A Survey of Deep Research Agents: From Search to Grounded Synthesis

Abstract

Deep research agents use language models to investigate questions over the web, private files and databases. They choose searches from earlier findings, retain evidence across long investigations, and produce short answers, collections or reports. We survey these agents by following evidence from the action that finds it to the claim it supports. This view connects search and planning, evidence management, grounded synthesis, and the training and evaluation of the investigation as a whole. Stopping improves when agents check the evidence against the question's requirements and track whether searches still find new material. Compact notes preserve access to omitted details when they retain a way to reopen their sources. Verification can improve the precision of an answer set while rejecting correct members whose evidence was never retrieved. Report revision often adds requested content while damaging earlier coverage and citation support. Training has a related dependence on evidence: when the corpus lacks the documents a question needs, an answer reward credits guesses, and repairing the corpus improves every reward design tested in a controlled comparison. These findings call for evaluations that measure outcomes, support, process and cost separately, and for methods that preserve a finding's meaning through compression, delegation and revision.

1 Introduction

Trained on vast amounts of text, large language models can answer questions, summarize documents and write code from instructions or a few examples [1, 2]. Hundreds of millions of people now use assistants built on them each week, often to look up information [3]. However, two limitations get in the way: (i) a model's knowledge is fixed when training ends, so it knows nothing of what happened afterwards; and (ii) it can state false claims confidently, with no source a reader could check. Retrieval-augmented generation (RAG) addresses both problems by looking information up [4]. Given a question, a RAG system retrieves relevant documents from an external collection, such as the web or a company's files, and generates its answer from them. The collection can be updated without retraining the model, and the answer can cite the documents it came from. In its standard form, RAG searches once with the user's question and answers from the top results.

One search works only when the question names what to look for. Suppose a local journalist asks, "What did metro Line A cost to build?" The top results are pages about Line A, and one of them likely states its cost. Now the journalist asks, "What did the metro line to the city's airport cost to build?" The top results are travel guides for getting to the airport, which name the line and give the fare but not the construction cost. Answering takes two searches: the first finds the line's name, and the second uses that name to find its cost. Systems that interleave search with reasoning choose each query from what earlier searches found [5, 6], and they answer such questions more accurately than systems that retrieve once [7].

Broader requests need many more searches, and answering them becomes a whole investigation. The journalist might go on to ask, "Why do some cities pay far more than others to build a metro line?" To answer, a system could compare the metro lines opened in European cities since

2015. No single page lists them, so it must first find them. It must then find each line's cost and length, often in separate sources, and compare the cost per kilometre. Systems now plan and coordinate these searches [8, 9], keep findings after the source text outgrows their context [10, 11], and use drafts to guide further research [12, 13]. We call this family *deep research agents*: language-model agents that conduct extended investigations over external sources, choosing what to search for or read, retaining the evidence, and synthesizing an answer, collection or report from it [14, 15]. Several commercial assistants now include such agents [14]. Deep research extends this interleaving of search and reasoning to longer inquiries and broader sources. It also specializes the freedom of general tool-using agents toward investigation, using browsing, code and delegated subtasks to gather, analyze and present evidence [14, 16].

Some difficulties carry over from RAG and grow with the investigation. (i) *The sources must support the answer, not merely match the topic.* A page reporting a metro line's approved budget may be relevant to its cost without establishing the final cost. In long reports, agents attach citations to claims their pages do not support [17]; later revision can damage support that was previously sound [18]. (ii) *The useful evidence must fit the context.* A long investigation reads more than the model can hold at once [10]. Keeping every page buries useful findings, but a summary can drop whether a cost was a budget or a final figure, or which year's prices it uses. Compression therefore changes what later searches and comparisons can establish [19]. These inherited problems connect evidence selection and memory to the output that eventually uses them.

Extended, adaptive search adds two further difficulties. (iii) *The agent must discover what remains to be found.* A fixed RAG pipeline has a predetermined end; an investigation must decide whether to continue. Agents search long after finding an answer, yet also stop with missing members or wrong attributes in a collection [20–22]. (iv) *Feedback does not identify which decisions helped.* A final-answer score covers many actions and cannot distinguish a successful investigation from a guess or a leaked answer key [20, 23]. We organize the survey around these connected difficulties. Search determines what the agent can observe (§3); evidence management determines what remains available (§4); and synthesis turns those findings into supported outputs (§5). Learning (§6) and evaluation (§7) must then distinguish improvements in the investigation from improvements in the answer alone.

Earlier surveys divide deep research by component or capability, including planning, question development, web exploration and report generation [27]; query planning, information acquisition, memory and answers [16]; or static and dynamic workflows run by one agent or several [14]. Others emphasize capabilities and applications [28], engineering layers [29], search-agent optimization [30], or reinforcement-learning pipelines [31]. These accounts describe the parts, but separating memory from report generation obscures how an early choice constrains a later one. A note that drops a cost's price base prevents a fair comparison, however capable the writer. We instead follow what each stage keeps and passes on (Figure 1), using a common investigation loop to connect decisions that component-based accounts treat separately. This yields three contributions:

- A formulation that separates observations, retained evidence and output claims (§2.2), so a failed search can be distinguished from a lost finding or an unsupported inference.
- A comparison of methods across those stages, showing when better search, recoverable memory and verification repair different failures, and when they trade completeness for support.
- A synthesis of training and evaluation evidence that explains why answer rewards can credit guesses and why outcomes, support, process and cost need separate measurement. The remaining problems concern evidence sufficiency, transfer and evaluation of whole investigations (§8).

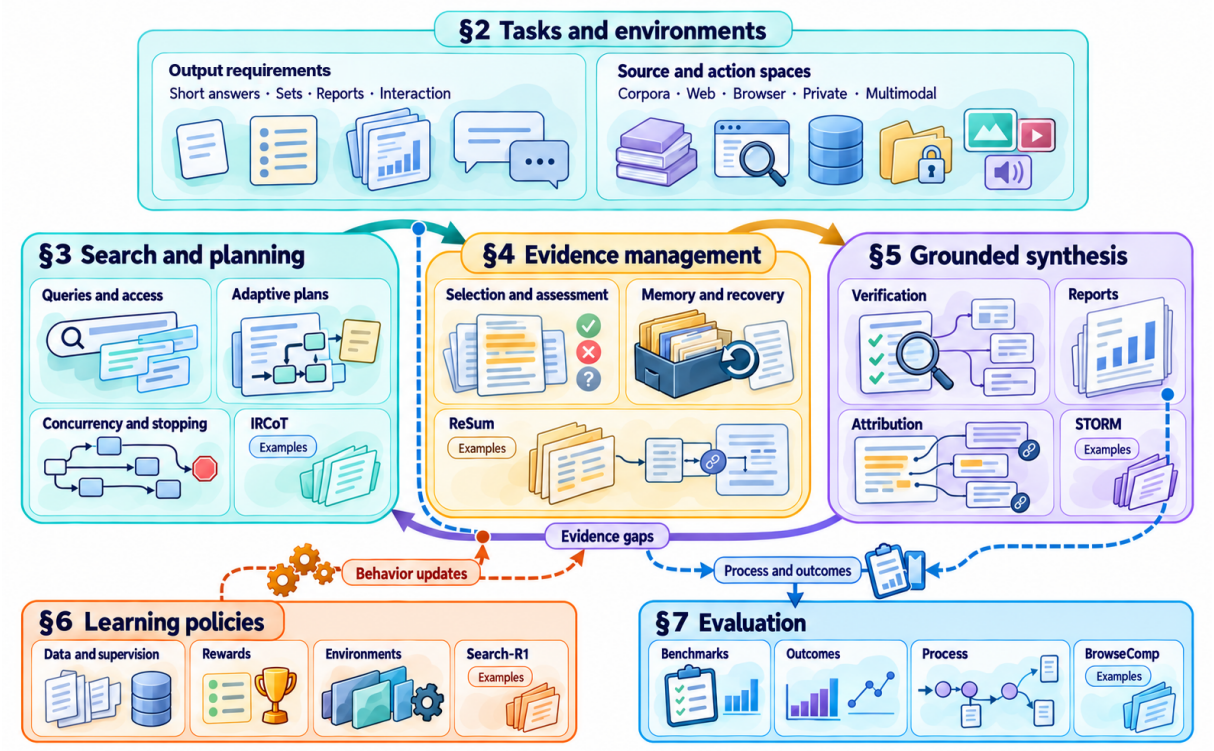


Figure 1: Search, evidence management and synthesis connect an agent's actions to its supported output (§3–5). Foundations establish the task, sources and investigation loop (§2); learning shapes the decisions (§6); evaluation measures their results (§7). Gaps found while writing can reopen the search [7, 10, 24–26].

2 Foundations

Retrieval-augmented generation gives a language model external documents from which to answer a request. Deep research builds on that goal by letting the model investigate before settling on an output. We develop the foundations in three layers: the RAG formulation (§2.1), the investigation loop that adapts it to extended research (§2.2), and the field's evolution into the design choices compared in the body (§2.3).

2.1 Retrieval-augmented generation

A retriever scores documents in the collection $\mathcal{D}$ against the request q with a matching function s and keeps the k best as evidence E ((1a)). A generator with parameters θ then produces an output y from the request and that evidence ((1b)):

$$E = \text{top-}k_{d\in\mathcal{D}}\, s(q, d), \tag{1a}$$

$$y \sim p_\theta(y \mid q, E). \tag{1b}$$

The collection can be updated without retraining the generator. The generator reads the selected documents and answers from them, so the evidence is fixed before generation begins [4]. An investigation changes this last condition: a finding or a draft can send the system back to retrieve again.

2.2 Deep research and the investigation loop

An investigation retains the request q and output y of RAG, but gathers and updates evidence over successive actions. Earlier findings supply names and relations absent from the original

request. Generating follow-up questions from intermediate answers helps with these dependencies (Self-Ask [5]); retrieving with successive reasoning sentences finds more supporting passages and improves the answer (IRCoT [7]). Longer inquiries extend this feedback loop to more sources and more demanding outputs.

Requests and requirements An investigation's *requirements* are what its output must establish. Four task families make different demands. (i) *Short answers and collections.* A short answer may require combining several sources [32]; a collection must also discover members and verify their attributes [22, 33]. For the journalist's question, finding which metro lines qualify and establishing comparable costs are separate requirements. (ii) *Judgments over conflicting evidence.* Sources may disagree because their scopes differ or because one is wrong, so the agent must retain source-specific findings until they can be compared [34, 35]. (iii) *Reports.* An explanation of costs per kilometre needs coverage, analysis and support across many claims [36, 37]. (iv) *Interactive inquiries.* Clarification or a later user turn can change the requirements, while earlier findings remain available to reuse [38, 39]. These families can combine in one investigation: a report may contain a collection that the user later narrows.

Sources and actions The source environment determines which actions the loop can take and what each action returns. A fixed collection supports repeatable searches over a snapshot, either through an index or by navigating raw files (SimpleWikiSearch [40]; LENS [41]). For time-sensitive questions, the snapshot date determines the answer: a metro line that opened after the snapshot does not appear in it. On the live web, a search service returns ranked snippets, while a browser opens pages and navigates their contents [42–44]. A result snippet and a full audit report give the agent different opportunities to find the same cost. The action must therefore be understood together with what the environment reveals in response.

Private and multimodal sources extend these actions further. Enterprise research combines company files with public pages [36, 45], and a database query returns structured rows that can depend on a name found on the web [46]. Visual sources add region selection and image interpretation: a cost shown only as a bar in a chart may be absent from extracted text. An agent may need a visual finding to construct its next text query [47], or crop and compare page regions to read the evidence [48, 49]. Across these environments, the loop must preserve enough of each observation's source and scope to support later reasoning.

Actions, observations and retained evidence Before step t, the agent holds its *retained evidence* E_t, including findings, quotations and source addresses, often alongside a plan of open subquestions. A *policy* π chooses an *action* a_t from the request q and this retained material ((2a)). Actions include searching, opening a page, querying a database, asking the user and delegating a subtask. The environment returns an *observation* o_t, such as a result list, page or image region. An *update rule* u decides what to keep from it ((2b)):

$$a_t \sim \pi(\cdot \mid q, E_t), \tag{2a}$$

$$E_{t+1} = u(E_t, a_t, o_t). \tag{2b}$$

When the policy stops after T steps, the agent writes the output y from q and the final evidence E_T. Its sequence of actions and observations is the *trajectory* τ. Its *cost* $c(\tau)$ measures the searches, tokens, time or money spent. The investigation seeks a supported output that meets the request at an acceptable cost:

$$\max_{\pi,\, u} \; \mathbb{E}\big[\, Q(y, E_T) - \lambda\, c(\tau) \,\big]. \tag{3}$$

Here *quality* Q assesses whether y meets the requirements of q and whether E_T supports its claims; $\lambda \geq 0$ weights cost against quality. Benchmarks approximate these demands with answer

Table 1: Retaining uncertainty about a cost figure prompts the next search in the journalist's investigation.

t	Action a_t	Observation o_t	Kept in E_{t+1}
1	Search for the metro line to the city's airport	Travel guides that name the line	The line's name and the guide's address
2	Search for that line's cost	A news report giving one figure, not labeled as budget or final cost	The figure, marked as possibly the budget, and the report's address
3	Open the city's audit report	A table of the approved budget and the final cost	Both figures, their price base and the page's address

Output y after $T = 3$ steps: the final cost, set against the approved budget and citing the audit report's table.

references, evidence annotations and report *rubrics*, criteria specifying the content and analysis a report should contain (§7).

Table 1 gives the full trace for the journalist's question. The first search identifies the airport line. The next returns a cost without saying whether it is the approved budget or the final figure. Keeping that uncertainty and the source address prompts the agent to open the audit report, which distinguishes both figures and states their price base. Keeping only the number would leave the answer model unable to justify calling it the final cost. The update rule thus changes which action follows and which claim the final evidence can support.

The loop does not run one way from search to writing. Drafting the comparison between two metro lines can reveal that their costs use different price bases, which sends the agent back to search for the adjustment. Systems that organize their evidence into a draft or an outline before answering use this signal to find gaps and unused findings [13, 50]. A conflict between sources can trigger another search in the same way [34], and a missing premise can add subquestions to the plan [51] or rewrite the next query [52]. The decision to continue is therefore made inside the loop, from what E_t shows to be missing or contested (§3.4).

2.3 Evolution and design space

Early search agents alternated reasoning with retrieval or browsed through a text interface and quoted sources [5–7, 53]. Figure 2 shows how this starting point developed into four directions of work. Search expanded into long reasoning chains, parallel execution and delegated investigations [8, 9, 54]. Evidence management added summaries and folded contexts, then source recovery to restore details omitted from memory [10, 48, 55]. Synthesis grew from cited answers and outlined articles into draft-guided research and evidence-linked arguments [12, 24, 50, 56]. Learning moved from demonstrations to reward-trained search and joint training of cooperating agents [25, 53, 57, 58]. Each direction changes a different part of the same loop.

We organize the design space around these four parts of an investigation. The policy π selects actions and decides whether to continue (§3); the update rule u selects and retains their evidence (§4); synthesis uses that evidence to construct supported outputs (§5); and learning determines how these behaviors are obtained (§6). Evaluation measures their contributions to quality and cost separately (§7), making it possible to tell an improved search from an improved use of the same sources.

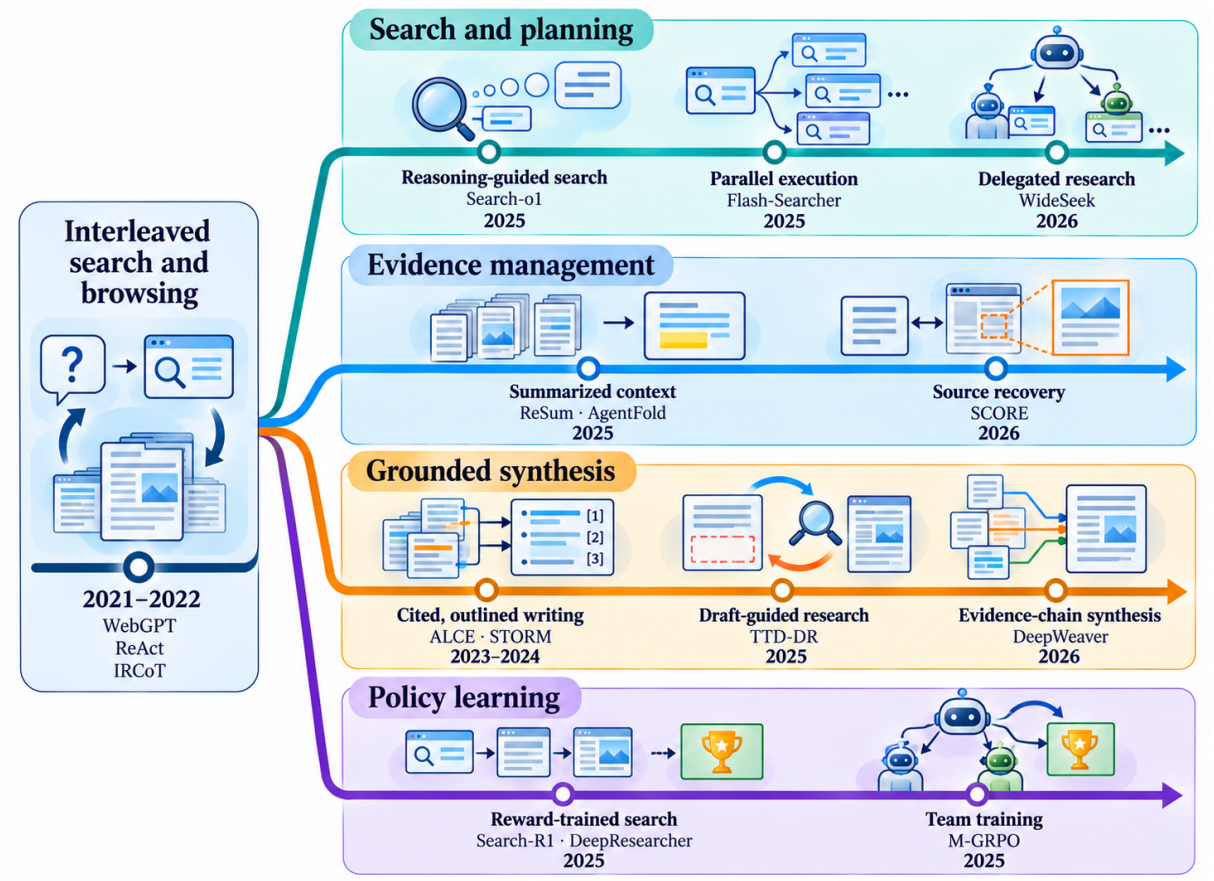


Figure 2: Early search-and-browse agents developed into four connected research directions: search and planning (§3), evidence management (§4), grounded synthesis (§5), and policy learning (§6). Each branch traces the capabilities that expanded the investigation loop [6–10, 12, 24, 25, 48, 50, 53–58].

3 Search and planning

Search and planning determine which evidence an agent gets to see. A search may supply a missing name, a page visit may test a candidate, and a finding may change which question comes next. Useful actions fill a gap; others repeat known facts or follow a wrong lead. We compare four decisions: which search or inspection action to take (§3.1); how to form and revise the plan (§3.2); how to divide independent work (§3.3); and when to stop (§3.4). Table 2 compares how systems combine these choices.

3.1 Search actions

A search returns previews, not evidence. The agent must still decide which pages to open, find the relevant passage inside each one and, for an image, choose how closely to look. These steps limit what the agent can find as much as its queries do. Treating them as actions of their own lets an agent reach evidence that searching alone would miss.

Search and inspection The search service controls how much its previews reveal (§2.2). Three search APIs yield similar answer accuracy, yet one exposes supporting text in its snippets for almost twice as many questions as the others [68]. In a long audit report, the relevant part can be a table hundreds of pages in. One approach separates an outer loop that decides what to investigate from an inner browser loop that clicks, scrolls and extracts the relevant content (NestBrowse [69]). Another navigates persistent views of a document collection instead

Table 2: Search policies differ in their actions, plan updates, division of work and stopping rules.

System	Actions	How the plan adapts	How work runs	When it stops
ReAct [6]	Search Wikipedia; look up text in a page	Each thought chooses the next action	One step at a time over the full history	A `finish` action
IRCoT [7]	Keyword search over paragraphs	The latest reasoning sentence becomes the next query	Alternates reasoning and retrieval, keeping every retrieved paragraph	An answer marker, or a cap on steps or paragraphs
PL-Search [59]	Search; retrieve documents; refine them	Ordered subquestions written before retrieval	One subquestion at a time, carrying refined evidence forward	The model answers once it judges the evidence enough
Dep-Search [60]	Dense retrieval with reranking; memory lookups	Graph of subquestions solved in dependency order	One step at a time; facts kept in a persistent memory	An answer, or a step budget
TreeSeeker [61]	Web search; page crawl; code	Deepens, explores, prunes or returns within a tree of goals	Goals whose inputs are ready run in parallel, each branch with its own memory	The evidence is judged sufficient, or a global step cap
DAGent [62]	Search; page access; recall of an earlier step's full trace	Evaluates completed steps, then grows or revises the graph	Independent steps run in batches and pass short summaries to the steps that need them	An answer step; up to three retries of a failed step
Flash-Searcher [8]	Web search; crawl and summarize pages	Dependency graph, revised periodically	Subtasks whose inputs are ready, and early probes, run in parallel over shared state	No subtasks remain
Hybrid DeepSearcher [63]	Web-search API; external summarizer	Reasoning chooses sequential or parallel queries	Several queries per turn, each returned with its summary	The model answers, or caps on turns and calls
WebSwarm [64]	Web search; page browsing	Each node probes the web, then searches itself or delegates	Children run in series or in parallel; results pass upward, and siblings share experience	The parent revises or ends the task; a step cap
SearchOS-V1 [65]	Search, open and find; tools for each role	Missing required fields set task priorities and dependencies	Each task starts as soon as an agent is free; agents share a record of tasks, evidence, coverage and failures	The coordinator decides; caps on iterations, searches and time
Enterprise DR [66]	Public search; connectors to enterprise data	Dependency graph from an outline; limited replanning, then fixed	Independent steps run once their inputs are ready, each seeing only the outputs it depends on	Evidence criteria set for each step
HALT [67]	The host agent's sentence retrieval	Keeps the host agent's plan; adds the claims each hop must establish	Host agent's loop unchanged; tracks which claims are supported	Every expected claim is supported, or the host's cap

of repeatedly submitting searches (AtlasNav [70]). The latter targets a substantial failure on BrowseComp-Plus, a fixed-corpus investigation benchmark: about three quarters of the previous best system's wrong answers involved evidence lost before it was fully located. These approaches make locating the relevant passage an explicit part of the action, so retrieving a document is followed by finding the evidence inside it.

Visual acquisition Visual inspection adds a choice of resolution and region. To read the audit report's cost chart, the agent must locate the page and then enlarge the labels. A page-level overview helps choose where to spend this effort: looking at thumbnails before fetching full-resolution pages improves accuracy (Doc-V* [71]). Within a page, cropping and zooming let an agent test a region and return to an earlier reasoning step when the crop is unhelpful (VISOR [72]). On the web, these operations can be combined with text, image and reverse-image search (Visual-Seeker [73]).

Inspection also depends on whether the question describes the target page directly. In one comparison, a visual retriever ranks directly described pages first 97.5% of the time, but only 2.5% when two intermediate pages are needed. Iterative agents recover much of this loss, ranking the target first 67.5% of the time with page images versus 37.5% with extracted text [74]. Following intermediate clues and reading the original image together make evidence reachable that direct matching misses.

3.2 Adaptive planning

A *plan* records the subquestions needed for an answer and their dependencies. Compare "What did metro Line A cost to build?" with "What did the metro line to the city's airport cost to build?" The first names the search target; the second needs its identity established first, unless the model already knows it. A collection adds subquestions only as new metro lines are discovered. Planning methods differ in whether these steps remain implicit in reasoning, are recorded and checked, or grow into alternative paths. User clarification can change the requirements on which any of these plans rests.

Interleaved reasoning and retrieval An implicit plan lets the model interpret each observation and choose its next action in free-form reasoning (ReAct [6]). This improves on acting without reasoning, but retrieval failures can erase the benefit: on HotpotQA, a multi-document question-answering benchmark, the approach falls slightly behind reasoning without search. Reasoning models extend the same mechanism over longer chains, summarizing search results into their reasoning (Search-o1 [54]) or drafting parts of a report along the way (WebThinker [75]). The original request can then anchor a search that starts to drift. On legal and regulatory documents, searching with the original question alongside its first rewrite counters drift, while deleting unhelpful passages prevents early stops caused by a full context [76]. Free-form reasoning supports flexible follow-ups, and retaining the original question keeps that flexibility directed at the user's task.

Plans and support checks Recording the plan makes a failed step easier to identify. Ordered subquestions tie searches to the requirement they serve and reduce drift (PL-Search [59]). A more local check states a goal before each search, tests whether the results meet it, and rewrites an unsuccessful query; this improves robustness to misleading results and small question changes (RE-Searcher [77]). Recording dependencies addresses repeated work instead: a later step can reuse a fact already held in memory (Dep-Search [60]). Each record catches a different failure: (i) a subquestion sequence counters drift; (ii) per-search goals expose failed retrieval; and (iii) dependencies reveal work that can be reused.

Alternative paths and growing graphs An early wrong turn becomes expensive when later searches depend on it. A search for the airport line might return both a metro line and a separate airport rail link. Committing to the rail link sends later cost searches toward the wrong project. Exploring competing hypotheses before committing raises success by 13 points over a single trajectory on a small BrowseComp subset, with fewer calls than majority voting over

complete trajectories (HypoSearch [78]). A search tree adds a way to return to earlier alternatives, recording each branch's findings and deciding whether to deepen, explore or prune it (TreeSeeker [61]). Dependency graphs instead record which results later steps need. Evaluating completed steps before growing such a graph improves accuracy and reduces calls relative to planning the full graph first and patching it (DAGent [62]). Even a single query benefits from conditional decomposition: splitting it only when retrieved passages fail to answer it beats iterative retrieval at matched model calls and lower cost (Hi-Q [79]). Across these designs, tested findings guide the next expansion more effectively than a plan fixed before retrieval.

Clarification and user steering Some planning errors require the user's help. Two cost figures may count different things, revealing that the journalist never specified whether trains and stations belong in the total. In search benchmarks with a simulated user, models ask useful clarifying questions once they notice the ambiguity, but often fail to notice it. Prompting them to watch for ambiguity raises average detection F1 from 45% to 65%; without that prompt, the strongest model answers only 43% of questions correctly end to end (DiscoBench [80]). User input can enter before or during research. An agent can consult user history and ask questions until it has a workable brief (G-STEER [81]), while an interactive interface lets the user revise retrieval directions and candidate sources before synthesis (FootprintRAG [82]). Clarification improves the target of the plan; steering improves the evidence collected for it.

3.3 Parallel and multi-agent search

The journalist's question contains both dependent and independent searches. The agent needs the names of the metro lines before looking up their costs, but the cost searches can run at the same time. We distinguish the dependency graph, the execution schedule, and the division of reasoning between a controller and workers (Figure 3). Concurrency changes when actions run; delegation also changes who reasons over their results and what returns to the coordinator.

Concurrent actions One controller can run independent searches concurrently while keeping the reasoning in one place. Executing a model-written query graph in parallel and returning tagged results uses about 70% fewer context tokens and half the latency of compared multi-agent frameworks (R-Search [83]). Revising a graph during execution and starting early probes before all prerequisites finish also reduces time relative to a baseline agent framework (Flash-Searcher [8]). Parallelism can also widen a search rather than only shorten it. Alternating broad query expansion and aggregation with sequential reasoning keeps improving as more turns or calls become available [63], and searching several visual entities in one round provides a similar benefit [85]. Independent attempts need different starting points to gain this breadth: choosing diverse first queries raises multi-hop accuracy by 5 to 7 points at matched computation [86]. Concurrent searches save time when their inputs are ready and improve discovery when they pursue different evidence.

Delegation and shared progress *Delegation* gives a subtask its own reasoning process and context, then returns selected findings to a coordinator. A worker for each city could find the metro lines it opened since 2015 and their costs. The hierarchy can grow with task breadth (WideSeek [9]), let each worker decide whether to delegate further (WebSwarm [64]), or separate parallel retrieval from aggregation (A-MapReduce [87]). Communication determines how much of the resulting work the coordinator must read. Keeping tool traces private and returning concise results makes one system three to five times faster than compared commercial deep research agents (InfoSeeker [84]). A shared record of tasks, evidence and failures instead helps workers avoid repeated searches (SearchOS-V1 [65]). Private contexts reduce what the coordinator reads; shared progress records reduce what the team repeats.

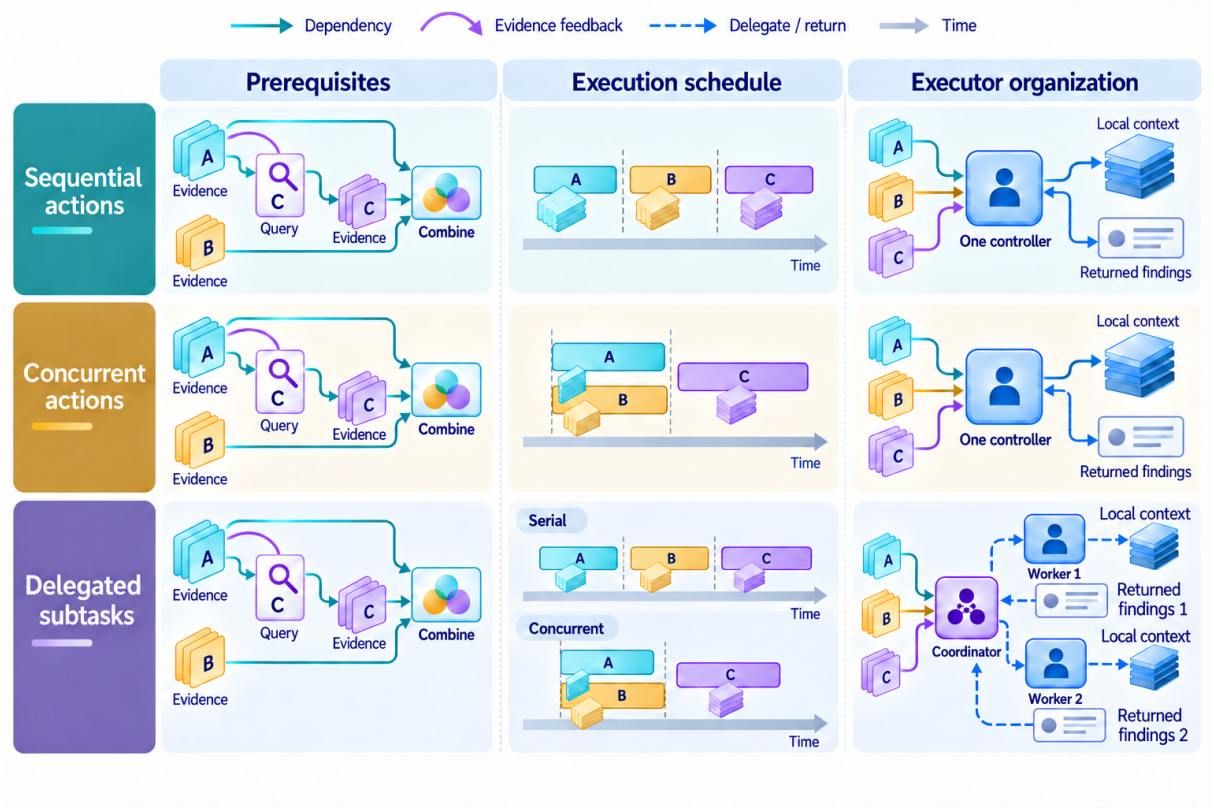


Figure 3: The same search dependencies permit different schedules and divisions of reasoning. Left, prerequisites; middle, sequential or concurrent execution; right, one controller or delegated workers [7, 8, 83, 84].

Choosing the subproblems accounts for much of delegation's benefit in a controlled comparison. With answer generation held fixed, separating delegation from execution raises exact match by 4.5 to 8.6 points over one shared search context. Scaling the delegating model adds about 11 points, versus about 2.6 for scaling the executor. A smaller executor trained on filtered demonstrations also matches the frontier executor with 37% fewer tokens [88]. For these tasks, model capacity is better spent deciding what workers should investigate than making every worker larger.

3.4 Search budgets and stopping

A stopping decision weighs the cost of another action against the chance of finding needed evidence, the trade-off in Eq. (3). Agents both over-search and stop too soon. On one open training set, successful trajectories average about 85 searches although the answer first appears around the ninth [20]. On questions an agent under-searched, requiring evidence for every hop doubles exact match [67]. We distinguish hard resource limits, checks against known requirements, and judgments about whether further searches will add useful evidence.

Cost budgets A hard cap bounds resource use, so budget methods try to spend each available action where it helps most. Giving an agent its remaining search and token allowances improves its allocation, though returns diminish as the search allowance grows (BCAS [89]). A controller can rank actions by expected value under the remaining budget [90], or divide retrieval between untested subqueries and ones that previously found useful material [91]. The amount of text returned also consumes the budget. Coarse-to-fine retrieval through a document hierarchy raises accuracy on BrowseComp-Plus from 44% to 67% (BEAR [92]), while jointly learning when to stop and how much detail to expose adds about 9 points over an answer reward alone (DeepControl

[93]). A useful budget policy allocates both actions and reading effort, rather than treating every search as equally costly and informative.

Task-relative completion When the required facts are known, the agent can check them directly. HALT turns a multi-hop question into claims and stops the host agent once retrieved sentences support each one [67]. It cuts loops with at most a 2-point loss in exact match in five of six settings, while the same check extends searches that stopped too early. Enterprise agents similarly write termination criteria for each outline step [66]; a record of confirmed facts, missing information and conflicts helps preserve these criteria during the search [94]. Completion depends on decisive support, rather than the volume read. One audit table can settle the journalist's cost question, just as a few visual clues can settle a misleading video's claim under a limited acquisition budget [95]. Checking the required claims lets an agent stop after a small amount of evidence when that evidence answers the question.

Stopping under uncertain judgments Collections make this check harder because some required members have not yet been discovered. When new searches return only metro lines already found, the search looks exhausted, but the collection may still be incomplete. Controllers therefore track both remaining gaps and the yield of new searches. Tracking requirement coverage, new documents, query variation and drift saves about 14 searches per question on BrowseComp-Plus while improving accuracy (RAAC [96]). A record of support, gaps and conflicts can likewise guide a choice between searching, rewriting, verifying and answering with fewer tokens (BELIEFRAG [97]). These signals do more than asking a judge whether the evidence looks sufficient. In a trained search policy, 27% of searches occur after the answer is reachable, yet a sufficiency judge removes only 3.7% of calls and costs some exact match [98]. Stopping improves when the controller checks what remains unanswered and whether its searches still add evidence.

4 Evidence management

Evidence management turns an observation o_t into retained evidence E_{t+1}. Keeping everything carries noise into later steps; keeping too little can discard a condition the answer needs. For example, a cost without its price base cannot support a fair comparison with another metro line. We distinguish selection, which decides what to admit and how to label it (§4.1), from context management, which keeps admitted evidence usable beyond the context window (§4.2). Table 3 compares the retained material, update timing and routes back to original sources.

4.1 Evidence selection

A passage that matches the question is not always worth keeping. It may repeat something the agent already knows, give a figure for a different project, or come from an unreliable source. The agent should therefore judge each passage by what it adds to the answer, not by how well it matches. That depends on three things: whether the passage answers something still open, what exactly it claims, and whether its source can be trusted.

Requirement-aware selection Selection should favor a passage that fills an unanswered requirement over another copy of a known fact. Three news reports repeating the approved budget leave the final cost unanswered. For example, LineageRAG records which requirement retrieved each passage and selects a set that covers different requirements [99]. It also gives the answer model verbatim spans; this improves exact match on MuSiQue by about 6 points without changing the retrieved passages. Region selection can also improve support without changing the

answer much: in raw documents, it greatly improves evidence recovery and citation while giving similar answer accuracy to an iterative agent [41]. Selection improves both the set's coverage and the inspectability of its support, even when the final answer score changes little.

Support, conflict and scope A selected passage must retain what it is about and whether it supports or conflicts with a claim. A budget differs from a final cost without contradicting it. Two different final costs may conflict, but only if they concern the same project and scope. Claim graphs make this distinction explicit by linking quoted spans to the claims they support or oppose. In EviGraph, a verifier assigns these links, another role proposes updates, and deterministic checks keep the graph consistent [100]. Separating the roles raises accuracy on BrowseComp-Plus from 2.7% to 26.9%, partly by preventing premature stopping; reinforcement learning raises it further to 35.9%. Admission checks can enforce scope before a claim is assessed. A clinical verifier admits a record only after matching the patient, occurrence and original source, and makes no false support claims when the required evidence is removed [101]. The equivalent cost check preserves which metro line or extension the figure covers and whether trains are included. Recording scope before support prevents unlike figures from being treated as comparable evidence.

Visual evidence For visual evidence, selection also allocates image detail. An agent can admit pages that answer open requirements, penalize redundant ones, condense resolved observations and reserve high resolution for regions needing closer inspection. On questions requiring at least four searches, this raises accuracy from 46% to 74% while cutting images sent to the model from about 93 to 15 per question (TAEC [102]). For the audit report, the chart needs full detail while its labels are being read; afterwards a value with its scope and source pointer can carry the finding forward. Selecting detail by the unresolved question reduces visual context without discarding the information the answer still needs.

Credibility and selection effects A page can match the question while giving an outdated or fabricated cost. Relevance alone gives the agent no reason to trust the figure. Giving a fact-checking model publisher backgrounds helps it weigh conflicting evidence, especially when the answer stage explicitly reasons about source reliability [103]. Plausible methodology can nevertheless overpower a clear defect: when sources are combined, a source with impossible statistics retains about 79% of the influence it has with valid statistics, even though models detect the fabrication in isolation [104]. Source assessment therefore has to affect synthesis, rather than remain a separate judgment that the writer ignores.

The agent's own selection also biases the evidence it sees. If highly ranked pages mostly report overruns, reading only those pages can inflate the estimated frequency of overruns. Logging each document's probability of being opened lets an auditor reweight the read pages toward the full candidate pool. For a public deep research agent, this reduces error against that pool's conclusion by 60% (CESS [105]). This correction addresses the agent's choice within the pool; assessing publishers addresses the credibility of the pool's contents. Both matter before agreement among selected pages is treated as evidence of consensus.

4.2 Context management

The raw history includes navigation text, repeated passages and abandoned reasoning alongside useful findings. Context-management methods differ in when they compress this history, what replaces it and whether an original source remains recoverable. Figure 4 illustrates recovery. A note may preserve a finding f, such as the final cost, while dropping a source detail d, such as its price base. A page address lets the agent recover that detail before comparing costs.

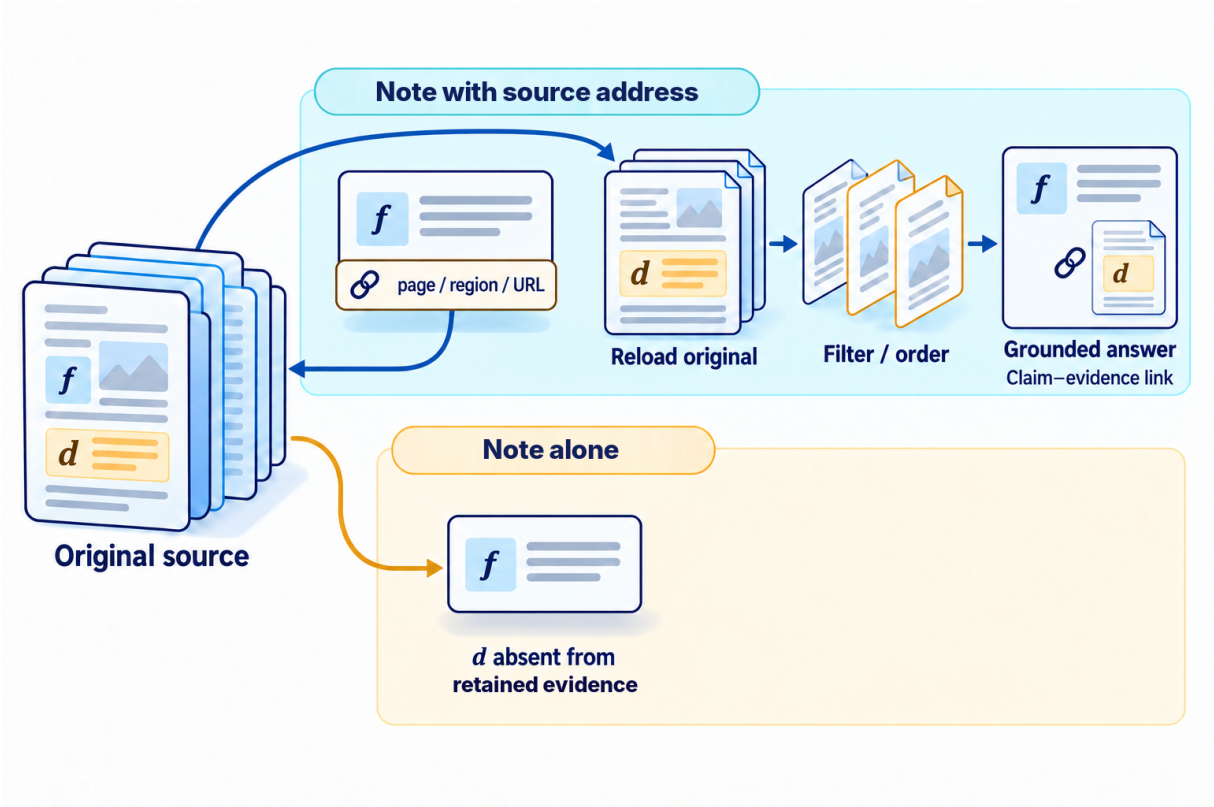


Figure 4: A source address lets a compact note recover omitted detail. The note keeps finding f; reopening the page or region restores detail d for the answer (top), while a note alone leaves it unavailable (bottom) [48].

Summarization and revision A summary replaces the accumulated history with the question and a compact account of progress. This lets an agent search beyond its context limit, and training on summarized trajectories improves the gain over adding summaries only at test time (ReSum [10]). Timing compression to the remaining context budget can even let a much smaller model with a shorter context outperform a larger one on BrowseComp-Plus (ContextBudget [106]). Updating the summary matters as much as producing it. Rewriting memory and the objective checklist when progress stalls or findings conflict improves accuracy by up to 11 points over passive compression (ARC [107]). If a later page identifies a recorded cost as a budget, the note must replace that assumption. A useful summary preserves current evidence rather than accumulating every earlier belief.

Isolation and folding Isolated contexts keep detailed work out of the main history before it needs compression. A tool or subtask returns findings and source identifiers while its intermediate deliberation stays private. On GAIA, a general-assistant benchmark, this raises accuracy from 12% to 33% without substantially more tokens or calls (HyMem [108]). *Folding* goes further by replacing a completed branch with a short return message and discarding its trace. Trained folding outperforms standard reinforcement learning on BrowseComp-Plus despite using about a tenth of the active context, 62% versus 54% (Context-Folding [11]). The same operation can condense one interaction or a resolved sequence of steps (AgentFold [55]). These designs choose the unit of compression to match completed work. Folding each capital's search into its returned rows keeps the coordinator focused on the collection rather than every page visit.

Source recovery Keeping source addresses lets the agent recover details omitted by compression. A visual agent can store findings with page identifiers and crop coordinates, then reload

Table 3: Evidence stores differ in what they retain, when they change and whether sources can be reopened.

System	Retained material	When it updates	Source recovery
ReSum [10]	Goal-oriented summary	Periodic summary calls	Only the sources the summary names
ARC [107]	Checklist, interaction memory, latest raw turn	Each turn; rewrite when progress stalls or contradicts itself	None beyond the latest turn
HyMem [108]	Findings returned by isolated subtasks; records of past episodes, current work and tool results	Planner folds a subtask, or a context or activity threshold is reached	Source IDs kept with findings; raw observations deleted
AgentFold [55]	Summary blocks; latest raw interaction	Each step: condense it or merge a resolved range	None reported
Context-Folding [11]	Main thread; each subtask's return message	Subtask ends with a `return` action	None; the subtask's trace is removed
SCORE [48]	Observation summaries and image pointers	Each relevant observation; final reload	Reloads original images by page and crop pointer
DocAtlas [109]	Notes tied to pages; annotated document tree	Agent writes notes; read pages can be archived	Fetches the original pages or the notes on demand
EviGraph [100]	Claims, quoted spans, support and conflict edges	Proposed updates pass a validator	Source ID, quote and its position when available
LineageRAG [99]	Which requirement retrieved each passage; verbatim span	Retrieval per requirement, joint selection, grounding	Requirement IDs, retrieval ranks, verbatim spans
Masking study [19]	Reasoning, actions, recent observations	Observations older than five turns are masked	Masked pages stay in a pool the agent can reopen

the original images for final selection and answering; both admission to the ledger and reloading improve accuracy (SCORE [48]). A document agent can similarly keep page-linked notes apart from bulky pages and fetch either on demand (DocAtlas [109]). The shared mechanism is a compact finding attached to a recoverable original. It lets the agent recover a price base omitted from the note without repeating the search that found the audit report.

When pruning helps Pruning tests whether an agent still benefits from older observations. In a controlled study, observations older than five turns are masked while their pages remain available to reopen [19]. With a weak retriever, pruning improves accuracy by 6 to 7 points across models: removing neglected text frees room for additional searches. With a strong retriever, the result depends on the model. A mid-sized model gains about 12 points on BrowseComp-Plus, while the strongest models lose about 1 point there and 5 on live-web GAIA. These models were still using the evidence that masking removed. Pruning helps when old text mainly consumes context, and harms when the model can already select and combine its useful content.

5 Grounded synthesis

Synthesis must turn retained evidence E_T into a supported and complete output y. These demands can fail separately: a report can cite real pages for claims they never make, or support every sentence while omitting part of the request. Fluent prose hides both. In a benchmark that checks the evidence and intermediate claims behind conclusions, models succeed only 4 to 12% of the time, mainly failing to identify evidence or build the intermediate claims [110]. We

distinguish choosing and verifying answers (§5.1), organizing and revising reports (§5.2), and preserving support through citations (§5.3).

5.1 Answer synthesis and verification

An agent should accept an answer only when the evidence supports it, not when several attempts happen to agree on it. Checking the evidence one condition or premise at a time shows exactly which support is missing. The agent can then search for that support or claim less, for example by reporting a conflict instead of picking one side.

Candidate-specific verification Candidate verification checks each condition rather than choosing the most plausible answer. Invert the journalist's question to ask for one metro line: "Which metro line ended up costing more than twice its budget?" Add that the line must have opened in a European city since 2015, and different searches return candidates that meet only some of these conditions. FineVerify classifies each candidate's conditions as supported, not found or contradicted and then applies explicit selection rules [111]. It beats majority voting by 11 points on BrowseComp-Plus, and counting only supported conditions works better than scores assigned by the model itself. Compressing trajectories into each candidate's supporting evidence, conflicting evidence and reasoning gives a verifier a more focused comparison (STEC [112]). Both approaches replace agreement among sampled answers with a comparison of the evidence for each candidate.

Complete answer sets For a collection, verification must remove false members while retaining every true one. A tram, or a metro line opened in 2014, should be rejected; a qualifying metro line should stay. One approach collects candidates from passages read independently, then generates questions to verify each candidate against more text from the retrieved pool (RI2VER [113]). This raises precision and set F1 for both tested answer models, but removes correct members too. Most of the correct members dropped were generated from partial evidence or model knowledge, with no retrieved passage to confirm them. Verification makes these missing sources visible. Recovering the members with support requires additional retrieval rather than a more permissive acceptance rule.

Identity and ambiguity complicate completeness further. About half the apparent false positives in the answer-set study are answers the annotations miss, name differently or record at another granularity [113]. A metro line's construction name and opening-day service name can otherwise become two rows for one project. Ambiguity changes membership rather than naming: a metro line opened in stages from 2014 to 2017 qualifies under one reading of "since 2015" and not another. On ambiguous multi-hop questions, models retain precision above about 70% but recover far fewer correct answers [114]. Complete sets require both resolving duplicate identities and covering the different readings of the request.

Aggregation and adjudication Aggregation connects findings into an argument. Explaining cost per kilometre requires joining each metro line's cost and length before relating that quantity to factors such as tunnelling. Evidence can be combined bottom-up through a tree, with sibling subquestions sharing findings [51], or organized into claim-linked chains from which sections are written (DeepWeaver [50]). The latter improves completeness and detail over writing from the full source pool. The next check is whether every premise is present. Models can fill gaps from memory and call the conclusion supported; explicitly testing whether the supplied evidence entails it catches more such cases [115]. A cost comparison that silently assumes a common price base has this missing-premise error. Linking the conclusion to its premises makes the required adjustment visible.

When sources disagree, *adjudication* determines what the conflict permits the answer to say. Keeping source-specific candidates separate exposes conflicts before they disappear into a summary (§2.2). A verifier can then request another search only where the conflict remains. Alternating search with a decision to verify or stop matches strong fact-checking frameworks at much lower model and search cost (FIRE [116]). Splitting reasoning and search between jointly trained agents instead makes each factual subquestion a traceable step in the verdict (HARIS [117]). A verifier can also report supported, conflicting, refuted or insufficient evidence instead of forcing a yes-or-no conclusion [101]. For disputed costs, these designs support either an explanation of different scopes or an explicit unresolved conflict, rather than an unjustified single figure.

5.2 Report generation

A report organizes supported claims into an explanation. Its structure can be set after initial research, revised alongside the evidence, or changed by gaps found during drafting. For the metro report, sections on tunnelling, procurement and wages may be planned first; a later comparison may reveal that land acquisition needs separate treatment. We compare these points of interaction between research and writing, then examine consistency and preservation during revision.

Research before drafting Research before drafting can give an outline a broader basis. STORM first collects perspectives from related articles, then stages conversations in which writers question experts answering from search results [24]. Only afterwards does it outline and write the article. In the metro report, engineering, procurement and economics perspectives would prompt different searches. Experienced Wikipedia editors judge the resulting articles well organized 25 percentage points more often than an outline-driven retrieval baseline. Linking evidence to the outline also narrows each section's writing context: a section-by-section writer beats one given the whole evidence bank on every report-quality measure tested (WebWeaver [13]). Broad research and focused writing serve different purposes. The first expands what the report can cover; the second keeps each section tied to its own support.

Revising knowledge and outlines An outline can look complete while leaving an explanatory relation untested. A section about tunnelling may never connect tunnel share to cost per kilometre. A *knowledge graph* records findings and their relations. Keeping this graph beside the outline makes weak relations and disconnected findings explicit search targets. Compared with using an outline alone, this improves report quality and the number of supported citations, despite a slightly lower fraction of citations supporting their statements (DualGraph [118]). Another approach scores proposed outline changes by their contribution to retrieval, organization and a trial draft (ScaffoldAgent [119]). Both revise structure in response to evidence. A separate knowledge record is especially useful when the missing element is a relation between findings rather than another heading.

Draft-driven research A draft makes evidence needs concrete at the point of use. Claiming that greater tunnel share explains higher cost per kilometre requires the tunnel share for each metro line; if it is missing, the writer has a precise search target. At sentence scale, a model can retrieve with a provisional next sentence and regenerate it after checking uncertain content (FLARE [120]). The same feedback can change a report's structure: inspecting successive drafts and adding research subsections improves quality until gains level off at about nine iterations. Reinforcement learning teaches agents to stop near that point, while imitation alone stops too early (AgentCPM-Report [121]). Whole-report revision combined with generating, critiquing and

merging alternatives also improves preference win rate against a commercial agent (TTD-DR [12]). Drafts turn an intention to cover a topic into specific claims whose missing evidence can be searched for.

Consistency across sections and figures Consistency requires each section and figure to use compatible terms and support the same argument. A survey generator can store prior text and terminology, schedule sections by their dependencies, and revise the unwritten plan from that record (SurveyGen-I [122]). Figures need a corresponding connection to claims. A report agent can filter decorative images, rank source figures against the outline and route each figure to the section whose claim it supports (ViDR [123]). This produces reports containing original charts that earlier multimodal systems rarely included. Checking that a visual element supplies evidence also separates it from decoration [124]. A shared account of the report's claims gives both text and figures a reason to be included and a basis for consistency checks.

Preservation during revision Revision must preserve established coverage and support while adding the requested change. Suppose the journalist asks for a section on land acquisition after reading the report. Agents carry out over 90% of requested changes in one study but damage 16 to 27% of previously covered content, even under formatting feedback [18]. Four rounds leave coverage 9 to 26 points below what preserving earlier successes would achieve. Citation support is especially fragile: one system loses up to 67 points in the fraction of cited claims its sources support. Localized edit prompts or a dedicated reviser reduce the damage, and a reviser turns one system's coverage loss into a gain, though citation support still falls. Current revision methods therefore need checks on the claims and citations outside the requested edit, as well as a check that the edit was made.

5.3 Citation grounding

Citation grounding connects statements to inspectable support. Three conditions matter: (i) the cited source exists and can be reached; (ii) it supports the attached statement; and (iii) statements needing support carry citations. *Citation precision* is the fraction of citations that support their attached statement. *Citation recall* is the fraction of statements fully supported by their citations [56]. A topical page that never states the claim can reduce both. Correctness and support can also move in opposite directions: replacing full passages with summaries can improve correctness while lowering citation precision [56]. We compare source availability, constraints and repair during writing, and preservation of support across agents.

Availability and support Source checking can repair unavailable links, but support requires reading the source. In one comparison, about 11% of deep research agents' URLs are hallucinated versus about 5% for search-augmented models. Checking live responses and archives and returning the result to the agent cuts non-resolving citations below 1% [125]. Yet another study finds citation precision of only 39 to 77% even though more than 94% of links resolve and more than 80% match the topic. Increasing the call limit from 2 to 150 lowers citation precision by 42 percentage points on average across two agents [17]. Link repair solves availability; additional source inspection must establish whether the working page supports the claim.

Evidence-constrained generation and repair Support checks can constrain writing at three stages. (i) *Before generation,* assigning papers or claim–evidence records to sections limits what each writer can cite [38, 126, 127]. (ii) *During generation,* a model can infer what a sentence needs, search for a paper, check support and rewrite the query when it fails. Training this sequence lets a compact model recover authors' cited papers more accurately than a much

larger model (ReCite [128]). (iii) *After generation,* breaking sentences into atomic facts permits checking each against its reference and revising unsupported content (Wyvern [129]). Visual citations can undergo an analogous check at region level; rewarding correct bounding boxes also improves answer accuracy (MCite-RL [130]). The stages complement one another: restricting the evidence reduces unsupported choices, checking while writing tests proposed claims, and repair catches what remains.

Support through intermediate agents A coordinator can strengthen a worker's finding beyond what its citation supports. A note saying that final cost exceeded budget can become a claim that it doubled. In one multi-agent system, 85% of final-report errors originate in the coordinator, about a third of those being hallucinations. Giving the coordinator raw snippets instead of synthesized notes, or instructing it to avoid uncited information, raises citation recall by 5 points and precision by 3 to 7 without lowering report quality [131]. Preserving the source text at this hand-off helps the coordinator retain the strength of the original claim.

The opposite failure is evidence that affects the answer but disappears from its citations. Graph-based agents may visit 10 to 12 entities while citing about two. In a small controlled ablation, masking the visited but uncited text lowers accuracy [132]. Checking cited statements alone misses this dependence. A trace linking sources, intermediate findings and final claims is needed to inspect both unsupported strengthening and omitted attribution.

6 Learning investigative policies

Training shapes how the policy searches and how it keeps evidence, but the learning signal often sees only the final answer. A reward on a whole trajectory τ does not identify which action a_t helped. Nor does answer matching check whether retained evidence supports the answer, as the quality term $Q(y, E_T)$ requires. A remembered cost can earn the same reward as a figure found in the audit report. Most recipes first imitate demonstrations and then optimize rewards through *reinforcement learning* (RL) [133, 134]. We compare the tasks and demonstrations used for imitation (§6.1), the rewards used for optimization (§6.2), and the environments that determine which investigations can be learned (§6.3).

6.1 Training data and supervised learning

A policy trained by imitation copies what its demonstrations show, shortcuts included. A demonstration that names the answer from memory, or finds it on the first page, teaches little about how to search. Supervised learning therefore depends on two choices: which demonstrations to train on, and which questions they answer. The best questions cannot be answered without finding new evidence.

Demonstrations and trajectory curation *Supervised fine-tuning* (SFT) teaches a policy to imitate actions from demonstrations. Early browsing work used human demonstrations and answer preferences; choosing among sampled answers with a learned reward model helped more than further RL (WebGPT [53]). Later pipelines use stronger model teachers and select demonstrations for diverse queries, useful retrieval and short successful paths. With this curation, 871 examples suffice to outperform compared RL-trained search agents (SimpleDeepSearcher [135]). Training on preferred pairs of existing trajectories can then improve on imitation alone without generating new trajectories through online RL (OffSeeker [136]). Carefully selected demonstrations make small supervised datasets effective, and offline preference training can refine the behavior they teach.

A teacher can receive cleaner information than the student will see. For example, it can choose actions from retrospectively summarized observations while the student learns those actions from the raw history. Training on these paired views outperforms an industrial RL-trained agent on a Chinese browsing benchmark (OpenSeeker [137]). A related approach distills a supervisor's feedback to several agents into one agent's own reflections (ProductResearch [138]). Both turn guidance available during data construction into behavior the student can reproduce without that guidance at test time.

Constructing search difficulty Synthetic questions determine which search dependencies the policy learns. Some pipelines set difficulty by construction, replacing a known entity with a description that requires another lookup (WebShaper [139]) or shortening clues until the target is harder to retrieve (WebExplorer [140]). A metro line's name, for instance, can be replaced by "the metro line that opened the year the city hosted a summit." Other pipelines use a solver's success to adjust difficulty (SAGE [141]), or train question proposers and solvers together without human-written questions (Dr. Zero [142]). Construction controls the intended dependency; solver feedback tests whether the resulting question actually requires it.

Shortcuts can defeat that intended dependency. If one page states both the summit year and the cost, the nominally multi-hop question needs one search; a memorized cost needs none. In one open training set, about a third of successful trajectories name the answer before any retrieved page contains it [20]. Building tasks from long-tail entities, different source types and obscured constants, then refining them adversarially, delays when the answer becomes available. SFT on this data outperforms comparable open agents on several deep-search benchmarks (FORT-Searcher [20]). Useful search difficulty comes from evidence the agent must acquire, rather than the number of hops written into the question.

Reconstructing process supervision Finished reports can supply training targets for decisions their prose leaves implicit. One approach reconstructs searches and readings that could have produced an expert document, filtering for faithfulness, grounding and consistency. Training on these reconstructions doubles accuracy on a literature question-answering benchmark, from 20% to 40%, while making the agent more likely to read after searching and stop after reading (RetroGen [143]). Reconstructing the process teaches when to inspect and when to stop, in addition to what a good final report says.

Supervised learning versus RL Matched tasks show what RL adds beyond imitation. On the same synthetic questions, RL raises GAIA accuracy by 13.5 points over SFT alone [139]. Curated supervised and offline recipes also match or beat separately trained RL agents [135–137]. RL provides a demonstrated gain within the matched recipe, while the offline results show that data selection can also produce competitive search policies.

6.2 Reinforcement learning

RL lets a policy try investigations that no demonstration showed, and the reward decides which ones it learns to repeat. The simplest reward checks only the final answer. Others also score the plan, the individual searches or the content of a report. In a comparison that changes only the reward, crediting individual actions changes how agents search but does not consistently improve their answers.

Outcome supervision A reward on final-answer correctness can teach search without labeled reasoning steps. It trains a model to pause reasoning for retrieval (Search-R1 [25]), while rewarding tool use and format before answer quality provides a staged route to the same behavior

(R1-Searcher [144]). In one multi-hop setting, outcome-reward training triples the exact match of prompted interleaved retrieval and elicits query correction without explicit process labels (ReSearch [145]). The reward's success still depends on the evidence available. In a widely used setup, the corpus lacks supporting documents for 3,321 of 9,000 questions, yet the agent answers about half of those correctly at least once. The 866 questions on which only some attempts succeed continue to generate a learning signal. For roughly a tenth of the training set, that signal rewards answers the corpus cannot support [146]. Repairing this mismatch requires changing the corpus as well as the reward (§6.3).

Module-specific credit Separate planner and executor training distinguishes a bad plan from a failed search. A plan to identify the airport line before finding its cost remains sound if the cost lookup fails. Training execution by imitation and the planner with a reward on valid subquestions raises one small model's exact match from 13.4% to 23.5% and then 33.5%. An untrained planner instead slightly lowers the small model's score and helps its larger counterpart only modestly (APEX-Searcher [147]). Reference subquestions make that planner reward possible. An executable graph can instead be scored for structure, search quality, coverage and synthesis, avoiding reference decompositions (DecomposeR [148]). Both designs give planning its own feedback, so an executor's failure does not obscure whether the chosen subproblems were useful.

Search and evidence credit Intermediate rewards address the mismatch between a trajectory's final outcome and its useful steps. One analysis finds no evidence in 83.5% of successful trajectories' steps, while 7% of failed trajectories' steps find useful information [149]. Three approaches assign finer credit. (i) *Penalize wasted search.* Comparing steps with better trajectories and penalizing redundant retrieval cuts searches by about 30% (LeTS [150]). (ii) *Reward recovered support.* Scoring entities, citations and complete evidence chains discourages shortcuts (CaRR [151]). (iii) *Trace useful sources to actions.* Following supported answer citations back to the actions that first retrieved their documents credits those actions (STAMP [149]). The third approach credits finding the audit report even if later actions did the writing; the second checks whether the evidence chain is complete.

Citing a retrieved page can itself become a shortcut if the answer never uses its content. Perturbing cited evidence tests that dependence directly: a policy must cite evidence identifiers and change its answer appropriately when that evidence changes (Proof-of-Use [152]). Together with evidence-chain and source-tracing rewards, this moves credit from the mere occurrence of a tool call toward what the action found and what the answer used. These rewards improve on outcome-only training in their respective experiments; controlled comparisons are needed to distinguish their mechanisms from differences in tasks and tools.

Report rewards Report training uses rubrics to reward coverage and analysis that one reference string cannot express. A metro-cost rubric might require a common price base, an explanation of tunnelling and support for each final cost. Rubrics can evolve with the policy's search results and failure modes, retaining criteria that distinguish better reports. A compact model trained this way outscores a commercial agent on a scientific question-answering benchmark at far lower query cost (DR Tulu [153]). Alternatively, constructing a question and rubric from an evidence tree fixes the requirements in advance and reaches comparable results with about 13 times fewer RL GPU-hours (DeepRubric [154]). Evolving rubrics follow what the policy discovers; fixed evidence-derived rubrics provide a stable target at lower training cost in this comparison.

Controlled reward comparisons Controlled comparisons hold the tasks, models and tools fixed while changing the reward. Table 4 compares four methods under five model and tool-call settings [146]: (i) a final-outcome reward (Search-R1); (ii) credit from similar states across

Table 4: Controlled reward comparisons favor the outcome reward in three of five settings; corpus repair improves every method in the paired setting [146].

Method	Training corpus	Avg. EM (%)	Search turns
Qwen3-8B; Hermes			
Search-R1	Wiki-fixed	49.49	1.72
Search-R1	Wiki-18	43.69	2.23
GiGPO	Wiki-fixed	47.23	1.62
GiGPO	Wiki-18	44.11	1.38
IGPO	Wiki-fixed	49.12	2.99
IGPO	Wiki-18	43.70	1.67
Tree-GRPO	Wiki-fixed	44.63	2.07
Tree-GRPO	Wiki-18	42.57	1.79
Qwen2.5-7B-Instruct; Hermes			
Search-R1	Wiki-fixed	45.20	2.09
GiGPO	Wiki-fixed	44.58	1.58
IGPO	Wiki-fixed	33.77	2.72
Tree-GRPO	Wiki-fixed	41.81	2.01
Qwen3-8B; Search-R1 format			
Search-R1	Wiki-fixed	42.40	2.13
GiGPO	Wiki-fixed	45.44	1.61
IGPO	Wiki-fixed	42.87	1.92
Tree-GRPO	Wiki-fixed	44.87	2.08
Qwen2.5-7B-Instruct; Search-R1 format			
Search-R1	Wiki-fixed	39.60	2.11
GiGPO	Wiki-fixed	41.21	1.67
IGPO	Wiki-fixed	38.13	2.01
Tree-GRPO	Wiki-fixed	41.26	1.88
Qwen2.5-7B-Base; Search-R1 format			
Search-R1	Wiki-fixed	45.52	1.99
GiGPO	Wiki-fixed	43.54	1.61
IGPO	Wiki-fixed	29.70	1.00
Tree-GRPO	Wiki-fixed	40.53	1.74

Scores use the repaired corpus (Wiki-fixed) at evaluation. Only the first model–format setting was trained on both the original corpus (Wiki-18) and Wiki-fixed. EM is exact match (§7.1), averaged over four sampled answers per question and then across HotpotQA, 2WikiMultihopQA, MuSiQue, Bamboogle and PopQA. Turns are mean search turns. All settings share the question pool and four-call limit.

trajectories (GiGPO); (iii) credit from changes in reference-answer probability (IGPO); and (iv) credit from branches sharing a prefix (Tree-GRPO). None checks evidence use. The tool-call formats are JSON function calls using the model's learned convention (Hermes), or tags specified in the prompt (Search-R1 format). The outcome reward gives the highest average exact match in three settings. Finer rewards produce different search patterns, including better document recovery, shallower searches or more redundancy, but no consistent accuracy gain. Under this fixed-corpus, four-call protocol, finer action credit changes search behavior more reliably than answer quality.

6.3 Training environments and scaling

A policy can learn only the investigations its training environment makes possible. A fixed or simulated collection is cheap and repeatable, but it must contain the documents each task needs. The live web is closer to real use but slower and less predictable. Training on long investigations, or on teams of agents, also needs ways to keep each run affordable and its learning stable.

Fixed corpora and simulated retrieval Fixed offline collections make long teacher trajectories reproducible with search, open and find operations (OpenResearcher [155]). But a collection without the audit report cannot teach a policy to find the final cost. Adding about 295,000 missing supporting documents to a standard Wikipedia corpus raises every reward method's exact match by 2 to 6 points in the paired model–format setting, under identical evaluation [146]. Methods that score alike before repair separate afterwards. Corpus coverage therefore changes both what the policy can learn and how reward designs compare.

Simulation controls the observations as well as the available documents. A learned search engine can return useful or noisy results, increasing the noise during policy training; such simulators match or surpass training with a real engine in the tested settings (ZeroSearch [156]). A graph and aligned document collection instead guarantee solvable tasks. A policy trained in this environment transfers to web benchmarks, with gains depending on separate search and page-opening actions and a stage of compositional tasks (SearchGym [157]). An expanding local web copy provides another route to faster training interaction (LiteResearcher [158]). These designs preserve the actions and dependencies needed at deployment while reducing the cost of gathering training trajectories.

Live interaction and training throughput Live-web training exposes the policy to failed fetches, irregular pages and changing results. It improves on static-corpus RL by up to 7 points in one comparison and elicits planning, cross-checking and strategy revision (DeepResearcher [57]). Longer trajectories then create uneven generation times: one investigation may finish in ten turns while another continues for a hundred. Decoupling trajectory generation from policy updates lets short runs feed learning while long runs continue, making training beyond 100 turns practical (ASearcher [159]). Exposure to web failures teaches recovery; asynchronous generation makes that exposure affordable across long, uneven investigations.

Context and optimization stability Long-investigation training must also match the context the deployed policy sees. Compressing distant tool responses while preserving reasoning extends a fixed context from about 15 to 100 turns (DeepMiner [160]). If the policy writes the summaries itself, updates change what later steps read and can destabilize learning. Separate summary and action losses, plus a consistency loss between full and compressed histories, stabilize this training (FoldAct [161]). A longer horizon still needs tasks that reward it. In a controlled fixed-corpus comparison, increasing both training and evaluation budgets from 4 to 32 turns lowers exact match for three of four methods [146]. Compression and stable optimization make longer runs possible; task requirements determine whether those extra turns improve answers.

Training cooperating agents Cooperating agents need credit aligned across unequal action sequences. A coordinator may call one worker per capital, with each worker taking a different number of steps. Aligning these trajectories allows joint optimization, rewarding workers for their own execution and the final outcome. It outperforms both a single trained agent and a coordinator with frozen workers (M-GRPO [58]). Dynamically forked trajectories can likewise be linearized for end-to-end training (WideSeek [9]). Joint credit lets improvements in task assignment and local execution reinforce each other rather than leaving one half of the team fixed.

7 Evaluation

Evaluation separates how well the output meets the request, whether its claims have support, how the investigation proceeded and what it cost. These are different parts of the objective in Eq. (3). Each needs a suitable reference: an answer, a complete set, report requirements or

annotations of necessary evidence. We distinguish benchmark task families and their measures (§7.1), the interpretation and validation of output scores (§7.2), and diagnostics of the process and resources (§7.3). Table 5 compares the available references.

7.1 Benchmark task families

A benchmark fixes what the agent receives, which sources it may use and what counts as success. The most informative benchmarks also score the parts of a task. Some check whether the agent found the supporting evidence as well as the answer. Others score a collection or a multi-turn task both item by item and as a whole. Agents often get many parts right and still fail the task as a whole.

Controlled retrieval and open-web investigation Short-answer benchmarks ask the agent to recover a fact by combining sources. Fixed collections include multi-article Wikipedia questions (FRAMES [162]), news-based questions (MultiHop-RAG [163]), and questions with verified supporting documents mixed among plausible distractors (BrowseComp-Plus [164]). The fixed corpus lets experiments change the retriever or policy separately. On the open web, questions instead require persistent search for an answer that is easy to verify (BrowseComp [26]) or navigation into subpages absent from search results (WebWalkerQA [165]). Native Chinese tasks test a different language environment (BrowseComp-ZH [166]); translating the evidence for English questions isolates the language of retrieval (XBCP [167]). Success requires the reference answer, with evidence annotations providing a second check on how it was reached.

Exact match (EM) gives one when a normalized prediction equals a reference answer and zero otherwise; the reported score averages these outcomes. Normalization can remove formatting differences before comparison (GAIA [168]). *Judged accuracy* is the fraction of answers an evaluator judges to match the reference (BrowseComp [26]). Some benchmarks also compare selected evidence with reference support. *Precision* P is the fraction of predicted items that match the reference. *Recall* R is the fraction of reference items recovered. *F1* combines them as $F_1 = 2PR/(P + R)$, with zero assigned when both are zero. Answer F1 compares tokens in the predicted and reference answers. Support F1 compares selected evidence with annotated sentences, paragraphs or documents. *Evidence recall* applies recall to annotated supporting documents.

These answer and evidence measures distinguish a correct result from recovery of its sources. A stronger retriever raises the same agent's BrowseComp-Plus accuracy by 14 points with fewer searches [164]. Translating supporting documents reduces accuracy by 16 to 23 points across agents, while supplying the annotated evidence produces much larger gains [167]. In these settings, access to the right evidence explains more than the number of searches alone.

Collections and reports Collection tasks ask for many qualifying members with verified attributes. When the complete answer set is known, the agent can be scored against it (DeepSearchQA [169]). When an exhaustive reference is infeasible, a task can specify qualifying records and requested counts, then verify each submission against its cited page (WANDR [33]). For the metro example, these alternatives correspond to recovering every qualifying metro line versus supplying a requested number of metro lines with supported costs. They measure different forms of completeness.

Set F1 uses the shared calculation above, comparing returned and reference members after identity matching. *Full correctness* is the fraction of questions whose returned set exactly matches the reference. WANDR instead measures record quality and completion of requested counts, with partial-credit and all-or-nothing variants.[1] On DeepSearchQA, full correctness runs about

[1]WANDR aggregates records through a hierarchy. At each level, precision averages submitted member scores.

15 percentage points below average F1 [169]. On WANDR, the strongest tested system reaches 36.3% soft F1 but only 13.3% hard F1 [33]. Partial recovery and fully supported collections remain far apart under both kinds of reference.

Reports receive a research task and sometimes private files or multimodal attachments. Success means covering the requested facts, connecting them through analysis and supporting the resulting claims. Archived-page tasks can check requested numbers, lists and datasets against task-specific answers (Deep Research Bench [170]). Report-oriented tasks instead use reference reports (DeepResearch Bench [171]) or known insights planted in private files (DRBench [36]). Expert-derived rubrics specify content directly [37]. *Rubric coverage* is the fraction of a task's criteria marked satisfied; *insight recall* applies recall to the planted insights. Relative report scores combine weighted judgments of coverage, depth, instruction following and readability.[2] Agents satisfy fewer than half the items in an expert-derived report benchmark, with the largest gaps in information recall and analysis [37]. Their reports therefore need better discovery and stronger connections between findings, as well as fluent presentation.

Multimodal and continuing investigations Multimodal tasks give the agent visual material to locate or interpret; continuing tasks also require reuse of earlier findings. Success includes reaching the required evidence and applying it in the current answer. InterLV-Search tests visual seeking and interleaved retrieval in both controlled collections and the open web [47]. EpiBench supplies multi-turn literature questions with annotated papers, figures, tables and tool paths, disabling tools in final turns to test reuse [39]. Turn-level answer scores use the matching rules above. *Episode success rate* is the fraction of episodes in which every turn is answered correctly. *Evidence correctness* is the fraction of required evidence units accessed on correctly answered turns. For the controlled levels of InterLV-Search, *retrieval recall@5* is the fraction of tasks whose target entity appears among the final retrieval's top five results [47].

EpiBench's strongest model completes every turn correctly in only about 29% of hard episodes. Its traces reveal incorrect figure references and failed reuse [39]. Separately checking access, each answer and the full episode exposes failures that a correct-sounding final response hides.

Research reports can also be evaluated together with their execution logs. MiroEval checks task-specific synthesis criteria, verifies atomic claims against web and attachment evidence, and audits search and reasoning in the trace [172].[3] Most tested systems lose 3 to 10 points on multimodal tasks relative to text-only tasks. Jointly inspecting the report and trace helps reveal whether an absent visual finding was never acquired, misread or omitted during writing.

7.2 Outcome evaluation

A score for the final output depends on how it was graded, not only on the output. Evaluators decide which answers count as the same, which qualities of a report to judge and how to count supported claims. Many of these decisions are left to a language model acting as judge. Each one can change a score while the output stays the same, so evaluations need to report these choices and check that the judge makes them reliably.

Recall merges duplicate identities using the lowest score for each identity, retains the best distinct members up to the requested count, zero-pads missing members and divides by that count. *Soft scores* allow partial subtree credit; *hard scores* require a fully correct subtree. F1 uses the shared formula, and task scores are averaged equally [33].

[2] DeepResearch Bench's RACE score first combines criterion scores using criterion and dimension weights. The report's final score is its weighted total divided by the sum of its total and the reference report's total [171].

[3] MiroEval's synthesis score weights task-specific criteria and dimensions. Its process score combines ratings of search breadth, analytical depth, refinement, critical thinking and efficiency with report–process alignment checks. Factuality labels distinguish right, wrong, conflicting and unknown claims [172].

Table 5: Benchmark references determine which answers, evidence and report requirements can be checked.

Benchmark	Sources and actions	Required output	Scoring reference	Main measures
FRAMES [162]	Wikipedia retrieval	Multi-hop answer	Answers; supporting articles	Answer accuracy
BrowseComp [26]	Live-web search and browsing	Short answer	Reference answer	Judged accuracy
BrowseComp-Plus [164]	Fixed-corpus retrieval	Short answer	Verified supporting documents	Judged accuracy; evidence recall; calls
BrowseComp-ZH [166]	Chinese web search	Short Chinese answer	Answers; source URLs	Judged accuracy
XBCP [167]	BrowseComp-Plus with translated evidence; fixed retrieval	English answer	Same answers; mapped evidence	Judged accuracy; evidence recall
GAIA: information-seeking tasks [168]	Web; files; multimodal tools	Answer or list	Answers; annotator steps	Exact match
DeepSearchQA [169]	Live-web investigation	Single answers; exhaustive sets	Ground-truth answer sets	Set F1; full correctness
WANDR [33]	Web collection; page evidence	Hierarchical records; excerpts; URLs	Task verifiers; refetched pages	Record precision, recall and F1
Deep Research Bench [170]	Archived pages; live-ranked search	Numbers; sources; lists; datasets	Task-specific answers and evidence	Task correctness; set recall and F1
DeepResearch Bench [171]	Live-web research	Cited report	Reference reports; cited pages	Report score; citation precision; effective citation count
DRBench [36]	Public web; synthetic private files	Cited insights/report	Injected insights; distractors; sources	Insight recall; factual precision; rubric scores
MiroEval [172]	Web; multimodal attachments	Report; process record	Adaptive criteria; retrieved evidence	Report and process rubrics; factuality labels
EpiBench [39]	Paper, figure and table access	Multi-turn answers	Answers; evidence IDs; tool checklist	Episode success; evidence correctness; calls
InterLV-Search [47]	Offline visual retrieval; open web	Short answer	Answers; target entities/chains	Answer accuracy; retrieval recall@5

Shared answer and evidence measures are defined in Section 7.1; report and factuality measures are discussed in Section 7.2. WANDR uses requested record counts rather than an exhaustive answer list. Deep Research Bench and DeepResearch Bench are different benchmarks.

Answer correctness and completeness Outcome scoring compares the submitted answer with the task's reference. Equivalent phrasing and entity identity must be resolved before the scores in Section 7.1 are calculated. "2.1 billion" and "2,100 million" express the same cost, but a budget and a final cost are different quantities. A lenient judge can merge the latter; exact string matching can split the former. Sets likewise require matching a metro line's project and service names to one entity. These decisions change which predictions count as correct. Reporting the matching policy alongside exact match or F1 makes the resulting comparison interpretable.

Report coverage and analysis Report scores depend on which criteria a judge can assess reliably. A reference report establishes an overall comparison, while expert rubric items test whether specific facts and explanations are present. DeepResearch Bench's relative scores agree with expert preferences in 71% of comparisons, versus 68% agreement between experts [171].

Fine-grained expert criteria also expose missing content [173]. Logical quality requires checking whether the explanation follows from its premises: long, persuasive reports can hide broken support between claims. Criteria tailored to each report's context help detect this failure (ReportLogic [174]). Overall preference, coverage and reasoning checks therefore answer different questions about a report's quality.

Factuality, attribution, and source quality Factuality checking breaks an output into atomic statements and tests them against a knowledge source. *Factual precision* is the fraction of those facts supported by that source [175]. Citation precision and recall instead check the pages attached to the statements (§5.3). A report can score well on these support measures while omitting required findings, so they complement coverage scores. Counting conventions also matter. DeepResearch Bench scores deduplicated statement–URL pairs, excludes uncited statements from citation precision, and reports the mean count of supported pairs per task as its *effective citation count* [171]. Other evaluations examine uncited statements through dependencies on earlier sentences [173], or compare attribution with the full retrieved context [176]. Source ratings add credibility, freshness and relevance [177]. Separating claim support, citation coverage and source quality prevents a high score on one from hiding a failure on another.

Judge validity Model judges need validation on the specific decision they make. For long-form answers, early automatic metrics fall well below expert agreement on factuality and completeness [178]. For reports, matching overall human preferences does not establish that every individual criterion is reliable. Criterion-level validation needs domain annotations and a tie policy; changing the latter alone moves annotator agreement from 55% to 83% [179]. Known failures planted in sound traces and reports offer a direct test. The best judges detect only about half, while stepwise assessment catches more than whole-artifact judging (REFLECT [180]). Citation judges also differ in false acceptance and rejection even at similar F1 [181]. A budget page is relevant to a metro line's cost but does not establish the final figure. Validation must therefore test support with cases like this, rather than assume a relevance judge can also verify the claim.

7.3 Process evaluation

A final score shows whether an investigation succeeded, but not why. A correct answer can rest on flawed steps, and a wrong one can follow from a single missed document. Checking the trajectory itself shows where the agent went wrong and what each result cost. Such checks can be repeated only if the sources the agent could reach are recorded, because live pages change and can expose answer keys.

Acquisition, inspection, and integration Process evaluation compares the actions and observations with the evidence the task requires. For the journalist's question, failing to find the audit report is a *retrieval gap*. Opening it but answering with the news report's budget is a *utilization gap*. Evidence recall measures the first against annotated documents; reading checks and interventions help diagnose the second. Across six agents on BrowseComp-Plus, accuracy tracks supporting-document recovery more closely than search count or context use. Retrieval gaps dominate five agents' errors, and most search episodes add no new evidence [21]. Repeated near-identical queries consequently signal stalled discovery rather than thorough investigation.

Finding a document must be followed by reading and integration. Blocking answers until retrieved text has been read raises accuracy by 15 to 20 points on trajectories that otherwise answered unread, and these failures overlap little with errors after reading the right document (Read-Gate [182]). Supplying gold evidence isolates integration: on a scientific reasoning benchmark, it raises accuracy by 19 to 34 points, yet the strongest model answers only 69% of

the hardest questions (Sci-MMR [183]). Complete evidence recovery also trails answer accuracy, showing that correct answers can be reached without all annotated support. Hint-free multi-hop tasks similarly expose agents that retrieve ample evidence but fail to connect it [184]. Finding, reading and combining evidence need separate checks because improving one does not repair failures in the others.

Local errors and controlled interventions A trajectory audit checks intermediate claims and their dependencies, including those in runs with a correct final answer. One audit finds process errors in 37% of successful trajectories [185]. In multimodal search, answer accuracy exceeds supported success by up to 7 points, while judges disagree more on subjective failure labels than on the answer itself [186]. Controlled interventions make the failure test more concrete. Adding a plausible false summary to unchanged true records tests reliance on an answer-shaped shortcut [35]. Recording document-opening probabilities supports a different test, reweighting the read evidence toward the candidate pool (§4.1). The first intervention measures sensitivity to misleading content; the second measures distortion introduced by the agent's own selection. Both connect an outcome change to a specific part of the investigation.

Resources and repeatability Resource measurements put those gains in context. *Latency* is elapsed time. *Monetary cost* sums the prices of model calls, tools and source access. Tool-call counts and token counts measure other parts of $c(\tau)$; concurrent calls can reduce latency without reducing either count. Evaluation adds its own cost when it retrieves pages and runs judges. An assessment of efficiency, grounding and robustness can also test how much of a reference solution the agent must be shown before succeeding, distinguishing inability from a need for hints (TRACE [187]). Resource profiles reveal whether a better answer came from a better investigation or simply more computation.

Repeatability requires recording the source conditions as well as the budget. Archiving pages stabilizes one benchmark, with similar scores on live and archived versions even though archive search still starts from a live engine (Deep Research Bench [170]). Refetching evidence during grading instead makes the score depend on the page's later state [33]. Live search can also retrieve leaked answer keys: one commercial agent visits such pages for 60% of a 100-question medical sample [23]. A comparable result must identify the corpus or snapshot, retrieval and tool settings, budget and judging protocol. For general-assistant benchmarks, it must also identify the information-seeking subset used [168]. These conditions let outcome, support, process and resource scores describe the same investigation.

8 Open problems

Methods now improve individual parts of an investigation, but several demands span the whole loop. An agent must judge completion while discovering the requirements, preserve findings when sources and goals change, and expose which decisions caused an improvement. These demands define the remaining problems of evidence sufficiency (§8.1), transfer across environments (§8.2), and evaluation of complete investigations (§8.3).

8.1 Evidence sufficiency

Evidence-sufficiency checks need requirements against which to test the findings. Known claims support useful stopping rules (§3.4), and predicting structured gaps extends these checks to requirements inferred during search [188]. Yet a requirement never formulated is never checked. Verifying every discovered metro line leaves undiscovered ones untested; adding doubtful candidates instead sacrifices precision [169]. A controller needs to update the requirements as it

discovers candidates and interpretations, and distinguish an unanswered question from one no reachable source can settle. That distinction would let it report an unresolved gap rather than continue indefinitely or guess.

Agreement can overstate confidence when several pages share an underlying source. Several pages may all repeat one cost from one press release. In a study that aggregates reports sharing a source as though they were independent, increasing their number from 1 to 32 lowers the fraction of uncertainty intervals containing the true value from 94% to 26%. Text deduplication barely helps because similar wording does not reliably identify shared origins [189]. Recording support and conflicts or correcting document-selection bias addresses different problems (§4.1). A sufficiency check also needs to trace where the sources obtained their information before treating their agreement as repeated confirmation.

The value of another action must be judged separately from the adequacy of current support. A few decisive clues can settle a narrow claim (§3.4), while broad collections suffer when discovery stops early or snippets replace page-level checks [33]. An agent may have strong support for every returned metro line but weak evidence that the list is complete. Declining search yield is therefore a reason to reconsider the strategy, not automatically a reason to declare completion. When the budget expires first, the output should identify which requirements remain unsupported.

8.2 Transfer across research environments

Transfer requires a finding to retain its meaning across different sources and actions. Standardized objects and knowledge graphs make heterogeneous private data jointly searchable [190], but agents still struggle when a web finding must become a database query or vice versa [46]. A metro line's public name must map to the agency's contract key without losing which project phase the figure concerns. A transferable policy needs to preserve these identity and scope conditions, as well as learn the new tool's query syntax.

Language and modality shifts change which part of the policy fails. Translating evidence primarily exposes retrieval failures (§7.1). Mixed text and image questions can instead leave substantial reasoning failures after the evidence is supplied: the best agent on one benchmark answers only 48% correctly even with gold evidence [191]. Merging specialized web, document and retrieval agents at the parameter level can match mixed-data training while preserving more of their other abilities [192]. Transfer evaluation should report acquisition and interpretation separately, so success in finding translated text cannot hide failure to compare a chart with a contract table.

Sources can change adversarially as well as naturally. Fabricated consensus and forged authority persuade some search agents far more than others; one endorses 31% of attacks overall and 73% of consensus-fabrication attacks [193]. A planted summary can also override records already retrieved (§2.2). A source record should retain the basis for trusting a finding, so altered claims can be checked against the records beneath them.

A user can change the goal without changing any source. Agents already clarify intent before research [194] and accept steering during it [195, 196]. If the journalist restricts the comparison to fully underground metro lines, some accepted members must be removed and earlier discarded evidence may become relevant. The agent needs dependencies between requirements and findings so that a changed goal updates the affected conclusions while preserving the rest.

8.3 Evaluation of complete investigations

Whole-investigation evaluation should attribute a gain to the decisions that produced it. Existing frameworks jointly inspect process and outcome [172], fix multimodal inputs and noisy files for reproducibility [197], or control when an agent delegates search [198]. These provide parts of the

needed test. If the metro report improves, an evaluation should distinguish finding more sources, reading them better, retaining the price base and preserving it during revision. It should also charge the extra verification and judging to the total cost. Connecting controlled interventions to these intermediate records would make a final score explain which behavior improved.

Incomplete references make coverage especially hard to validate. In literature search, a neutral judge rates only about half of human citations as relevant, compared with 88% for the best automatic rerankers; human authors also cite collaborators more often [199]. A research agent can thus find a useful source absent from a human reference list. The metro report may likewise cite an audit missing from the reference report. Expert criteria and independent support checks need to credit such additions while rejecting merely topical material. Judges must be validated for that decision (§7.2), so reference coverage does not become a penalty for finding better evidence.

Continuing investigations need evaluation over successive revisions. Process feedback raises report scores by 8 to 15 points in the second turn, but gains stop compounding for the tested GPT models. By the third turn, one loses about a quarter of previously satisfied criteria [200], consistent with the preservation failures in Section 5.2. Fixed sandboxes make repeated-turn comparisons reproducible, while live sources introduce page changes and answer leakage [23, 197]. Progress should be measured by requirements newly satisfied, earlier support preserved and added cost under the same stated source conditions.

9 Conclusion

Deep research agents extend retrieval into an adaptive investigation. Their actions determine what evidence is found, memory determines what survives, and synthesis determines which claims that evidence becomes. Following these connections separates failures that a final-answer score merges. An agent can miss the audit report, discard a cost's price base or strengthen a supported statement beyond what its citation says. Better retrieval, recoverable notes and claim-level checks address different failures, so each must be assessed at the stage where it acts.

The evidence favors requirements-based stopping over a general judgment of sufficiency, compact findings with routes back to sources over irreversible summaries, and verification that checks explicit conditions over candidate popularity. Verification exposes missing support but can reduce answer-set completeness; revision adds requested content but often damages earlier support. Training depends on the same chain: when required sources are absent, answer rewards credit guesses, and finer action rewards have not consistently improved accuracy under controlled comparisons. Reliable progress therefore requires preserving meaning from source to claim and measuring outcomes, support, process and cost together.

References

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, 2020.

[2] OpenAI. GPT-4 Technical Report. arXiv preprint arXiv:2303.08774, 2023.

[3] Aaron Chatterji, Thomas Cunningham, David J. Deming, Zoë Hitzig, Christopher Ong, Carl Yan Shan, and Kevin Wadman. How People Use ChatGPT. Working Paper 34255, National Bureau of Economic Research, 2025.

[4] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, 2020.

[5] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and Narrowing the Compositionality Gap in Language Models. Conference on Empirical Methods in Natural Language Processing, 2022.

[6] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. ICLR, 2023.

[7] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. arXiv preprint arXiv:2212.10509, 2023.

[8] Tianrui Qin, Qianben Chen, Sinuo Wang, et al. Flash-Searcher: Fast and Effective Web Agents via DAG-Based Parallel Execution. arXiv preprint arXiv:2509.25301, 2025.

[9] Ziyang Huang, Haolin Ren, Xiaowei Yuan, Jiawei Wang, Zhongtao Jiang, Kun Xu, Shizhu He, Jun Zhao, and Kang Liu. WideSeek: Advancing Wide Research via Multi-Agent Scaling. arXiv preprint arXiv:2602.02636, 2026.

[10] Xixi Wu, Kuan Li, Yida Zhao, et al. ReSum: Unlocking Long-Horizon Search Intelligence via Context Summarization. arXiv preprint arXiv:2509.13313, 2025.

[11] Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. Scaling Long-Horizon LLM Agent via Context-Folding. arXiv preprint arXiv:2510.11967, 2025.

[12] Rujun Han, Yanfei Chen, Zoey CuiZhu, et al. Deep Researcher with Test-Time Diffusion. arXiv preprint arXiv:2507.16075, 2025.

[13] Z Li, X Guan, B Zhang, S Huang, H Zhou, S Lai, M Yan, et al. WebWeaver: Structuring Web-Scale Evidence with Dynamic Outlines for Open-Ended Deep Research. arXiv preprint arXiv:2509.13312, 2025.

[14] Yuxuan Huang, Yihang Chen, Haozheng Zhang, et al. Deep Research Agents: A Systematic Examination And Roadmap. arXiv preprint arXiv:2506.18096, 2025.

[15] Abhinav Java, Ashmit Khandelwal, Sukruta Midigeshi, Aaron Halfaker, Amit Deshpande, Navin Goyal, Ankur Gupta, Nagarajan Natarajan, and Amit Sharma. Characterizing Deep Research: A Benchmark and Formal Definition. arXiv preprint arXiv:2508.04183, 2025.

[16] Z Shi, Y Chen, H Li, W Sun, S Ni, Y Lyu, RZ Fan, B Jin, et al. Deep Research: A Systematic Survey. arXiv preprint arXiv:2512.02038, 2025.

[17] Hailey Onweller, Elias Lumer, Austin Huber, Pia Ramchandani, Vamse Kumar Subbiah, and Corey Feld. Cited but Not Verified: Parsing and Evaluating Source Attribution in LLM Deep Research Agents. arXiv preprint arXiv:2605.06635, 2026.

[18] Bingsen Chen, Boyan Li, Ping Nie, Yuyu Zhang, Xi Ye, and Chen Zhao. Beyond Single-shot Writing: Deep Research Agents are Unreliable at Multi-turn Report Revision. arXiv preprint arXiv:2601.13217, 2026.

[19] Haoxiang Zhang, Qixin Xu, Zhuofeng Li, Lei Zhang, Pengcheng Jiang, Yu Zhang, and Julian McAuley. Masking Stale Observations Helps Search Agents – Until It Doesn't: A Regime Map and Its Mechanism. arXiv preprint arXiv:2606.00408, 2026.

[20] Jia Deng, Yimeng Chen, Xiaoqing Xiang, et al. FORT-Searcher: Synthesizing Shortcut-Resistant Search Tasks for Training Deep Search Agents. arXiv preprint arXiv:2606.12087, 2026.

[21] Qi Liu, Jiaxin Mao, Fengbin Zhu, and Tat-Seng Chua. Diagnosing Search Behavior and Failure Modes in Long-Horizon Search Agents. arXiv preprint arXiv:2608.01913, 2026.

[22] Minbyul Jeong. Ko-WideSearch: A Korean Breadth-Search Benchmark for Exhaustive Set Enumeration by Web Agents. arXiv preprint arXiv:2606.27595, 2026.

[23] Yongjie Wang, Xinyue Zhang, Kunhong Yao, Zhiwei Zeng, Kaisong Song, Jun Lin, and Zhiqi Shen. Search-Time Contamination in Deep Research Agents: Measuring Performance Inflation in Public Benchmark Evaluation. arXiv preprint arXiv:2606.05241, 2026.

[24] Yijia Shao, Yucheng Jiang, Theodore A. Kanell, Peter Xu, Omar Khattab, and Monica S. Lam. Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models. arXiv preprint arXiv:2402.14207, 2024.

[25] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arık, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. COLM, 2025.

[26] 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.

[27] 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, 2025.

[28] Renjun Xu and Jingwen Peng. A Comprehensive Survey of Deep Research: Systems, Methodologies, and Applications. arXiv preprint arXiv:2506.12594, 2025.

[29] Jinyan Cai and Ruochong Yao. DeepResearch: A Survey of LLM-based Research Agents. International Journal of Engineering and Technology, 18(2):92–101, 2026.

[30] 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, 2025.

[31] Wenjun Li, Zhi Chen, Jingru Lin, et al. Reinforcement Learning Foundations for Deep Research Systems: A Survey. arXiv preprint arXiv:2509.06733, 2025.

[32] Preetam Prabhu Srikar Dammu, Arnav Palkhiwala, Tanya Roosta, and Chirag Shah. iAgentBench: Benchmarking Sensemaking Capabilities of Information-Seeking Agents on High-Traffic Topics. arXiv preprint arXiv:2603.04656, 2026.

[33] Vitaliy Polshkov, Marcin Pitera, Jeremy Yang, Kirill Priemko, Maksim Gaiduk, Aleksandr Nikolenko, Denis Bykov, Clare Southern, Denis Yarats, and Jerry Ma. WANDR: A Benchmark for Wide and Deep Research. arXiv preprint arXiv:2608.14747, 2026.

[34] Shuai Wang, Yize Zhao, and Qingyu Chen. CLEAR: Cross-Source Evidence Adjudication for Large Language Models in Medicine. arXiv preprint arXiv:2609.16301, 2026.

[35] Jun Nie, Zhiqin Yang, Zhenheng Tang, Yonggang Zhang, Xiaowen Chu, Xinmei Tian, and Bo Han. DRNOISE: Benchmarking Deep Research Agents in Misleading Evidence Environments. arXiv preprint arXiv:2607.17291, 2026.

[36] Amirhossein Abaskohi, Tianyi Chen, Miguel Muñoz-Mármol, et al. DRBench: A Realistic Benchmark for Enterprise Deep Research. arXiv preprint arXiv:2510.00172, 2025.

[37] Ruizhe Li, Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. DeepResearch Bench II: Diagnosing Deep Research Agents via Rubrics from Expert Reports. arXiv preprint arXiv:2601.08536, 2026.

[38] Xincheng You, Qi Sun, Neha Bora, Huayi Li, Shubham Goel, Kang Li, and Sean Culatana. Orchestrating Specialized Agents for Trustworthy Enterprise RAG. arXiv preprint arXiv:2601.18267, 2026.

[39] Xuan Dong, Huanyang Zheng, Tianhao Niu, Zhe Han, Pengzhan Li, Bofei Liu, Zhengyang Liu, Guancheng Li, Qingfu Zhu, and Wanxiang Che. EpiBench: Benchmarking Multi-turn Research Workflows for Multimodal Agents. arXiv preprint arXiv:2604.05557, 2026.

[40] Guanming Xiong and Penghui Zhang. SimpleWikiSearch: A Clean Offline Wikipedia Environment for Agentic Search. arXiv preprint arXiv:2607.26070, 2026.

[41] Xingjun Wang, Gongsheng Li, Qi Fan, Yunlin Mao, Luyan Su, and Yingda Chen. LENS: In-Context Search via Latent Evidence Exploration over Dynamic Raw Documents. arXiv preprint arXiv:2608.16185, 2026.

[42] Salaheddin Alzubi, Creston Brooks, Purva Chiniya, et al. Open Deep Search: Democratizing Search with Open-source Reasoning Agents. arXiv preprint arXiv:2503.20201, 2025.

[43] Yujia Qin, Zihan Cai, Dian Jin, et al. WebCPM: Interactive Web Search for Chinese Long-form Question Answering. arXiv preprint arXiv:2305.06849, 2023.

[44] Tao Yu, Zhengbo Zhang, Zhiheng Lyu, et al. BrowserAgent: Building Web Agents with Human-Inspired Web Browsing Actions. arXiv preprint arXiv:2510.10666, 2025.

[45] Jiejun Tan, Zhicheng Dou, Yan Yu, Jiehan Cheng, Qiang Ju, Jian Xie, and Ji-Rong Wen. HierSearch: A Hierarchical Enterprise Deep Search Framework Integrating Local and Web Searches. arXiv preprint arXiv:2508.08088, 2025.

[46] Ruofan Wu, Peiran Xu, Xiaolong Li, et al. Benchmarking Hybrid Deep Research Across Database Querying and Web Search. arXiv preprint arXiv:2609.09410, 2026.

[47] Bohan Hou, Jiuning Gu, Jiayan Guo, Ronghao Dang, Sicong Leng, Xin Li, Xuemeng Song, and Jianfei Yang. InterLV-Search: Benchmarking Interleaved Multimodal Agentic Search. arXiv preprint arXiv:2605.07510, 2026.

[48] Yucheng Shen, Lingyong Yan, Jiulong Wu, Shuaiqiang Wang, Jianmin WU, Dawei Yin, and Min Cao. Navigating Sparse Evidence: Agentic Visual RAG via Explicit Context Selection and Consolidation. arXiv preprint arXiv:2609.15800, 2026.

[49] Maria Wang, Srinivas Sunkara, Gilles Baechler, Jason Lin, Yun Zhu, Fedir Zubach, Lei Shu, and Jindong Chen. WebQuest: A Benchmark for Multimodal QA on Web Page Sequences. arXiv preprint arXiv:2409.13711, 2024.

[50] Xujia Wang, Yizhe Zhang, Bin Xu, Lei Hou, and Juanzi Li. DeepWeaver: Bridging the Evidence Synthesis Gap in Open-Ended Question Answering. arXiv preprint arXiv:2608.18988, 2026.

[51] Songeun Lee, Kyungjin Min, Injae Na, Suyeong Lee, Chiyoung Kim, and Woohwan Jung. A Tree-based RAG Framework for Evidence-Intensive QA via Adaptive Planning and Topology-Aware Evidence Gathering. arXiv preprint arXiv:2609.04981, 2026.

[52] Qi Dong, Ziheng Lin, and Ning Ding. Stateful Evidence-Driven Retrieval-Augmented Generation with Iterative Reasoning. arXiv preprint arXiv:2604.14170, 2026.

[53] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, et al. WebGPT: Browser-assisted question-answering with human feedback. arXiv preprint arXiv:2112.09332, 2022.

[54] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic Search-Enhanced Large Reasoning Models. Conference on Empirical Methods in Natural Language Processing, 2025.

[55] Rui Ye, Zhongwang Zhang, Kuan Li, et al. AgentFold: Long-Horizon Web Agents with Proactive Context Management. arXiv preprint arXiv:2510.24699, 2025.

[56] Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling Large Language Models to Generate Text with Citations. arXiv preprint arXiv:2305.14627, 2023.

[57] Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. DeepResearcher: Scaling Deep Research via Reinforcement Learning in Real-world Environments. Conference on Empirical Methods in Natural Language Processing, 2025.

[58] Haoyang Hong, Jiajun Yin, Yuan Wang, et al. Multi-Agent Deep Research: Training Multi-Agent Systems with M-GRPO. arXiv preprint arXiv:2511.13288, 2025.

[59] Zhipeng Qian, Zihan Liang, Yufei Ma, Ben Chen, Huangyu Dai, Jiayi Ji, Chenyi Lei, Wenwu Ou, Xiaoshuai Sun, and Qibin Hou. Plan Before Search: Search Agents Need Plan. arXiv preprint arXiv:2605.28354, 2026.

[60] Yanming Liu, Xinyue Peng, Zixuan Yan, Yanxin Shen, Wenjie Xu, Yuefeng Huang, Xinyi Wang, Jiannan Cao, Jianwei Yin, and Xuhong Zhang. Dep-Search: Learning Dependency-Aware Reasoning Traces with Persistent Memory. arXiv preprint arXiv:2601.18771, 2026.

[61] Zhuofan Shi, Mingzhe Ma, Lu Wang, et al. TreeSeeker: Tree-Structured Trial, Error, and Return in Deep Search. arXiv preprint arXiv:2606.11662, 2026.

[62] Hanwen Liu, Yuanfu Sun, and Qiaoyu Tan. DAGent: Evaluate-then-Grow Planning for Deep Research Agents. arXiv preprint arXiv:2609.39154, 2026.

[63] Dayoon Ko, Jihyuk Kim, Haeju Park, Sohyeon Kim, Dahyun Lee, Yongrae Jo, Gunhee Kim, Moontae Lee, and Kyungjae Lee. Hybrid Deep Searcher: Scalable Parallel and Sequential Search Reasoning. arXiv preprint arXiv:2508.19113, 2025.

[64] Xiaoshuai Song, Liancheng Zhang, Kangzhi Zhao, et al. WebSwarm: Recursive Multi-Agent Orchestration for Deep-and-Wide Web Search. arXiv preprint arXiv:2607.08662, 2026.

[65] Yuyao Zhang, Junjie Gao, Zhengxian Wu, et al. SearchOS-V1: Towards Robust Open-Domain Information-Seeking Agent Collaboration. arXiv preprint arXiv:2607.15257, 2026.

[66] Prafulla Kumar Choubey, Kung-Hsiang Huang, Pranav Narayanan Venkit, Jiaxin Zhang, Vaibhav Vats, Yu Li, Xiangyu Peng, and Chien-Sheng Wu. Dont Stop Early: Scalable Enterprise Deep Research with Controlled Information Flow and Evidence-Aware Termination. arXiv preprint arXiv:2604.24978, 2026.

[67] Daeyoung Roh and Donghee Han. HALT: Verification-Aware Stopping for Retrieval-Augmented Search Agents. arXiv preprint arXiv:2608.02009, 2026.

[68] Sriram Selvam and Anneswa Ghosh. Similar Accuracy, Unequal Evidence: Search APIs as Decision Surfaces for Tool-Using Agents. arXiv preprint arXiv:2607.10198, 2026.

[69] Baixuan Li, Jialong Wu, Wenbiao Yin, et al. Nested Browser-Use Learning for Agentic Information Seeking. arXiv preprint arXiv:2512.23647, 2025.

[70] Hongyu Guo, Zhiyu Zheng, Minghao Hu, Shuai Lei, Zhunchen Luo, Chao Ma, Yunbo Cao, and Zhao Cao. AtlasNav: Mitigating Evidence Blindness with Persistent Corpus Navigation. arXiv preprint arXiv:2608.24764, 2026.

[71] Yuanlei Zheng, Pei Fu, Hang Li, et al. Doc-V*: Coarse-to-Fine Interactive Visual Reasoning for Multi-Page Document VQA. arXiv preprint arXiv:2604.13731, 2026.

[72] Yucheng Shen, Jiulong Wu, Jizhou Huang, Dawei Yin, Lingyong Yan, and Min Cao. VISOR: Agentic Visual Retrieval-Augmented Generation via Iterative Search and Over-horizon Reasoning. arXiv preprint arXiv:2604.09508, 2026.

[73] Zhengbo Zhang, Changtao Miao, Jinbo Su, et al. Visual-Seeker: Towards Visual-Native Multimodal Agentic Search via Active Visual Reasoning. arXiv preprint arXiv:2606.15231, 2026.

[74] Lexiang Hu, Yanzhao Zhang, Mingxin Li, Dingkun Long, Yikang Li, Fuwei Zhang, Yisen Wang, and Zhouchen Lin. VisDocAgentBench: Benchmarking Agents for Visually Rich Document Retrieval. arXiv preprint arXiv:2608.17889, 2026.

[75] 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 preprint arXiv:2504.21776, 2025.

[76] Huifeng Lin, Gang Su, Jintao Liang, You Wu, Rui Zhao, and Ziyue Li. Fishing for Answers: Exploring One-shot vs. Iterative Retrieval Strategies for Retrieval Augmented Generation. arXiv preprint arXiv:2509.04820, 2025.

[77] Daocheng Fu, Jianbiao Mei, Licheng Wen, et al. RE-Searcher: Robust Agentic Search with Goal-oriented Planning and Self-reflection. arXiv preprint arXiv:2509.26048, 2025.

[78] Ruochen Zhou, Zhengyu Chen, Luan Zhang, Siyang Gao, Yee Whye Teh, and Shiqi Chen. Explore Before Committing: Hypothesis-Guided Search for Deep Research Agents. arXiv preprint arXiv:2609.01294, 2026.

[79] Jueun Kim, Sungho Park, and Wook-Shin Han. Hi-Q: Hierarchical Evidence-guided Query Refinement for Multi-Hop Question Answering. arXiv preprint arXiv:2608.30468, 2026.

[80] Yiling Tao, Shihan Deng, Meiling Tao, Pengzhi Wei, Zhichao Hu, and Zhihao Zhu. When Search Agents Should Ask: DiscoBench for Clarification-Aware Deep Search. arXiv preprint arXiv:2606.27669, 2026.

[81] Soojin Yoon and Dongha Lee. Personalized Deep Research Query Refinement with Graph-Scaffolded Evidence Grounding. arXiv preprint arXiv:2608.05876, 2026.

[82] Xingyu Liu, Yu Dong, Qizhen Yu, Shiyu Cheng, Zhe Wang, Guan Li, Guihua Shan, Dong Tian, Christy Jie Liang, and Quang Vinh Nguyen. FootprintRAG: Visual Analytics for Evidence Context Refinement in RAG-based Scientific Literature Exploration. arXiv preprint arXiv:2609.19601, 2026.

[83] 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.

[84] Ka Yiu Lee, Yuxuan Huang, Zhiyuan He, Huichi Zhou, Weilin Luo, Kun Shao, Meng Fang, and Jun Wang. InfoSeeker: A Scalable Hierarchical Parallel Agent Framework for Web Information Seeking. arXiv preprint arXiv:2604.02971, 2026.

[85] Guankai Li, Jiabin Chen, Yi Xu, Xichen Zhang, and Yuan Lu. HyperEyes: Dual-Grained Efficiency-Aware Reinforcement Learning for Parallel Multimodal Search Agents. arXiv preprint arXiv:2605.07177, 2026.

[86] Sidhaarth Murali, João Coelho, Jingjie Ning, João Magalhães, Bruno Martins, and Chenyan Xiong. Beyond Parallel Sampling: Diverse Query Initialization for Agentic Search. arXiv preprint arXiv:2606.17209, 2026.

[87] Mingju Chen, Guibin Zhang, Heng Chang, Yuchen Guo, and Shiji Zhou. A-MapReduce: Executing Wide Search via Agentic MapReduce. arXiv preprint arXiv:2602.01331, 2026.

[88] Qinnan Cai, Yibo Zhao, and Xiang Li. Think Big, Search Small: Where Capacity Matters in Hierarchical Search Agents? arXiv preprint arXiv:2607.07548, 2026.

[89] Kyle A. McCleary and James M. Ghawaly. Quantifying the Accuracy and Cost Impact of Design Decisions in Budget-Constrained Agentic LLM Search. arXiv preprint arXiv:2603.08877, 2026.

[90] Zhengru Fang, Senkang Hu, Zhonghao Chang, Yu Guo, Yihang Tao, Hongyao Liu, Mengzhe Ruan, Jun Huang, and Yuguang Fang. Inference-Time Budget Control for LLM Search Agents. arXiv preprint arXiv:2605.05701, 2026.

[91] Roxana Petcu, Kenton Murray, Daniel Khashabi, Evangelos Kanoulas, Maarten de Rijke, Dawn Lawrie, and Kevin Duh. Query Decomposition for RAG: Balancing Exploration-Exploitation. arXiv preprint arXiv:2510.18633, 2025.

[92] Lin Sun, Linglin Zhang, Jingang Huang, Change Jia, Zhengwei Cheng, and Xiangzheng Zhang. BEAR: Budgeted Evidence Allocation for Multi-Document Reasoning. arXiv preprint arXiv:2601.18116, 2026.

[93] Siheng Xiong, Oguzhan Gungordu, James C. Kerce, and Faramarz Fekri. Adaptive Information Control for Search-Augmented LLM Reasoning. arXiv preprint arXiv:2602.01672, 2026.

[94] Ming Ma, Yi Zhu, Yiran Zhong, Feida Zhu, Yuhao Wang, Junhan Shi, Lingrui Mei, Tianming Yang, and Steven Hoi. Omni-Decision: Evidence-Ledger Planning for Omni-Modal Agents. arXiv preprint arXiv:2607.11433, 2026.

[95] Haochen Zhao, Yongxiu Xu, Xinkui Lin, Dong Xie, Jiarui Lu, Yuqi Qian, Yubin Wang, Hongbo Xu, and Gaopeng Gou. Sparse Evidence Can Suffice: Agentic Evidence Seeking for Multimodal Video Misinformation Detection. arXiv preprint arXiv:2607.18080, 2026.

[96] Heydar Soudani, Elizabeth Lingg, Faegheh Hasibi, and Navid Rekabsaz. When Deep Research Agents Stagnate: Enhancing Reasoning with Retrieval-Aware Agent Control. arXiv preprint arXiv:2608.15191, 2026.

[97] Hongji Pu. BELIEFRAG: Making Adaptive RAG State-Aware under Evolving Evidence. arXiv preprint arXiv:2609.39139, 2026.

[98] Weimeng Luo. When Should Multi-Round RAG Stop? Structured Stopping Judgments and Retrieval Reduction in Search-R1. arXiv preprint arXiv:2608.13237, 2026.

[99] Linyao Zheng, Xuhang Shi, Zhifang Mao, Sai Zhou, Shuaixian An, Xiuquan Hou, and Jinze Li. LineageRAG: Harnessing GraphRAG by Constructing Evidence Lineages with Source Grounding. arXiv preprint arXiv:2608.16004, 2026.

[100] Jiashun Chen, Yirong Mao, and Wenhui Que. EviGraph: Towards Verifiable Evidence Construction for Information-Seeking Agents. arXiv preprint arXiv:2608.24667, 2026.

[101] Xingtao Lin, Yubo Feng, Weixin Liu, Hangqi Ren, Junchao Zhou, Caiwan Sun, and You Chen. Search Broadly, Seek Evidence on Both Sides, Decide Narrowly: Evidence-Admissible GraphRAG for Longitudinal Clinical Event Verification. arXiv preprint arXiv:2608.22062, 2026.

[102] Yalun Wu, Bingzhou Wang, Boyang Wang, Peiying Wang, Shaojie He, Yunhan Wang, Shaozu Yuan, and Jiawei Wang. TAEC: Trajectory-Aware Evidence Coordination for Multi-Step Visual RAG. arXiv preprint arXiv:2609.37349, 2026.

[103] Ziyu Ge, Yuhao Wu, Daniel Wai Kit Chin, Roy Ka-Wei Lee, and Rui Cao. Resolving Conflicting Evidence in Automated Fact-Checking: A Study on Retrieval-Augmented LLMs. arXiv preprint arXiv:2505.17762, 2025.

[104] Rohan N. Pradhan and Steve Goley. Trust, but Don't Verify: Epistemic Blind Spots in LLM Source Evaluation. arXiv preprint arXiv:2606.05403, 2026.

[105] Shuyao Xiao, Shengling Wang, Xuan Chen, et al. Search Shapes Conclusions: Auditing Evidence Selection Bias in Deep Research Agents. arXiv preprint arXiv:2609.39026, 2026.

[106] Yong Wu, YanZhao Zheng, TianZe Xu, et al. ContextBudget: Budget-Aware Context Management for Long-Horizon Search Agents. arXiv preprint arXiv:2604.01664, 2026.

[107] Yilun Yao, Shan Huang, Elsie Dai, Zhewen Tan, Zhenyu Duan, Shousheng Jia, Yanbing Jiang, and Tong Yang. ARC: Active and Reflection-driven Context Management for Long-Horizon Information Seeking Agents. arXiv preprint arXiv:2601.12030, 2026.

[108] XinQi Wang, Jinwei Xiao, Sijia Cui, Hongming Zhang, Yanna Wang, Qingyang Zhang, and Bo Xu. HyMem: Hierarchical Context Management for Long-Horizon Agents via Information Isolation. arXiv preprint arXiv:2608.15703, 2026.

[109] Hongchen Wei, Yuanzhe Wang, Bei Liu, et al. DocAtlas: Long-Document Understanding as

Mutable-State Interaction. arXiv preprint arXiv:2608.07527, 2026.

[110] Yubo Sun, Chunyi Peng, Yukun Yan, Zhenghao Liu, Sen Mei, Bangrui Xu, Xuanhe Zhou, Chi Chen, and Maosong Sun. HiEviDR-Bench: A Benchmark for Hierarchical Evidence Aggregation in Deep Research. arXiv preprint arXiv:2607.25151, 2026.

[111] James Xu Zhao, Hui Chen, Bryan Hooi, and See-Kiong Ng. FineVerify: Scaling Test-Time Compute with Fine-Grained Self-Verification for Agentic Search. arXiv preprint arXiv:2606.00660, 2026.

[112] Xinkang Li, Rong Jiang, Xin Song, Ye Wang, Yue Han, and Changjian Li. STEC: Evidence Compression for Deep Search in Open-domain Multi-Hop QA. arXiv preprint arXiv:2607.10795, 2026.

[113] Bingsen Chen, Shengjie Wang, Xi Ye, and Chen Zhao. Inter-Passage Verification for Multi-evidence Multi-answer QA. arXiv preprint arXiv:2506.00425, 2025.

[114] Jiabao Ji, Min Li, Priyanshu Kumar, Shiyu Chang, and Saloni Potdar. DEEPAMBIGQA: Ambiguous Multi-hop Questions for Benchmarking LLM Answer Completeness. arXiv preprint arXiv:2511.01323, 2025.

[115] Cheng Jiayang, Qianqian Zhuang, Haoran Li, Chunkit Chan, Xin Liu, Lin Qiu, and Yangqiu Song. InteGround: On the Evaluation of Verification and Retrieval Planning in Integrative Grounding. arXiv preprint arXiv:2509.16534, 2025.

[116] Zhuohan Xie, Rui Xing, Yuxia Wang, Jiahui Geng, Hasan Iqbal, Dhruv Sahnan, Iryna Gurevych, and Preslav Nakov. FIRE: Fact-checking with Iterative Retrieval and Verification. arXiv preprint arXiv:2411.00784, 2024.

[117] Qisheng Hu, Quanyu Long, and Wenya Wang. Coordinating Search-Informed Reasoning and Reasoning-Guided Search in Claim Verification. arXiv preprint arXiv:2506.07528, 2025.

[118] Zhuofan Shi, Ming Ma, Zekun Yao, Fangkai Yang, Jue Zhang, Dongge Han, Victor Rühle, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. A Tale of Two Graphs: Separating Knowledge Exploration from Outline Structure for Open-Ended Deep Research. arXiv preprint arXiv:2602.13830, 2026.

[119] Zhibang Yang, Xinke Jiang, Yuzhen Xiao, et al. ScaffoldAgent: Utility-Guided Dynamic Outline Optimization for Open-Ended Deep Research. arXiv preprint arXiv:2606.20122, 2026.

[120] Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active Retrieval Augmented Generation. arXiv preprint arXiv:2305.06983, 2023.

[121] Yishan Li, Wentong Chen, Yukun Yan, et al. AgentCPM-Report: Interleaving Drafting and Deepening for Open-Ended Deep Research. arXiv preprint arXiv:2602.06540, 2026.

[122] Jing Chen, Zhiheng Yang, Yixian Shen, Jie Liu, Adam Belloum, Chrysa Papagianni, and Paola Grosso. SurveyGen-I: Consistent Scientific Survey Generation with Evolving Plans and Memory-Guided Writing. arXiv preprint arXiv:2508.14317, 2025.

[123] Zhuofan Shi, Peilun Jia, Baoqin Sun, Haiyang Shen, Sixiong Xie, Yun Ma, and Xiang Jing. ViDR: Grounding Multimodal Deep Research Reports in Source Visual Evidence. arXiv preprint arXiv:2605.13034, 2026.

[124] Chenghao Zhang, Guanting Dong, Yufan Liu, Tong Zhao, and Zhicheng Dou. Towards Verifiable Multimodal Deep Research: A Multi-Agent Harness for Interleaved Report Generation. arXiv preprint arXiv:2605.29861, 2026.

[125] Delip Rao, Eric Wong, and Chris Callison-Burch. Detecting and Correcting Reference Hallucinations in Commercial LLMs and Deep Research Agents. arXiv preprint arXiv:2604.03173, 2026.

[126] Ziyue Yang, Da Ma, Hanqi Li, et al. DeepSurvey: Agent-Oriented Automated Survey Generation with Analytical Depth and Citation Reliability. arXiv preprint arXiv:2605.29522, 2026.

[127] Hang Ding, Yilun Zhao, Tiansheng Hu, Manasi Patwardhan, and Arman Cohan. SciRAG: Adaptive, Citation-Aware, and Outline-Guided Retrieval and Synthesis for Scientific Literature. arXiv preprint arXiv:2511.14362, 2025.

[128] Yuyang Huang, Bobo Li, Jiajia Song, Yuzhe Ding, Chong Teng, Fei Li, and Donghong Ji. ReCite: Agentic Reasoning for Faithful Citation. arXiv preprint arXiv:2609.09156, 2026.

[129] Beatrice Alessandra Motetti, Emilien Guandalino, Daniele Jahier Pagliari, Alessio Burrello, Lorenz K. Müller, Konstantin Berestizshevsky, and Lukas Cavigelli. Wyvern: An Agentic Framework for

Generating Grounded Multimodal Reports. arXiv preprint arXiv:2608.14446, 2026.

[130] Suifeng Zhao, Zida Liu, Xinyu Lei, Lei Sun, Jun Gao, and Sujian Li. MCite-RL: Towards Reliable Multimodal RAG via Citation-enhanced Agentic Reinforcement Learning. arXiv preprint arXiv:2608.21808, 2026.

[131] Eran Hirsch, David Wan, Han Wang, Elias Stengel-Eskin, Mohit Bansal, and Ido Dagan. Who is the Agent to Blame? Localizing Faithfulness and Citation Mistakes in Agentic Deep Research. arXiv preprint arXiv:2608.24306, 2026.

[132] Riccardo Terrenzi, Maximilian von Zastrow, and Serkan Ayvaz. Why Neighborhoods Matter: Traversal Context and Provenance in Agentic GraphRAG. arXiv preprint arXiv:2605.15109, 2026.

[133] Jialong Wu, Baixuan Li, Runnan Fang, et al. WebDancer: Towards Autonomous Information Seeking Agency. arXiv preprint arXiv:2505.22648, 2025.

[134] Kuan Li, Zhongwang Zhang, Huifeng Yin, et al. WebSailor: Navigating Super-human Reasoning for Web Agent. arXiv preprint arXiv:2507.02592, 2025.

[135] Shuang Sun, Huatong Song, Yuhao Wang, et al. SimpleDeepSearcher: Deep Information Seeking via Web-Powered Reasoning Trajectory Synthesis. Conference on Empirical Methods in Natural Language Processing, 2025.

[136] Yuhang Zhou, Kai Zheng, Qiguang Chen, Mengkang Hu, Qingfeng Sun, Can Xu, and Jingjing Chen. OffSeeker: Online Reinforcement Learning Is Not All You Need for Deep Research Agents. arXiv preprint arXiv:2601.18467, 2026.

[137] Yuwen Du, Rui Ye, Shuo Tang, Xinyu Zhu, Yijun Lu, Yuzhu Cai, and Siheng Chen. OpenSeeker: Democratizing Frontier Search Agents by Fully Open-Sourcing Training Data. arXiv preprint arXiv:2603.15594, 2026.

[138] Jiangyuan Wang, Kejun Xiao, Huaipeng Zhao, Tao Luo, and Xiaoyi Zeng. ProductResearch: Training E-Commerce Deep Research Agents via Multi-Agent Synthetic Trajectory Distillation. arXiv preprint arXiv:2602.23716, 2026.

[139] Zhengwei Tao, Jialong Wu, Wenbiao Yin, et al. WebShaper: Agentically Data Synthesizing via Information-Seeking Formalization. arXiv preprint arXiv:2507.15061, 2025.

[140] Junteng Liu, Yunji Li, Chi Zhang, et al. WebExplorer: Explore and Evolve for Training Long-Horizon Web Agents. arXiv preprint arXiv:2509.06501, 2025.

[141] Fangyuan Xu, Rujun Han, Yanfei Chen, Zifeng Wang, I-Hung Hsu, Jun Yan, Vishy Tirumalashetty, Eunsol Choi, Tomas Pfister, and Chen-Yu Lee. SAGE: Steerable Agentic Data Generation for Deep Search with Execution Feedback. arXiv preprint arXiv:2601.18202, 2026.

[142] Zhenrui Yue, Kartikeya Upasani, Xianjun Yang, Suyu Ge, Shaoliang Nie, Yuning Mao, Zhe Liu, and Dong Wang. Dr. Zero: Self-Evolving Search Agents without Training Data. arXiv preprint arXiv:2601.07055, 2026.

[143] Junjie Huang, Jiarui Qin, Di Yin, Weiwen Liu, Yong Yu, Xing Sun, and Weinan Zhang. From Final Artifacts to Trajectories: Retrospective Process Supervision for Evidence-Grounded Long-Form Generation. arXiv preprint arXiv:2608.30461, 2026.

[144] 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, 2025.

[145] Mingyang Chen, Linzhuang Sun, Tianpeng Li, et al. ReSearch: Learning to Reason with Search for LLMs via Reinforcement Learning. arXiv preprint arXiv:2503.19470, 2025.

[146] Yibo Zhao, Zichen Ding, Jiayi Wu, Zun Wang, and Xiang Li. Retrieval, Reward, and Training Protocols: What Matters in Training Search Agents? arXiv preprint arXiv:2605.27881, 2026.

[147] Kun Chen, Qingchao Kong, Feifei Zhao, and Wenji Mao. APEX-Searcher: Refining Credit Assignment with Subgoaling for Agentic Retrieval-Augmented Generation. arXiv preprint arXiv:2603.13853, 2026.

[148] Mustafa Anis Hussain, Xinle Wu, and Yao Lu. DecomposeR: Planner-Centric Reinforcement Learning for Deep Research with Structure-Aware Reward. arXiv preprint arXiv:2605.30824, 2026.

[149] Ke Xu, Han Xu, Xinran Chen, Yuqian Wang, Zhixuan Li, Xiaojian Liu, Changwo Wu, Jianqiang Xia, and Yuchen Li. STAMP: Provenance-Guided Credit Assignment for Deep Search Agents. arXiv preprint arXiv:2607.11172, 2026.

[150] Qi Zhang, Shouqing Yang, Lirong Gao, et al. LeTS: Learning to Think-and-Search via Process-and-Outcome Reward Hybridization. arXiv preprint arXiv:2505.17447, 2025.

[151] Jiajie Zhang, Xin Lv, Ling Feng, Lei Hou, and Juanzi Li. Chaining the Evidence: Robust Reinforcement Learning for Deep Search Agents with Citation-Aware Rubric Rewards. arXiv preprint arXiv:2601.06021, 2026.

[152] Shengjie Ma, Chenlong Deng, Jiaxin Mao, Jiadeng Huang, Teng Wang, Junjie Wu, Changwang Zhang, and Jun Wang. Proof-of-Use: Mitigating Tool-Call Hacking in Deep Research Agents. arXiv preprint arXiv:2510.10931, 2025.

[153] Rulin Shao, Akari Asai, Shannon Zejiang Shen, et al. DR Tulu: Reinforcement Learning with Evolving Rubrics for Deep Research. arXiv preprint arXiv:2511.19399, 2025.

[154] Minghang Zhu, Chuyang Wei, Junhao Xu, Yilin Cheng, Zhumin Chen, and Jiyan He. DEEPRUBRIC: Evidence-Tree Rubric Supervision for Efficient Reinforcement Learning of Deep Research Agents. arXiv preprint arXiv:2606.17029, 2026.

[155] Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen Xie, Yu Zhang, and Wenhu Chen. OpenResearcher: A Fully Open Pipeline for Long-Horizon Deep Research Trajectory Synthesis. arXiv preprint arXiv:2603.20278, 2026.

[156] 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.

[157] Xichen Zhang, Ziyi He, Yinghao Zhu, Sitong Wu, Shaozuo Yu, Meng Chu, Wenhu Zhang, Haoru Tan, and Jiaya Jia. SearchGym: Bootstrapping Real-World Search Agents via Cost-Effective and High-Fidelity Environment Simulation. arXiv preprint arXiv:2601.14615, 2026.

[158] Wanli Li, Bince Qu, Bo Pan, Jianyu Zhang, Zheng Liu, Pan Zhang, Wei Chen, and Bo Zhang. LiteResearcher: A Scalable Agentic RL Training Framework for Deep Research Agent. arXiv preprint arXiv:2604.17931, 2026.

[159] 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.

[160] Qiaoyu Tang, Hao Xiang, Le Yu, et al. Beyond Turn Limits: Training Deep Search Agents with Dynamic Context Window. arXiv preprint arXiv:2510.08276, 2025.

[161] Jiaqi Shao, Yufeng Miao, Wei Zhang, and Bing Luo. FoldAct: Efficient and Stable Context Folding for Long-Horizon Search Agents. arXiv preprint arXiv:2512.22733, 2025.

[162] 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. North American Chapter of the Association for Computational Linguistics, 2024.

[163] Yixuan Tang and Yi Yang. MultiHop-RAG: Benchmarking Retrieval-Augmented Generation for Multi-Hop Queries. arXiv preprint arXiv:2401.15391, 2024.

[164] Zijian Chen, Xueguang Ma, Shengyao Zhuang, et al. BrowseComp-Plus: A More Fair and Transparent Evaluation Benchmark of Deep-Research Agent. arXiv preprint arXiv:2508.06600, 2025.

[165] Jialong Wu, Wenbiao Yin, Yong Jiang, et al. WebWalker: Benchmarking LLMs in Web Traversal. arXiv preprint arXiv:2501.07572, 2025.

[166] Peilin Zhou, Bruce Leon, Xiang Ying, et al. BrowseComp-ZH: Benchmarking Web Browsing Ability of Large Language Models in Chinese. arXiv preprint arXiv:2504.19314, 2025.

[167] Yuheng Lu, Qingcheng Zeng, Heli Qi, Puxuan Yu, Fuheng Zhao, Rui Yang, Hitomi Yanaka, Naoto Yokoya, and Weihao Xuan. Beyond Monolingual Deep Research: Evaluating Agents and Retrievers with Cross-Lingual BrowseComp-Plus. arXiv preprint arXiv:2606.15345, 2026.

[168] G. Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: a benchmark for General AI Assistants. arXiv preprint arXiv:2311.12983, 2023.

[169] N Gupta, R Chatterjee, L Haas, C Tao, A Wang, C Liu, et al. DeepSearchQA: Bridging the Comprehensiveness Gap for Deep Research Agents. arXiv preprint arXiv:2601.20975, 2026.

[170] 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.

arXiv preprint arXiv:2506.06287, 2025.

[171] Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. DeepResearch Bench: A Comprehensive Benchmark for Deep Research Agents. arXiv preprint arXiv:2506.11763, 2025.

[172] Fangda Ye, Yuxin Hu, Pengxiang Zhu, et al. MiroEval: Benchmarking Multimodal Deep Research Agents in Process and Outcome. arXiv preprint arXiv:2603.28407, 2026.

[173] Janghoon Han, Heegyu Kim, Changho Lee, Dahm Lee, Min Hyung Park, Hosung Song, Stanley Jungkyu Choi, Moontae Lee, and Honglak Lee. DEER: A Benchmark for Evaluating Deep Research Agents on Expert Report Generation. arXiv preprint arXiv:2512.17776, 2025.

[174] Jujia Zhao, Zhaoxin Huan, Zihan Wang, Xiaolu Zhang, Jun Zhou, Suzan Verberne, and Zhaochun Ren. ReportLogic: Evaluating Logical Quality in Deep Research Reports. arXiv preprint arXiv:2602.18446, 2026.

[175] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FactScore: Fine-grained atomic evaluation of factual precision in long form text generation. arXiv preprint arXiv:2305.14251, 2023.

[176] Yumo Xu, Peng Qi, Jifan Chen, Kunlun Liu, Rujun Han, Lan Liu, Bonan Min, Vittorio Castelli, Arshit Gupta, and Zhiguo Wang. CiteEval: Principle-Driven Citation Evaluation for Source Attribution. arXiv preprint arXiv:2506.01829, 2025.

[177] Hexi Jin, Stephen Liu, Yuheng Li, Simran Malik, and Yiying Zhang. SourceBench: Can AI Answers Reference Quality Web Sources? arXiv preprint arXiv:2602.16942, 2026.

[178] Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. A Critical Evaluation of Evaluations for Long-form Question Answering. arXiv preprint arXiv:2305.18201, 2023.

[179] Jena D. Hwang, Varsha Kishore, Amanpreet Singh, et al. Deep Research, Shallow Evaluation: A Case Study in Meta-Evaluation for Long-Form QA Benchmarks. arXiv preprint arXiv:2603.06942, 2026.

[180] Leyao Wang, Yanan He, Peng Chen, Asaf Yehudai, Yixin Liu, Rex Ying, Michal Shmueli-Scheuer, and Arman Cohan. Time to REFLECT: Can We Trust LLM Judges for Evidence-based Research Agents? arXiv preprint arXiv:2605.19196, 2026.

[181] Ethan Leung, Elias Lumer, Corey Feld, Austin Huber, Vamse Kumar Subbiah, and Kevin Paul. Do You Need a Frontier Model as a Citation Verifier? Benchmarking Rubric LLMs for Deep-Research Source Attribution. arXiv preprint arXiv:2607.08700, 2026.

[182] Daeyoung Roh and Donghee Han. Before Reasoning Can Fail: Pre-Evidence Procedural Failures in Agentic RAG. arXiv preprint arXiv:2608.02011, 2026.

[183] Jiaqiang Li, Yajie Yang, Zhiheng Xi, et al. Sci-MMR: Benchmarking Multi-Step Evidence-Grounded Scientific Reasoning in Multimodal Agents. arXiv preprint arXiv:2609.11243, 2026.

[184] Maojia Song, Renhang Liu, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, Jingren Zhou, Dorien Herremans, and Soujanya Poria. Demystifying deep search: a holistic evaluation with hint-free multi-hop questions and factorised metrics. arXiv preprint arXiv:2510.05137, 2025.

[185] Jiaming Wang, Ziteng Feng, Jiangtao Wu, et al. Where Do Deep-Research Agents Go Wrong? Span-Level Error Localization in Agent Trajectories. arXiv preprint arXiv:2606.02060, 2026.

[186] Zhengxian Wu, Junjie Gao, and Kai Yang. Silent Failures in Multimodal Agentic Search: A Diagnostic Taxonomy and Cross-Judge Evaluation. arXiv preprint arXiv:2607.19793, 2026.

[187] Yanyu Chen, Jiyue Jiang, Jiahong Liu, Yifei Zhang, Xiao Guo, and Irwin King. TRACE: Trajectory-Aware Comprehensive Evaluation for Deep Research Agents. arXiv preprint arXiv:2602.21230, 2026.

[188] Minghan Li, Junjie Zou, Xinxuan Lv, Chao Zhang, and Guodong Zhou. S2G-RAG: Structured Sufficiency and Gap Judging for Iterative Retrieval-Augmented QA. arXiv preprint arXiv:2604.23783, 2026.

[189] Marc Bara. Epistemic Sybil Resistance: Multiplying AI Agents Without Multiplying Evidence. arXiv preprint arXiv:2609.01873, 2026.

[190] Zhuofan Shi, Zijie Guo, Xinjian Ma, Gang Huang, Yun Ma, and Xiang Jing. IoDResearch: Deep Research on Private Heterogeneous Data via the Internet of Data. arXiv preprint arXiv:2510.01553, 2025.

[191] Han Wang, David Wan, Hyunji Lee, Thinh Pham, Mikaela Cankosyan, Weiyuan Chen, Elias Stengel-

Eskin, Tu Vu, and Mohit Bansal. MERRIN: A Benchmark for Multimodal Evidence Retrieval and Reasoning in Noisy Web Environments. arXiv preprint arXiv:2604.13418, 2026.

[192] Guochen Yan, Jialong Wu, Zhengwei Tao, et al. Exploring Information Seeking Agent Consolidation. arXiv preprint arXiv:2602.00585, 2026.

[193] Yimeng Chen, Zhe Ren, Firas Laakom, Yu Li, Dandan Guo, and Jürgen Schmidhuber. How Much Can We Trust LLM Search Agents? Measuring Endorsement Vulnerability to Web Content Manipulation. arXiv preprint arXiv:2606.16821, 2026.

[194] Haohao Luo, Zexi Li, Yuexiang Xie, Wenhao Zhang, Yaliang Li, and Ying Shen. IntentRL: Training Proactive User-intent Agents for Open-ended Deep Research via Reinforcement Learning. arXiv preprint arXiv:2602.03468, 2026.

[195] Lyumanshan Ye, Xiaojie Cai, Xinkai Wang, et al. Interaction as Intelligence: Deep Research With Human-AI Partnership. arXiv preprint arXiv:2507.15759, 2025.

[196] Bo Pan, Lunke Pan, Yitao Zhou, Qi Jiang, Zhen Wen, Minfeng Zhu, and Wei Chen. InterDeep-Research: Enabling Human-Agent Collaborative Information Seeking through Interactive Deep Research. arXiv preprint arXiv:2603.12608, 2026.

[197] Qianqian Xie, Qingheng Xiong, He Zhu, et al. DR3-Eval: Towards Realistic and Reproducible Deep Research Evaluation. arXiv preprint arXiv:2604.14683, 2026.

[198] Xinhao Yao, Yuanzhuo Liu, Changhao Wang, Yunfei Yu, Haoran Tan, Yuyao Zhang, Ruifeng Ren, Minlong Peng, and Yong Liu. Delegation Intelligence in Deep Search: A Controllable Framework for Disentangled Capability Diagnosis. arXiv preprint arXiv:2607.23524, 2026.

[199] Gaurav Sahu, Laurent Charlin, and Christopher Pal. Rethinking Literature Search Evaluation: Deep Research Helps, and Human Citation Lists Are Not a Ground Truth. arXiv preprint arXiv:2605.29234, 2026.

[200] Rishabh Sabharwal, Hongru Wang, Amos Storkey, and Jeff Z. Pan. Multi-Turn Evaluation of Deep Research Agents Under Process-Level Feedback. arXiv preprint arXiv:2606.09748, 2026.