoning: A survey of rag-reasoning systems in llms. *arXiv preprint arXiv:2507.09477*, 2025g.

Yizhi Li, Ge Zhang, Yinghao Ma, Ruibin Yuan, Kang Zhu, Hangyu Guo, Yiming Liang, Jiaheng Liu, Jian Yang, Siwei Wu, Xingwei Qu, Jinjie Shi, Xinyue Zhang, Zhenzhu Yang, Xiangzhou Wang, Zhaoxiang Zhang, Zachary Liu, Emmanouil Benetos, Wenhao Huang, and Chenghua Lin. Omnibench: Towards the future of universal omni-language models, 2024. URL `https://arxiv.org/abs/2409.15272`.

Zhenghao Liu, Xingsheng Zhu, Tianshuo Zhou, Xinyi Zhang, Xiaoyuan Yi, Yukun Yan, Ge Yu, and Maosong Sun. Benchmarking retrieval-augmented generation in multi-modal contexts, 2025a. URL `https://arxiv.org/abs/2502.17297v2`.

Ziyu Liu, Yuhang Zang, Yushan Zou, Zijian Liang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual agentic reinforcement fine-tuning. *arXiv preprint arXiv:2505.14246*, 2025b.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning, 2025. URL `https://arxiv.org/abs/2508.03680`.

Jianbiao Mei, Tao Hu, Daocheng Fu, Licheng Wen, Xuemeng Yang, Rong Wu, Pinlong Cai, Xinyu Cai, Xing Gao, Yu Yang, Chengjun Xie, Botian Shi, Yong Liu, and Yu Qiao. O^2-searcher: A searching-based agent model for open-domain open-ended question answering, 2025a. URL `https://arxiv.org/abs/2505.16582`.

Jianbiao Mei, Tao Hu, Daocheng Fu, Licheng Wen, Xuemeng Yang, Rong Wu, Pinlong Cai, Xing Gao, Yu Yang, Chengjun Xie, et al. O^2-searcher: A searching-based agent model for open-domain open-ended question answering. *arXiv preprint arXiv:2505.16582*, 2025b.

MiroMindAI and contributors. Miroflow: A consistent agent framework with reproducible performance. `https://github.com/MiroMindAI/MiroFlow`, 2025. Accessed 2025-08-13.

OpenAI. Introducing deep research. `https://openai.com/index/introducing-deep-research/`, February 2025. Accessed: 2025-08-18; initial release Feb 2, 2025; page later updated Jul 17, 2025.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022. URL `https://arxiv.org/abs/2203.02155`.

Perplexity Team. Introducing perplexity deep research. `https://www.perplexity.ai/hub/blog/introducing-perplexity-deep-research`, February 2025. Blog post; Accessed: 2025-08-18.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

Haoran Que, Feiyu Duan, Liqun He, Yutao Mou, Wangchunshu Zhou, Jiaheng Liu, Wenge Rong, Zekun Moore Wang, Jian Yang, Ge Zhang, Junran Peng, Zhaoxiang Zhang, Songyang Zhang, and Kai Chen. Hellobench: Evaluating long text generation capabilities of large language models, 2024. URL https://arxiv.org/abs/2409.16191.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

Baochang Ren, Shuofei Qiao, Wenhao Yu, Huajun Chen, and Ningyu Zhang. Knowrl: Exploring knowledgeable reinforcement learning for factuality, 2025. URL https://arxiv.org/abs/2506.19807.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297. ACM, March 2025. doi: 10.1145/3689031.3696075. URL http://dx.doi.org/10.1145/3689031.3696075.

Wentao Shi and Yiqing Shen. Reinforcement fine-tuning for reasoning towards multi-step multi-source search in large language models. *arXiv preprint arXiv:2506.08352*, 2025.

Wenxuan Shi, Haochen Tan, Chuqiao Kuang, Xiaoguang Li, Xiaozhe Ren, Chen Zhang, Hanting Chen, Yasheng Wang, Lifeng Shang, Fisher Yu, et al. Pangu deepdiver: Adaptive search intensity scaling via open-web reinforcement learning. *arXiv preprint arXiv:2505.24332*, 2025a.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Autonomous retrieval-augmented reasoning of llms. *arXiv preprint arXiv:2505.11277*, 2025b.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Autonomous retrieval-augmented reasoning of llms, 2025c. URL https://arxiv.org/abs/2505.11277.

Joykirat Singh, Raghav Magazine, Yash Pandya, and Akshay Nambi. Agentic reasoning and tool integration for llms via reinforcement learning. *arXiv preprint arXiv:2505.01441*, 2025.

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

Huatong Song, Jinhao Jiang, Wenqing Tian, Zhipeng Chen, Yuhuan Wu, Jiahao Zhao, Yingqian Min, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher++: Incentivizing the dynamic knowledge acquisition of llms via reinforcement learning, 2025b. URL https://arxiv.org/abs/2505.17005.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, and Yu Cheng. Openthinkimg: Learning to think with images via visual tool reinforcement learning, 2025. URL https://arxiv.org/abs/2505.08617.

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

Haochen Tan, Zhijiang Guo, Zhan Shi, Lu Xu, Zhili Liu, Yunlong Feng, Xiaoguang Li, Yasheng Wang, Lifeng Shang, Qun Liu, and Linqi Song. ProxyQA: An alternative framework for evaluating long-form text generation with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6806–6827, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.368. URL https://aclanthology.org/2024.acl-long.368/.

Sijun Tan, Michael Luo, Colin Cai, Tarun Venkat, Kyle Montgomery, Aaron Hao, Tianhao Wu, Arnav Balyan, Manan Roongta, Chenguang Wang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. rllm: A framework for post-training language agents. https://pretty-radio-b75.notion.site/rLLM-A-Framework-for-Post-Training-Language-Agents-21b81902c146819db63cd98a54ba5f31, 2025a. Notion Blog.

Zhiwen Tan, Jiaming Huang, Qintong Wu, Hongxuan Zhang, Chenyi Zhuang, and Jinjie Gu. Rag-r1 : Incentivize the search and reasoning capabilities of llms through multi-query parallelism, 2025b. URL https://arxiv.org/abs/2507.02962.

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

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, Kuan Li, Liangcai Su, Litu Ou, Liwen Zhang, Pengjun Xie, Rui Ye, Wenbiao Yin, Xinmiao Yu, Xinyu Wang, Xixi Wu, Xuanzhong Chen, Yida Zhao, Zhen Zhang, Zhengwei Tao, Zhongwang Zhang, Zile Qiao, Chenxi Wang, Donglei Yu, Gang Fu, Haiyang Shen, Jiayin Yang, Jun Lin, Junkai Zhang, Kui Zeng, Li Yang, Hailong Yin, Maojia Song, Ming Yan, Peng Xia, Qian Xiao, Rui Min, Ruixue Ding, Runnan Fang, Shaowei Chen, Shen Huang, Shihang Wang, Shihao Cai, Weizhou Shen, Xiaobin Wang, Xin Guan, Xinyu Geng, Yingcheng Shi, Yuning Wu, Zhuo Chen, Zijian Li, and Yong Jiang. Tongyi deepresearch technical report, 2025. URL https://arxiv.org/abs/2510.24701.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 809–819, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074/.

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

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

Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M. Ni, Linyi Yang, Ying Wen, and Weinan Zhang. Openr: An open source framework for advanced reasoning with large language models, 2024. URL https://arxiv.org/abs/2410.09671.

Qiuchen Wang, Ruixue Ding, Yu Zeng, Zehui Chen, Lin Chen, Shihang Wang, Pengjun Xie, Fei Huang, and Feng Zhao. Vrag-rl: Empower vision-perception-based rag for visually rich information understanding via iterative reasoning with reinforcement learning, 2025a. URL https://arxiv.org/abs/2505.22019.

Teng Wang, Hailei Gong, Changwang Zhang, and Jun Wang. Sage: Strategy-adaptive generation engine for query rewriting. *arXiv preprint arXiv:2506.19783*, 2025b.

Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows, 2025c. URL https://arxiv.org/abs/2508.09124.

Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, Zichen Liu, Haizhou Zhao, Dakai An, Lunxi Cao, Qiyang Cao, Wanxi Deng, Feilei Du, Yiliang Gu, Jiahe Li, Xiang Li, Mingjie Liu, Yijia Luo, Zihe Liu, Yadao Wang, Pei Wang, Tianyuan Wu, Yanan Wu, Yuheng Zhao, Shuaibing Zhao, Jin Yang, Siran Yang, Yingshui Tan, Huimin Yi, Yuchi Xu, Yujin Yuan, Xingyao Zhang, Lin

Qu, Wenbo Su, Wei Wang, Jiamang Wang, and Bo Zheng. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library, 2025d. URL https://arxiv.org/abs/2506.06122.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*, 2025e.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024. URL https://arxiv.org/abs/2411.04368.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. Widesearch: Benchmarking agentic broad info-seeking, 2025. URL https://arxiv.org/abs/2508.07999.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025a.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Deyu Zhou, Pengjun Xie, and Fei Huang. Webwalker: Benchmarking llms in web traversal, 2025b. URL https://arxiv.org/abs/2501.07572.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. Mmsearch-r1: Incentivizing lmms to search. *arXiv preprint arXiv:2506.20670*, 2025c.

Mingyuan Wu, Jingcheng Yang, Jize Jiang, Meitang Li, Kaizhuo Yan, Hanchao Yu, Minjia Zhang, Chengxiang Zhai, and Klara Nahrstedt. Vtool-r1: Vlms learn to think with images via reinforcement learning on multimodal tool use. *arXiv preprint arXiv:2505.19255*, 2025d.

Shirley Wu, Parth Sarthi, Shiyu Zhao, Aaron Lee, Adrian Mladenic Grobelnik, Herumb Shandilya, Nurendra Choudhary, Eddie Huang, Karthik Subbian, Linjun Zhang, Diyi Yang, James Zou, and Jure Leskovec. Optimas: Optimizing compound ai systems with globally aligned local rewards. *arXiv*, 2025e. URL https://arxiv.org/abs/2507.03041.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, et al. Writingbench: A comprehensive benchmark for generative writing. *arXiv preprint arXiv:2503.05244*, 2025f.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. A survey of llm-based deep search agents: Paradigm, optimization, evaluation, and challenges. *arXiv preprint arXiv:2508.05668*, 2025a.

Yunjia Xi, Jianghao Lin, Menghui Zhu, Yongzhao Xiao, Zhuoying Ou, Jiaqi Liu, Tong Wan, Bo Chen, Weiwen Liu, Yasheng Wang, et al. Infodeepseek: Benchmarking agentic information seeking for retrieval-augmented generation. *arXiv preprint arXiv:2505.15872*, 2025b.

Ziyi Xia, Kun Luo, Hongjin Qian, and Zheng Liu. Open data synthesis for deep research, 2025. URL https://arxiv.org/abs/2509.00375.

Chengxing Xie, Zilin Zhu, Haoran Wang, Yao Wei, and Zhenyu Hou. Agent-oriented design: An asynchronous and decoupled framework for agentic rl. https://www.notion.so/Agent-Oriented-Design-An-Asynchronous-and-Decoupled-Framework-for-Agentic-RL-2278e692d081802cbdd5d37cef76a547, 2025. Blog post on Notion.

Renjun Xu and Jingwen Peng. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*, 2025.

Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zujie Liang, and Yongfeng Zhang. A-mem: Agentic memory for llm agents, 2025. URL `https://arxiv.org/abs/2502.12110`.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

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

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022 Track on Datasets and Benchmarks)*. Neural Information Processing Systems Foundation, 2022.

Chengyue Yu, Siyuan Lu, Chenyi Zhuang, Dong Wang, Qintong Wu, Zongyue Li, Runsheng Gan, Chunfeng Wang, Siqi Hou, Gaochi Huang, Wenlong Yan, Lifeng Hong, Aohui Xue, Yanfeng Wang, Jinjie Gu, David Tsai, and Tao Lin. Aworld: Orchestrating the training recipe for agentic ai, 2025a. URL `https://arxiv.org/abs/2508.20404`.

Qinhan Yu, Zhiyou Xiao, Binghui Li, Zhengren Wang, Chong Chen, and Wentao Zhang. Mramg-bench: A beyondtext benchmark for multimodal retrieval-augmented multimodal generation. *arXiv preprint arXiv:2502.04176*, 2025b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025c.

Lingfeng Zeng, Fangqi Lou, Zixuan Wang, Jiajie Xu, Jinyi Niu, Mengping Li, Yifan Dong, Qi Qi, Wei Zhang, Ziwei Yang, Jun Han, Ruilun Feng, Ruiqi Hu, Lejie Zhang, Zhengbo Feng, Yicheng Ren, Xin Guo, Zhaowei Liu, Dongpo Cheng, Weige Cai, and Liwen Zhang. Fingaia: A chinese benchmark for ai agents in real-world financial domain, 2025a. URL `https://arxiv.org/abs/2507.17186`.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in llm agents via turn-level credit assignment, 2025b. URL `https://arxiv.org/abs/2505.11821`.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in llm agents via turn-level credit assignment. *arXiv preprint arXiv:2505.11821*, 2025c.

David Zhang and contributors. deep-research: An ai-powered research assistant. `https://github.com/dzhng/deep-research`, 2025. Accessed 2025-08-13.

Dingchu Zhang, Yida Zhao, Jialong Wu, Baixuan Li, Wenbiao Yin, Liwen Zhang, Yong Jiang, Yufeng Li, Kewei Tu, Pengjun Xie, et al. Evolvesearch: An iterative self-evolving search agent. *arXiv preprint arXiv:2505.22501*, 2025a.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, Yifan Zhou, Yang Chen, Chen Zhang, Yutao Fan, Zihu Wang, Songtao Huang, Yue Liao, Hongru Wang, Mengyue Yang, Heng Ji, Michael Littman, Jun Wang, Shuicheng Yan, Philip Torr, and Lei Bai. The landscape of agentic reinforcement learning for llms: A survey, 2025b. URL `https://arxiv.org/abs/2509.02547`.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Xiangyu Zhao. Process vs. outcome reward: Which is better for agentic rag reinforcement learning, 2025c. URL `https://arxiv.org/abs/2505.14069`.

Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. Deep research: A survey of autonomous research agents. *arXiv preprint arXiv:2508.12752*, 2025d.

Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, and Jie Tang. Pre-frag: Preference-driven multi-source retrieval augmented generation, 2025a. URL `https://arxiv.org/abs/2411.00689`.

Qingfei Zhao, Ruobing Wang, Dingling Xu, Daren Zha, and Limin Liu. R-search: Empowering llm reasoning with search via multi-reward reinforcement learning. *arXiv preprint arXiv:2506.04185*, 2025b.

Qingfei Zhao, Ruobing Wang, Dingling Xu, Daren Zha, and Limin Liu. R-search: Empowering llm reasoning with search via multi-reward reinforcement learning, 2025c. URL `https://arxiv.org/abs/2506.04185`.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. Sglang: Efficient execution of structured language model programs, 2024. URL `https://arxiv.org/abs/2312.07104`.

Y. Zheng et al. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv*, 2025. URL `https://arxiv.org/abs/2504.03160`.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, et al. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*, 2025a.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*, 2025b.