OPENPANGU DEEPDIVER-V2: MULTI-AGENT LEARNING FOR DEEP INFORMATION SEEKING

openPangu Team, Huawei

ABSTRACT

We present **DeepDiver-V2**, a multi-agent system (MAS) learned to address the limitations of single-agent framework in complex information-seeking tasks. Unlike DeepDiver-V1, which relies on a single LLM to process complex problems, DeepDiver-V2 decomposes problems into subtasks distributed to specialized agents, scaling up collaborative reasoning. To support this coordination, DeepDiver-V2 introduces a lightweight yet effective file-based communication protocol, where agents interact through structured operations—reading, writing, and interpreting documents within a shared workspace. This design ensures scalable collaboration and efficient memory usage. To train DeepDiver-V2, we propose a novel credit assignment and broadcasting learning framework, which provides both stepwise and trajectory-level supervision, allowing precise attribution of agent contributions across long-horizon tasks. This framework is powered by Ascend NPUs and integrated into our in-house reinforcement learning library, enabling robust and efficient MAS optimization. To develop and evaluate DeepDiver-V2, we expand the WebPuzzle-V1 to encompass more challenging open- and closed-ended tasks rooted in real-world open-web settings. WebPuzzle empowers DeepDiver-V2 with ability on both complex question answering and to write high-quality long-form reports exceeding 15,000 tokens. Empirically, **DeepDiver-V2-38B achieves competitive results across multiple benchmarks, including 34.6 on BrowseComp-zh and 13.4 on BrowseComp-en, outperforming most models of comparable scale**. On our internal WebPuzzle-Writing benchmark, DeepDiver-V2 also exhibits superior content quality and factual reliability, as reflected by its high overall score on both automatic and human evaluations.

1 Introduction

Information-seeking (Wilson, 1999) is a fundamental cognitive skill essential for solving complex real-world questions and generating informative content. Despite significant advancements in artificial intelligence, large language models (LLMs) still encounter substantial difficulties in effectively extracting and denoising information from complex, noisy, real-world open-web environments.

Techniques such as Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), reasoning frameworks like ReAct (Yao et al., 2023), and both well-engineered training (Song et al., 2025; Yu et al., 2024; Zheng et al., 2025) and training-free pipelines (Jiang et al., 2023; Li et al., 2025b) have shown promise in building agentic systems to address these challenges (Huang et al., 2025). However, as LLMs evolve, so does the complexity of the tasks they are expected to handle. Real-world agent-oriented problems often require iterative evidence gathering, information denoising, conflict resolution, and reflection with correction. Moreover, complex problems typically need to be decomposed into multiple subtasks—some sequential and interdependent, others parallel and independent. These characteristics pose significant challenges for traditional single-agent frameworks (Russell and Norvig, 2009), as they require the agent to (1) manage long sequences of intermediate processing steps within a single context window, leading to high memory consumption and reduced decoding efficiency, and (2) handle multiple tasks concurrently, where cross-task interference can substantially degrade performance.

To address these challenges, LLM-based multi-agent systems (MAS) (Hong et al., 2024; Li et al., 2023) have been proposed, which decompose complex tasks into manageable subtasks distributed across multiple agents (Minsky, 1986; Wooldridge, 2009). Such decomposition alleviates the burden on individual agents and improves overall efficiency and effectiveness. However, MAS development still faces critical challenges in communication and training. In particular, as each agent's policy evolves during training, the environment

becomes non-stationary from the perspective of individual agents (Hernandez-Leal et al., 2017; Papoudakis et al., 2019). Training agents independently, without accounting for their interactions, can therefore lead to convergence failures or suboptimal performance. Moreover, unlike single-agent settings—where a single policy observes all information, executes all actions, and is solely responsible for outcomes—MAS suffer from more complex credit assignment problems (Mahajan et al., 2019; Sunehag et al., 2018). Each agent operates with partial observations and limited functionality, making it essential to ensure that an agent is (1) **not** penalized for failures it did not cause and (2) not rewarded for successes it did not contribute to. These difficulties, combined with the need to isolate agents' contexts for memory efficiency while still propagating essential information among them, further complicate MAS training and optimization.

To overcome these issues, we extend the DeepDiver (Shi et al., 2025) to a planner-orchestrated MAS equipped with a simple yet effective credit assignment mechanism. In DeepDiver-V2, agents interact through task summaries and shared files stored in a common workspace. Although agents execute tasks in a decentralized manner, they can communicate via two channels: (1) directly reporting intermediate results to relevant agents, or (2) sharing insights and findings by making specific files publicly accessible. This file-based communication design enables indirect information exchange, fostering coordinated problem solving while avoiding excessive context overhead. For credit assignment during training, each step of the planner's trajectory is evaluated using the LLM-as-a-Judge framework (Chiang and Lee, 2023; Dubois et al., 2023; Zheng et al., 2023), and the resulting feedback is propagated to executor trajectories according to their invocation relationships.

To support the training and rigorous evaluation of DeepDiver-V2, we extend WebPuzzle-V1 with a broader set of challenging tasks, including both open-web, closed-ended problems and open-ended, long-form writing tasks. The close-ended tasks are enhanced with more difficult instances and enriched with high-quality, verifiable samples, while the open-ended tasks are specifically designed to evaluate DeepDiver-V2's ability to generate extensive, high-quality reports exceeding 15,000 tokens. Particularly for long-form writing tasks, which have long been challenging to evaluate, we shift away from the traditional fully open-ended report writing based solely on a topic. Instead, we develop rubric-based writing queries that encompass the main topic along with specific, detailed writing points. This modification enables a more streamlined and accurate evaluation process.

The training of the DeepDiver-V2 is accelerated by Atlas 800I A2 and integrated with our in-house custom, production-ready reinforcement learning library, enabling robust and efficient MAS optimization. DeepDiver-V2 demonstrates robust performance across both open-ended long-form writing benchmarks and complex closed-ended QA tasks. With a 38B backend, DeepDiver-V2 achieves scores of 34.6 on BrowseComp-zh (Zhou et al., 2025), 13.4 on BrowseComp (Wei et al., 2025), and 53.0 on Xbench (Xbench-Team, 2025), outperforming most open-source competitors with greater model size. In long-form writing tasks, DeepDiver-V2 rivals the proprietary Deep Research agents on our in-house benchmark WebPuzzle-Writing and produces highly reliable long-form content, as evidenced by a factuality and faithfulness score of 68.09. Further in-depth analysis reveals several key insights: (1) Planner-orchestrated MAS enables collaborative execution for solving extremely complex problems, while the executor remains the primary performance driver and a moderately capable planner covers most coordination needs. (2) Emergent single-agent competence arises as a byproduct of collaborative training: despite being optimized for team objectives, subagents become strong standalone problem solvers. (3) The planner-centric credit assignment provides greater benefit for MAS training compared to approaches that rely solely on filtering training samples by the global outcome.

2 Preliminaries

2.1 Single-Agent vs. Multi-Agent Systems

$$a_t \sim \pi_\theta\left(\cdot \,|\, o_t,\, s_t,\, h_{t-1}\right), \tag{1}$$

where $s_t \in \mathcal{S}$ is the environment state at time t, $o_t \in \mathcal{O}$ the observation (e.g., a query), $a_t \in \mathcal{A}$ the action, and h_{t-1} the interaction history up to step $t-1$.

A **single-agent system** runs one policy π_θ against environment $\mathcal{E}$, handling all subtasks in a single context window. This preserves global coherence and avoids inter-agent synchronization, but it (i) grows context length with intermediate outputs, increasing memory and latency; (ii) forces sequential processing, limiting flexibility to match execution order/structure to complex workflows; and (iii) mixes goals in one context, causing interference and hurting task-specific performance.

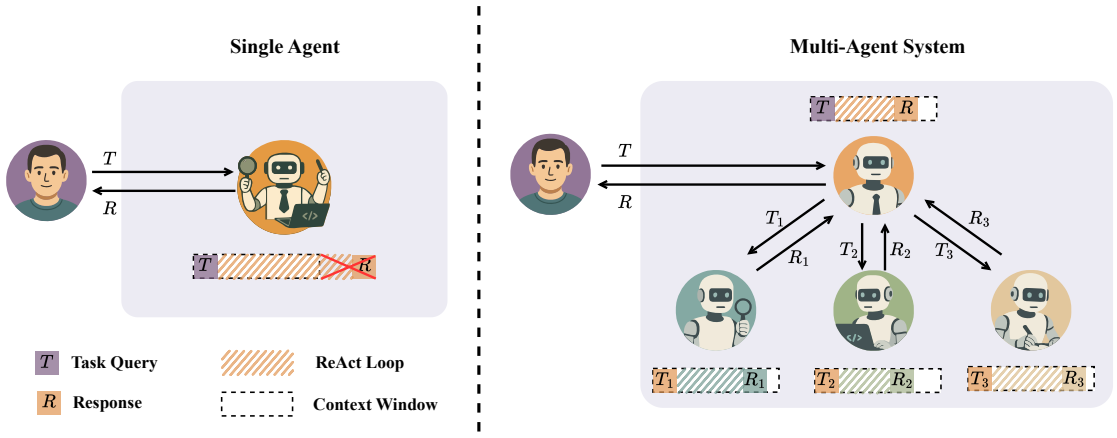


Figure 1: Single agent v.s. multi-agent system. **Left**: a single agent masters all tools and handles all work; complex long-horizon tasks may exceed its context window. **Right**: a multi-agent system (MAS) handles tasks in a divide-and-conquer manner – specialized executor agents handle a subtask at a time under the coordination of a planner agent.

A **multi-agent system** (MAS) comprises policies $\{\pi_{\theta_i}\}_{i=1}^{N}$ with separate local contexts and (optionally) specialized abilities. Agents exchange compressed messages via

$$m_{i\to j}^{t} = \mathcal{C}_{i\to j}\left(c_i^t,\, s_t\right), \tag{2}$$

where $m_{i\to j}^{t}$ is the message from agent i to j at step t, c_i^t is i's private state, and s_t is a globally shared state. By partitioning tasks, MASs shorten per-agent context: each agent works on (c_i^t, s_t) and receives concise $\{m_{k\to i}^{\tau}\}$ instead of full histories, lowering decoding cost while retaining information necessary for coordination. MASs also enable specialization and hierarchical control to align parallel, sequential, or iterative execution with complex workflows.

Two challenges dominate MAS design: **communication** and **training**. Given c_i^t and a memory/token budget B, a communication–memory trade-off is

$$\max_{\{\mathcal{C}_{i\to j}\}} \; I\left(c_j^{t+1};\, m_{i\to j}^{t} \,\middle|\, s_t\right) \quad \text{s.t.} \quad |m_{i\to j}^{t}| \le B, \tag{3}$$

where $I(\cdot;\cdot)$ denotes mutual information under the distribution induced by the agents and $\mathcal{E}$, Overly compact messages risk omitting essential state information, whereas verbose ones negate per-agent context savings.

In addition, MASs suffer from long-horizon credit assignment. Global rewards may be easy to score (e.g., binary correctness in QA), but attributing credit to intermediate (t, i) is hard. With only a global reward R is observed, per-agent credit is challenging to identify. A marginal contribution of agent i at step t can be written as

$$\Delta_{i,t} \;\triangleq\; \mathbb{E}\big[R \,\big|\, h_{t-1},\, s_t,\, a_i^t\big] \;-\; \mathbb{E}\big[R \,\big|\, h_{t-1},\, s_t,\, \tilde{a}_i^t\big]\,, \tag{4}$$

As N and T grow, estimating $\{\Delta_{i,t}\}$ becomes high-variance and combinatorial, making per-agent credit in MASs intrinsically difficult.

2.2 Training Schemas of MAS

Centralized Training & Centralized Execution A CTCE setting models the multi-agent system as a single-agent Markov Decision Process (MDP) governed by a centralized policy. The joint policy

$$\pi : \mathcal{O}_{\text{joint}} \;\to\; P(\mathcal{A}_{\text{joint}})$$

maps the joint observation $\mathcal{O}_{\text{joint}}$—which aggregates all agents' observations—into a probability distribution over the joint action space $\mathcal{A}_{\text{joint}} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_N$. Here, $P(\mathcal{A}_{\text{joint}})$ denotes the set of distributions over joint actions.

Given a complete joint observation, π outputs a distribution over joint actions

$$(a_1, \ldots, a_N) \in \mathcal{A}_{\text{joint}}.$$

At execution time, the central controller observes the full system state and selects all agents' actions jointly, enabling decisions that exploit global information. During training, all experience tuples (s, a, r) are stored and processed centrally to update π using standard single-agent reinforcement learning algorithms.

This paradigm reduces multi-agent learning to a single-agent problem, simplifying algorithm design. However, its scalability is limited: the joint state and action spaces grow exponentially with the number of agents (Gronauer and Diepold, 2022), rendering learning and decision-making computationally intractable in large-scale settings.

Decentralized Training & Decentralized Execution In DTDE, each agent learns and acts independently, without any central coordination (Jaderberg et al., 2019; McMahan et al., 2023; Tampuu et al., 2015; Tan, 1993). For agent i in a multi-agent system (MAS), the policy is defined as

$$\pi_i : \mathcal{O}_i \ \rightarrow \ P(\mathcal{A}_i),$$

where $\mathcal{O}_i$ is the local observation space, $\mathcal{A}_i$ is the action space, and π_i maps a local observation $o_i \in \mathcal{O}_i$ to a probability distribution over $\mathcal{A}_i$. At execution time, each agent selects actions solely based on its own observations. During training, no information is shared between agents; each policy π_i is updated using only local experience—its own observations, actions, and received rewards.

The DTDE paradigm is straightforward to implement and compatible with off-the-shelf single-agent reinforcement learning algorithms. However, the lack of shared information renders each agent's learning environment both non-stationary and partially observed, often leading to instability and suboptimal convergence. Without access to other agents' actions, accurate credit assignment becomes difficult, increasing the risk of coordination failures.

Centralized training & Decentralized Execution CTDE (Gu et al., 2017; Kraemer and Banerjee, 2016; Lowe et al., 2020) combines the advantages of centralized learning with the scalability of decentralized action. For execution, each agent i maintains an individual policy identical in form to DTDE, mapping its local observation $o_i \in \mathcal{O}_i$ to a probability distribution over $\mathcal{A}_i$.

During training, however, agents have access to additional global information or coordination signals. Formally, a centralized learning function $F(\cdot; I_{\text{train}})$ leverages extra information I_{train}—such as the full state s, the actions of all other agents, or other agents' observations—to update policy parameters. This centralized view enables more accurate credit assignment, as the trainer can attribute team rewards to specific agent contributions. In execution, policies operate independently using only local observations, eliminating the need to share raw state or action data at runtime—a critical property in scenarios with strict communication constraints. CTDE thus balances coordination benefits during training with the operational efficiency of decentralized decision-making.

3 Framework Design

To mitigate the context overhead and cross-task interference in DeepDiver-V1's single-agent setup, DeepDiver-V2 adopts a planner-orchestrated multi-agent design with file-based communication. This shift replaces direct context passing with a shared workspace paradigm, enabling efficient information exchange while preserving agent specialization. All agents, including the planner and domain-specific sub-agents, follow the ReAct pattern (Reasoning + Acting), executing tasks through iterative reasoning and tool calling.

3.1 Architecture Overview

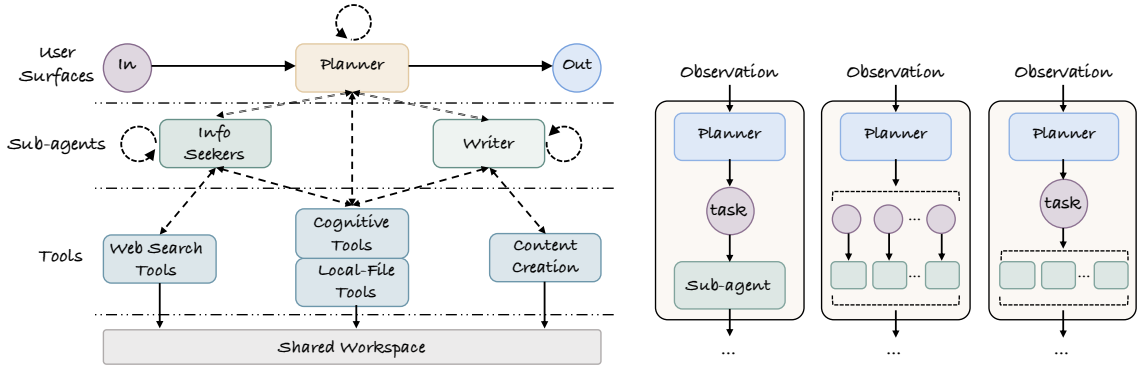
The system comprises a central planner and specialized sub-agents that interact via a persistent shared workspace $\mathcal{W}$ containing structured files $\{f_1, f_2, \ldots, f_k\}$. Rather than exchanging raw task data, agents communicate through standardized reports, balancing communication cost and memory persistence.

For agent i with local state c_i^t, the communication protocol is:

$$m_{ij}^t = \mathcal{C}(c_i^t, \mathcal{W}^t) = \{\text{task\_summary}, \text{key\_files}\}, \tag{5}$$

where,

Task Summary: Concise account of execution steps, tools used, findings, and encountered issues. Agents exchange only compressed task summaries rather than full contexts, reducing interference from irrelevant tasks and minimizing memory overhead.



(a) (b)

Figure 2: Illustration of DeepDiver V2's planner-orchestrated framework. (a) The planner preprocesses queries, decomposes and assigns tasks, orchestrates agents, and assembles the final answer; agents share common tools (e.g., cognitive and local-file ops) while retaining private toolkits; communication occurs via messages and shared files. (b) The planner schedules sequential or parallel subtasks, and can issue "competitive tasks," where the same problem is assigned to multiple agents; their outputs are then verified and consolidated to produce a more reliable final result.

Key Files: Metadata for created or updated files, including file path, purpose, content description, rationale for changes, and whether the file constitutes a primary deliverable. Key files serve as the persistent record of each agent's insights and results. Agents are encouraged (1) to store intermediate outputs, reasoning steps, and supporting materials in separate files, and (2) to retrieve details as needed using our tailored file-operation toolkits, forming a structured, agent-friendly knowledge base.

Equipped with both basic file utilities (e.g., "grep," "find") and LLM-powered "smart" file tools, DeepDiver-V2 agents propagate only summaries while persisting detailed content in files. This design reduces context transmission costs and improves inter-agent communication efficiency. The architecture reduces the cost of context transmission and improves the efficiency of inter-agent communication. This architecture offers three key advantages: (1) **Scalable communication**: the complexity of the messages exchanged between agents remains a controllable size, regardless of the full length of an agent's conversation histories or the deliverable file size. (2) **Persistent state management**: Agents can access historical information without maintaining full context histories; (3) **Parallel execution**: Independent subtasks can be processed concurrently without context conflicts and interference.

3.2 Planner-Orchestrated Coordination

The planner serves as the central coordinator, responsible for query interpretation, task decomposition, and agent orchestration. Given a user query, the planner performs adaptive complexity assessment: simple queries are handled directly without sub-agent invocation, while complex queries trigger structured multi-agent coordination.

Hierarchical Task Decomposition. For complex queries, the planner constructs a depth-tagged task tree where subtasks are organized hierarchically with the user query as the root. The decomposition follows a structured approach where tasks execute level-by-level from shallow to deep, with each depth level completing before the next begins. All tasks at the same depth are mutually independent and can be executed in parallel, ensuring no cross-dependencies within each level. For high-impact subtasks that significantly influence subsequent reasoning, the planner employs a competitive redundancy mechanism by creating near-identical variants at the same depth level to enhance robustness and completion rates.

Iterative Planning and Documentation. The planner maintains comprehensive documentation through versioned planning files that systematically record **task specifications**, **success criteria**, **execution context**, and **structured execution plans**. These documents serve as persistent coordination mechanisms, enabling the planner to track progress and maintain continuity across execution phases. Based on completion results from each depth level, the planner adaptively refines subsequent layers through iterative planning. When

tasks fail or encounter challenges, the system triggers introspective analysis to explore alternative approaches and generate revised task plans.

3.3 Specialized Sub-agents

DeepDiver-V2 uses specialized sub-agents that operate over the shared workspace and exchange standardized messages instead of raw context. We detail two roles used in our experiments: the Information Seeker for evidence acquisition and the Writer for long-form content synthesis.

Information Seeker The Information Seeker agent is responsible for research-oriented subtasks delegated by the planner. The agent continually integrates newly gathered evidence into its working context, refining hypotheses and subsequent actions until sufficient information has been assembled to support decision-making.

The information seeker begins with initial landscape scanning to identify relevant sources, analyzes retrieved candidates (e.g., titles, abstracts/snippets, and provenance), and selectively saves high-fidelity content. Once sources are collected, the agent conducts focused analysis through targeted questioning over the saved content to extract key facts, definitions, quantitative results, and methodological details. When needed, the agent iterates over the same content with multiple, progressively refined questions, using the answers to build a coherent understanding and fill knowledge gaps. Upon completion, the agent produces a structured report summarizing objectives, search strategy, evidence paths, synthesized findings, limitations, and open questions, along with metadata describing the created artifacts to facilitate coordination with other system components.

Writer The Writer agent transforms provided evidence and the problem specification into coherent long-form outputs. It builds a concise outline, assigns references to sections, and writes chapters sequentially with quality control to preserve coherence, avoid redundancy, and remain faithful to the objective.

The writer first reviews the materials to generate an outline aligned with the user's question, prioritizing authoritative, rich, relevant, and timely sources. It then assigns the available materials to outline chapters so that each section is supported by the most pertinent evidence. The agent writes sections strictly sequentially, producing one chapter at a time as a separate artifact. Throughout, it maintains a running summary of completed chapters to ensure global coherence and avoid repetition, without modifying the recorded content of prior chapters. After all chapters are written, it merges the chapter artifacts into a single consolidated document and finalizes the deliverable.

3.4 Tools

DeepDiver-V2 employs a suite of specialized MCP tools to support both the Information Seeker and Writer agents. These tools are designed to extend the agents' capabilities and enable them to perform complex tasks efficiently. We categorize them into two groups: **internal cognitive tools** and **external capability tools**.

Internal Cognitive Tools DeepDiver-V2 formalizes the reasoning capabilities of LLMs into a suite of internal cognitive tools (Anthropic, 2025). Instead of requiring explicit chain-of-thought before generating the final output (e.g., via "<think>" tags), it encapsulates operations such as **thinking**, **reflection**, and **task-oriented message exchange** into dedicated tool calls. These tools are invoked selectively to perform reasoning steps, execute reflection loops, and generate inter-agent messages, with the corresponding cognitive content passed as tool arguments. This design allows DeepDiver-V2 to engage in critical and reflective reasoning only when necessary, preserving both efficiency and effectiveness.

External Capability Tools External tools enable the LLM to interact with its environment and access resources beyond its internal context. DeepDiver-V2 organizes these tools into three categories:

1. **Web-search-centric tools:** These include integrated web search engines, paired with post-processing utilities such as HTML-to-Markdown converters and file downloaders. Additionally, open-source MCP services can be invoked to access domain-specific knowledge. Together, they provide access to external knowledge sources and return information in an LLM-friendly format.
2. **Local-file-oriented tools:** These include file reading, writing, previewing, and search utilities, as well as LLM-based document understanding tools that allow agents to query the contents of a local file and receive context-aware answers. Because DeepDiver-V2 offloads inter-agent communication from direct message passing to a shared file system, these tools allow agents to record information for others and retrieve shared data efficiently.

3. **Content creation tools:** Tailored for the Writer agents, these include document quality evaluation, content-to-section mapping, and section drafting tools. They support structured, high-quality content generation aligned with the overall document plan.

Together, these internal and external tools form a cohesive framework that enables DeepDiver-V2 agents to reason effectively, access and process information seamlessly, and produce high-quality outputs in a structured and efficient manner.

4 Data Curation

We extend WebPuzzle-V1 in two key directions: (1) it introduces more challenging closed-ended problems that require deeper exploration and broaden the scope of information gathering and verification, (2) it incorporates open-ended problems that test the agents' ability to produce comprehensive, long-form content exceeding 20K tokens. In addition to the original WebPuzzle-V1 benchmark, which focuses on evaluating close-ended problem-solving, we propose WebPuzzle-Writing, a complementary benchmark specifically designed to assess the long-form writing capabilities of LLMs.

Example of Riddle

Seed Page: 《Dota 2》是一款由Valve开发免费的多人在线战斗竞技类游戏。2011年...

Key Entities: [Dota 2, Valve, Steam, ...]

Selected Entity: Dota 2

Linked Entity 1: 魔兽争霸III：混乱之治(Warcraft III: Reign of Chaos.)

Linked Entity 2: 起源引擎(Source Engine)

Linked Entity 3: OpenAI

Synthesized Question: 一个美国人工智能研究机构的AI系统在一款知名即时战略游戏的自定义地图所衍生出的游戏作品中，展现出超越职业选手的水平；请问该游戏在什么时间正式完成迁移到其开发商的第二代自研引擎？（精确到年月）？
Solution: 2015年9月/ September 2015

Example of Writing Query

System: SYSTEM_PROMPT

User: 帮我生成5个"体育"领域符合用户日常习惯且有一定深度的搜索查询。

Assistant: ["NBA球队薪资帽演变"，"世界杯扩军会带来啥影响"，"2024年奥运会新增了哪些项目"]

Search API: Searching Query="世界杯扩军会带来啥影响"

Search Results: Web1: {"Snippet": xxx}, Web2: {"Snippet": xxx}, ...

User (Generated): 基于查询语句"世界杯扩军会带来啥影响"以及网页内容{Web}，设计出具有深度的报告写作主题并为每个主题...

Assistant: 请撰写一份题为《世界杯扩军对非传统足球强国的崛起契机：竞技提升、经济溢出与国家形象塑造的跨案例比较》的研究报告，报告需系统覆盖以下5个核心要素：
1. 界定"非传统足球强国"标准，选取至少5个在近两届大赛中表现亮眼或获扩军直接利好的国家...
2. 比较这些国家在参赛前后的FIFA排名、青训体系投入及海外球员输出量，评估大赛经验对技术水平的促进作用...

Figure 3: Examples of close-ended problem and open-ended writing query.

4.1 Extension of the Close-ended Problem

The original WebPuzzle dataset in DeepDiver-V1 includes two task types: (1) Riddle problems, which deliberately obfuscate or generalize properties of a target entity, and (2) Cross-page QA problems, which require reasoning across multiple webpages to identify the correct answer. While WebPuzzle already presents significant challenges for LLMs, we identified complementary limitations in these subsets. Specifically, (1) Riddle queries employ intentionally obscured search clues, but typically do not require multi-hop information seeking; whereas (2) Cross-page QA tasks involve multi-hop reasoning, but the queries often contain overly explicit search cues.

To address these gaps, we extend the dataset by combining cross-page information synthesis with contextual obfuscation, as illustrated in Figure 4. Given a single candidate entity's Wiki corpus, we first identify informative content and extract related entities using an LLM. For each related entity, we collect a bunch of corresponding Wiki fragments, then iteratively expand outward by extracting and following additional related entities. This process is repeated until sufficiently rich contextual information is assembled. Finally, an LLM selects key fragments across the gathered context and applies obfuscation techniques to generate more challenging, closed-ended problems.

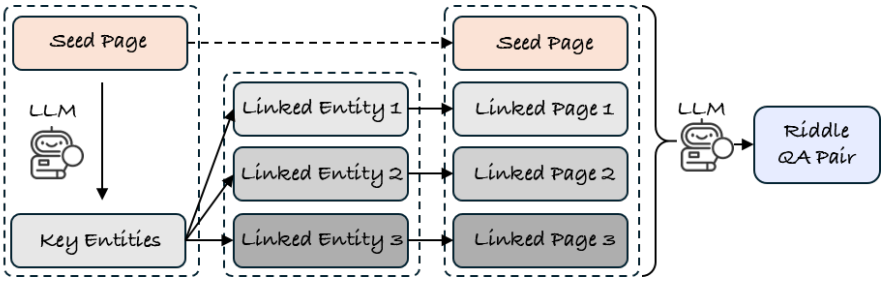


Figure 4: Data curation pipeline of close-ended problems.

4.2 Open-ended Writing Query Crafting

Analogous to the closed-ended setting, open-ended writing-query generation can be organized into two approaches: (1) Internal-knowledge–driven pipeline. This approach leverages the parametric knowledge of the LLM itself. We provide the model with domain-specific seed keywords and instruct it to propose candidate writing topics without relying on external retrieval. The resulting topics are shaped by the LLM's prior knowledge, which allows for broad coverage of concepts, and (2) Retrieval-driven "inverted" pipeline. Similar to the extension of the closed-ended problems, this approach begins not from the model's memory but from targeted web search. As shown in Figure 5, for a given domain, our pipeline first formulates candidate search queries, retrieves relevant web pages, and then synthesizes the retrieved information into draft topics. These drafts are iteratively refined as additional evidence is incorporated, ensuring that the final report queries are grounded in external sources rather than solely in parametric recall. This "inverted" process thus emphasizes evidence accumulation and cross-document integration, producing richer and more contextually faithful writing topics.

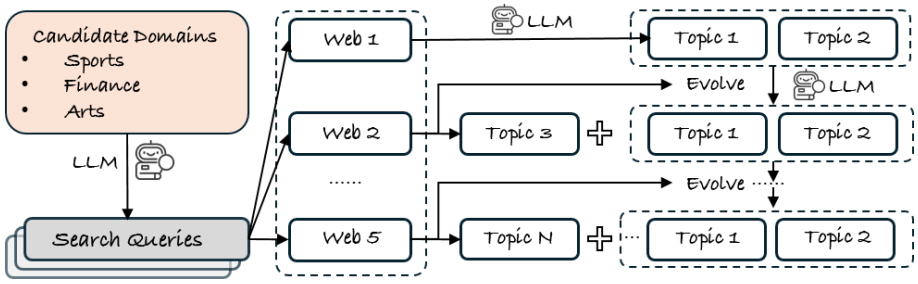


Figure 5: Automatic data collection pipeline of open-ended writing queries.

4.3 WebPuzzle-Writing Benchmark Construction

Existing writing benchmarks, such as EQ-Bench (Paech, 2023), LongWriter (Bai et al., 2025), and WritingBench (Wu et al., 2025b), tend to rely on fixed instruction templates, restrict queries to short contexts, and lack well-specified, knowledge-intensive requirements, thereby undermining their diagnostic effectiveness. To comprehensively and faithfully evaluate LLMs' capabilities for long-form, knowledge-intensive writing, we introduce the WebPuzzle-Writing Benchmark, comprising 100 high-quality writing queries. Each query includes a writing topic and an associated set of writing points that ground rubric-based evaluations. Importantly, all writing points are knowledge-driven, encouraging models to retrieve relevant documents from the web rather than relying solely on their intrinsic knowledge to generate responses.

Specifically, we first meticulously define a seed set of domains, comprising 70% domains overlapping with the training set and 30% out-of-distribution domains. Following the procedure used for collecting training queries, we gather at least five writing queries for each domain. Subsequently, these queries undergo manual filtering and cleaning, with a maximum of three out of the five queries being retained for each domain. These candidate samples are evaluated by eight expert annotators, all of whom hold a master's degree or higher. Each expert is instructed to adhere to the annotation guidelines, which assess the samples across three dimensions: clarity and fluency, knowledge intensiveness, and accessibility of knowledge from websites.

To mitigate hacking attempts, such as search engines identifying the original web pages from which a query is derived, we employ the following two approaches: (1) For queries derived from documents, we select private documents as our knowledge base. These documents cannot be directly accessed through web searches; however, their content can be inferred by comprehensively analyzing a collection of relevant web pages. (2) For queries derived from web-based information, we enforce a restriction that the queries must incorporate information from at least three distinct websites.

Through multiple rounds of iterative annotation and validation, we ultimately collect 100 high-quality queries spanning 10 domains. Our benchmark distinguishes itself from existing writing benchmarks by encompassing diverse domains, longer contexts, grounding in external evidence, and specific requirements that collectively enhance the verifiability of the queries.

5 Training Algorithm

DeepDiver-V2 adopts a Rejection Sampling Fine-Tuning (RFT) stage following an initial cold-start Supervised Fine-Tuning (SFT). Combined with our planner-orchestrated framework, it employs a planner-centric credit assignment algorithm that broadcasts rewards from the planner's perspective to its sub-agents and the LLM-based tools. In this section, we describe the overall training pipeline, the reward design for both close- and open-ended problems, and the credit assignment and broadcasting process.

5.1 Cold Start SFT

A multi-agent framework requires its policy model to follow instructions for multi-role play, enable collaboration among agents, and complete tool invocations—particularly interactions with the local file system involved in such invocations. To equip the policy model with basic, general multi-agent capabilities and ensure its stable operation within our DeepDiver-V2 multi-agent framework, we collected problems from multiple domains, including some of the closed-ended and open-ended problems we constructed as mentioned in Section 4, as well as some in-house problems from other vertical domains. Finally, we carefully synthesized and post-processed thousands of agent reaction trajectories as our cold-start data.

5.2 RFT with Fine-grained Step Filtering

To further strengthen DeepDiver-V2's performance, we employ Rejection Sampling Fine-Tuning (RFT), which generates a large pool of trajectories for self-training. DeepDiver-V2 follows an iterative RFT paradigm (DeepSeek-AI et al., 2025; Yang et al., 2024b), where seed training samples are generated by the model itself from the cold-start checkpoint. These seed samples are then used to continue fine-tuning the checkpoint, producing an updated model that generates additional training samples. After filtering, the newly collected samples are merged with the original seeds, and training is rolled back to the cold-start checkpoint using this expanded dataset.

Importantly, sampling and filtering across iterations follow a coarse-to-fine scheme, ensuring that only high-quality reasoning steps are retained for training.

Trajectory-level filtering via final answer verification. At the trajectory level, we apply an automatic correctness check against ground-truth answers when available. Only trajectories that yield the correct final solution are preserved, while incorrect ones are discarded. Since we sample multiple trajectories for each query, there can be several trajectories that achieve the correct solution for each query. For such cases, we only retain the one with the fewest ReAct steps. We found such a quality-first strategy yields better performance than using all correct trajectories. Such an observation motivates us to conduct further filtering at a fine-grained step level, as illustrated below.

Step-level filtering with LLM as a judge. While final answer verification provides a coarse filter, it does not guarantee that each intermediate reasoning or execution step is valid or useful. To address this, we introduce

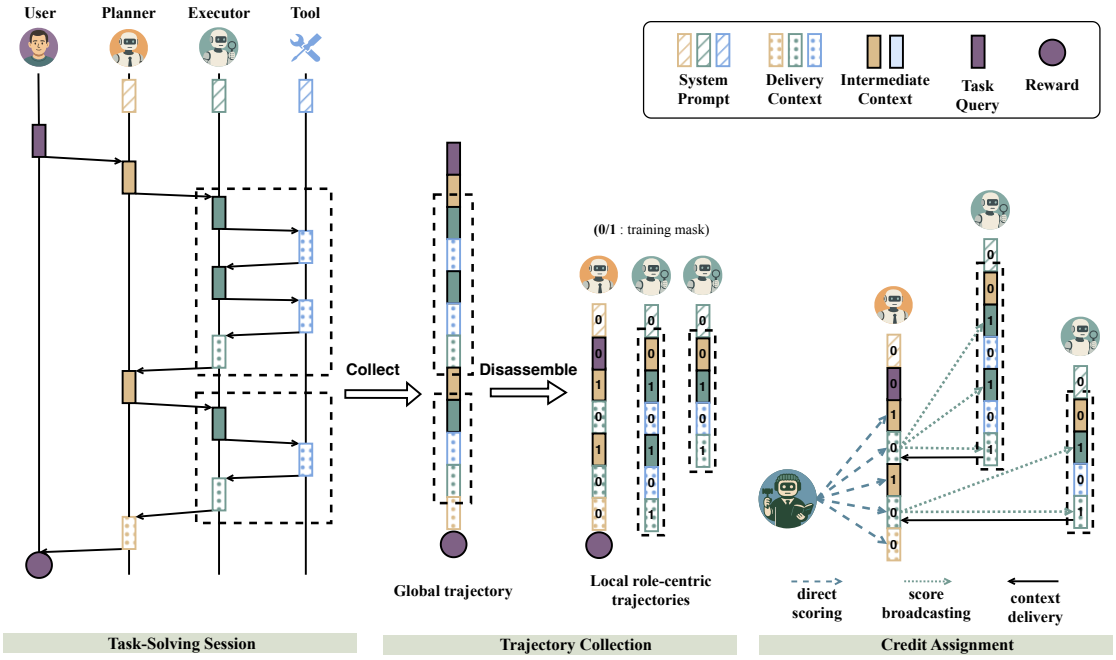


Figure 6: Data organization and processing for training. Given a task-solving session, the global trajectory is disassembled to local role-centric ones for SFT. Fine-grained credit assignment is applied for fine-grained step-level rejection sampling.

an LLM-based critic to assign fine-grained credit scores, enabling identification of steps that meaningfully contribute to task completion. As shown in fig. 6, the global trajectory of a task-solving session is disassembled into local role-centric ones. Then, the scoring process follows a centralized training paradigm co-designed with the planner-orchestrated inference: given the query, reference answer, and the corresponding steps in a planner trajectory, the LLM outputs per-step scores ranging from 1 to 10, based on plausibility, logical coherence, and relevance to the final response. These scores are then propagated to executor trajectories via invocation links: if a planner context block corresponds to a sub-task handled by an executor, its score is broadcast to the associated intermediate contexts within the executor trajectory. This planner-centric scoring leverages the planner's global view of all subtasks, allowing training the MAS as a whole with inter-role interaction considered, while avoiding the overhead of processing full global or all local trajectories. During training, agent-generated content with scores below a predefined threshold τ is masked out from the loss calculation, ensuring that only high-quality steps influence model updates.

By combining trajectory-level correctness checks with step-level plausibility assessments, our coarse-to-fine filtering framework yields a curated set of reasoning and execution traces that are both accurate and interpretable. These "positive samples" provide a safe and effective training signal for improving MAS performance under the current RFT scheme, which can be viewed as an offline, reward-based learning paradigm without explicit use of negative samples (Shao et al., 2024).

Importantly, the planner-centric credit assignment is forward-compatible with future reinforcement learning (RL) extensions, which perform on-policy trajectory sampling and explicitly incorporate negative samples for gradient-based updates. As an initial step, we introduce an online RFT stage (see section 5.3) to explore the impact of on-policy sampling. Leveraging negative samples is left for future work.

5.3 Online RFT

Building on the planner-centric credit assignment strategy, we extend offline RFT to the online paradigm. Instead of relying on a static dataset collected beforehand, DeepDiver-V2 continuously interacts with the environment, collecting new data tailored to the current policy. Migrating from offline to online RFT, however, introduces additional complexity in batching: filtering rollouts produces a variable number of training samples for a fixed inference batch size. To address this, we adopt a rollout-buffered batching strategy.

Concretely, the inference engine generates rollouts continuously and asynchronously, and once a predefined batch size of valid samples is accumulated, these samples are passed to the training engine to update the policy parameters, which are then synchronized with the inference engine. Beyond dynamic batching, we further improve efficiency through the use of partial rollout (Team et al., 2025). This approach enables scalable handling of very long reasoning chains and trajectories, making online RFT practical for the multi-agent and long-horizon information-seeking tasks targeted by DeepDiver-V2.

6 Training Infrastructure

Supporting large-scale multi-agent training necessitates the need to ensure usability, efficiency, and stability. To address this, we design a comprehensive reinforcement learning (RL) framework that integrates disaggregated and colocated architectures, agent factory, and staleness-aware synchronization protocols.

6.1 Compute Cluster

Our multi-agent RL algorithm is trained on a large-scale computing cluster consisting of over one thousand Ascend Neural Processing Units (NPUs) . Each node contains 8 NPUs, which are connected through the Huawei Cache Coherence System (HCCS) in a full-mesh topology and every NPU is provisioned with 64 GB of memory. For inter-node communication, the cluster employs RDMA over Converged Ethernet (RoCE), utilizing 200 Gbps links to enable high-bandwidth connectivity between NPUs across nodes.

6.2 Agent Factory: Algorithm-Oriented Design for Agentic RL

In order to simplify the development and training process in agentic reinforcement learning (RL), we introduce the Agent Factory, an algorithm-first codebase designed to abstract away the underlying complexities of agent training. The Agent Factory prioritizes algorithmic workflows over infrastructure. It provides a modular, declarative, and extensible interface that enables researchers to rapidly instantiate, train, and evaluate agents without needing to manually configure low-level system details.

6.3 StaleSync: Staleness-aware Synchronization Strategy

In reinforcement learning (RL), especially in multi-agent systems (MAS), data skewness often leads to inefficient resource utilization. Some agents in the system may complete their rollouts much more slowly than others, causing idle times and bottlenecks in the training pipeline. This imbalance in data generation can severely affect overall throughput.

To address this issue, we introduce StaleSync, a staleness-aware synchronous scheduling mechanism that enhances training efficiency by tolerating stale gradients during model updates in multi-agent systems. We allow training tasks to run in parallel within a specified staleness threshold, while dynamically adjusting this threshold to prevent performance degradation from excessive gradient staleness.

Through this approach, StaleSync improves resource utilization by minimizing idle NPU time and maximizing data throughput, particularly in MAS where interactions and rollout times can vary significantly. This ensures faster and more efficient multi-agent RL training while maintaining a balance between training speed and model performance.

7 Experiments

7.1 Setup

Benchmark Datasets For complex question answering evaluation, we select challenging benchmarks including BrowseComp-zh (Zhou et al., 2025), BrowseComp (Wei et al., 2025), and Xbench-DeepSearch (Xbench-Team, 2025). For the long-form writing task, we evaluate on WebPuzzle-Writing benchmark, as introduced in Section 4.3, which consists of 100 high-quality writing queries.

Baseline Models Our DeepDiver-V2-7B and DeepDiver-V2-38B are based on openPangu-7B (Chen et al., 2025b) and Pangu-38B (Chen et al., 2025a) respectively. We thoroughly evaluate our model's performance through comparison with following baselines:

- **Complex Question Answering**: Existing methods can be categorized into three types: First, in the Direct Inference category—where models leverage their internal knowledge to answer questions—we include openPangu-7B, Pangu-38B, and along with models from the Qwen (Team, 2025b; Yang et al., 2024a), GPT (OpenAI, 2025b,c,d), and DeepSeek (DeepSeek-AI et al., 2025) families. Second, among Proprietary Agents, we compare against OpenAI DeepResearch (OpenAI, 2025a), Grok-DeepResearch (xAI, 2025), and Doubao with Deep Think and Search (Doubao, 2025). Finally, for Open-source Agents, we benchmark against recent open-source web search agents such as R1-Searcher (Song et al., 2025), WebThinker (Li et al., 2025c), WebDancer (Wu et al., 2025a), WebSailor (Li et al., 2025a), and MiroThinker (Team, 2025a).
- **Long-form Writing**: We primarily compare our approach against proprietary commercial DeepResearch products, including OpenAI o3 DeepResearch (OpenAI, 2025a), Gemini-2.5-Pro Deep Research, and Doubao Deep Research.

Evaluation Framework For the evaluation of the three datasets in complex question answering, we all adopt the LLM-as-a-Judge approach, and the prompts used for evaluation are consistent with the official ones of each benchmark. To comprehensively assess the quality of the generated reports, we propose a two-phase evaluation framework comprising rubric-based evaluation and holistic evaluation. In the rubric-based evaluation phase, for each rubric, we first identify the most relevant sections of the report using a pre-trained embedding model. Subsequently, we employ the LLM-as-a-Judge approach to assign a score ranging from 1 to 10 to the selected sections. The rubric-based evaluation score reflects two key aspects: (1) the extent to which the chosen sections fulfill the requirements of the rubric, and (2) the overall writing quality of the sections in terms of fluency, logical coherence, and content diversity. During the holistic evaluation phase, we utilize the LLM to assess the overall writing quality of the entire report. This evaluation considers various dimensions, including logical coherence, structural completeness, information richness, quality of argumentation, language and style, and content diversity. Each dimension is scored on a scale from 1 to 10.

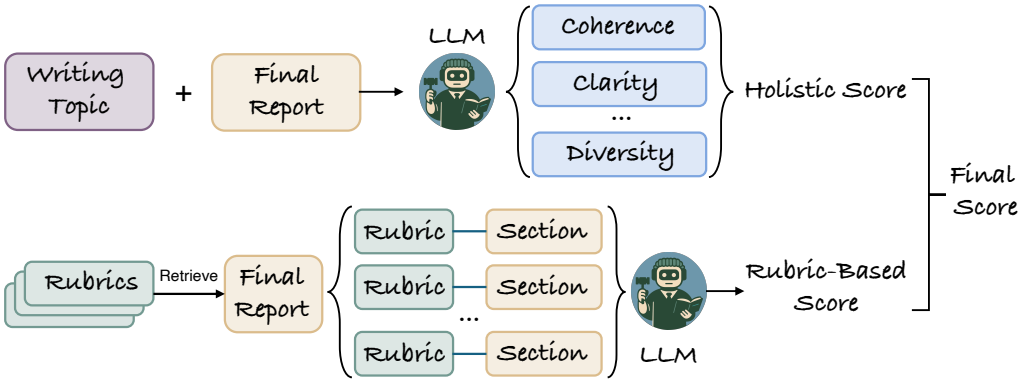


Figure 7: Evaluation framework for long-form report writing.

7.2 DeepDiver-V2's Ability on Complex Question Answering

DeepDiver-V2 demonstrates competitive performance compared to leading proprietary LLMs, while substantially surpassing most open-source agents of comparable size. Despite having fewer parameters than many competing systems, DeepDiver-V2-38B delivers strong performance across all benchmarks. On BrowseComp-zh, it achieves 34.6%, outperforming WebSailor-72B (30.1%) by 4.5 points and matching the current most capable model GPT-5 (34.3%). On Xbench-Deepsearch, it scores 53.0, rivaling the much larger WebSailor-72B (55.0). Moreover, although trained primarily on Chinese queries, DeepDiver-V2 generalizes well to English, attaining 13.4% on BrowseComp-en—the highest among open-source agents.

Despite its smaller size, DeepDiver-V2-7B outperforms several larger baselines and consistently leads across all benchmarks when compared to WebSailor-7B of the same scale. It achieves 18.3% on BrowseComp-zh, surpassing WebSailor-7B by over 4 pts, and delivers 39.0% on Xbench-DeepSearch, exceeding WebSailor-7B and matching the performance of WebDance-QwQ (39.0%). These consistent gains show that even at modest scales, DeepDiver-V2 provides obvious improvements in the information seeking targeted QA tasks.

	Paradigm	BrowseComp-zh	BrowseComp-en	Xbench-DeepSearch
Direct Inference				
openPangu-7B	Direct	2.8	0.0	7.0
Pangu-38B	Direct	4.5	0.4	12.0
Qwen-2.5-32B‡	Direct	3.9	0.6	8.7
Qwen-2.5-72B‡	Direct	7.0	0.6	12.7
QwQ-32B‡	Direct	10.0	0.5	10.7
DeepSeek-R1‡	Direct	26.3	2.0	**32.7**
GPT-4o‡	Direct	6.2	0.6	18.0
GPT-4.1‡	Direct	14.4	1.5	17.0
GPT-5	Direct	**34.3**	**19.8**	30.0
Proprietary Agents				
Grok-3 (Research)‡	-	12.9	-	50+
Doubao (Deep Search)‡	-	26.0	-	50+
Openai DeepResearch‡	-	**42.9**	**51.5**	-
Open-source Agents				
R1-Searcher-7B‡	ReAct	0.6	0.4	4.0
WebThinker-32B-RL‡	ReAct	7.3	2.8	24.0
WebDancer-QwQ‡	ReAct	18.0	3.8	39.0
MiroThinker-8B-DPO-v0.1‡	MAS	13.6	8.7	-
MiroThinker-32B-DPO-v0.1‡	MAS	17.0	13.0	-
WebSailor-7B‡	ReAct	14.2	6.7	34.3
WebSailor-32B‡	ReAct	25.5	10.5	53.3
WebSailor-72B‡	ReAct	30.1	12.0	**55.0**
DeepDiver-V2-7B	MAS	18.3	8.3	39.0
DeepDiver-V2-38B	MAS	**34.6**	**13.4**	53.0

Table 1: Main results on three complex question answering benchmarks. ‡ indicates that the results are cited from the official results of the respective methods or from previous related work.

7.3 Performance of DeepDiver on Long-form Writing

DeepDiver-V2 achieves competitive long-form writing quality, closely matching proprietary research agents while with better content diversity and informativeness. As shown in Table 2, automation evaluation scores indicate that DeepDiver-V2-38B delivers balanced performance across all dimensions of report writing. In particular, it achieves an average score of 5.83, close to the performance of the Gemini-2.5-Pro Deep Research while slightly better than the OpenAI DeepResearch. In the rubric-based automatic evaluation, as shown in Table 3, our model also demonstrates strong instruction-following capabilities with an average score of 5.91, remaining competitive with the top baseline. A notable characteristic of our model is its generation of substantially more detailed reports, with an average length of 24.6K tokens, more than double that of its competitors. This advantage in comprehensiveness appears to be a key factor in human evaluations. As presented in Table 4, when compared directly with OpenAI o3 DeepResearch, DeepDiver-V2-38B was preferred by human evaluators in 58% of holistic evaluations (vs. 19%) and 52% of rubric-based evaluations (vs. 10%). The preference shown by human judges for the richer, more detailed content our model produces indicates that comprehensiveness is a critical component of high-quality long-form writing. The smaller 7B variant, while scoring lower overall, also performs comparably and produces significantly longer reports (25.9K), demonstrating the scalability of our approach in prioritizing content richness.

	Logical Coherence	Structural Integrity	Argumentative Quality	Language Clarity	Content Diversity	Average Score	Average Length
OpenAI o3 DeepResearch	**5.90**	5.81	4.90	6.23	5.93	5.75	10.6K
Gemini-2.5-Pro Deep Research	5.86	**5.97**	**5.14**	**6.44**	6.10	**5.90**	12.2K
Doubao Deep Research	5.36	5.91	4.58	6.13	5.84	5.56	18.1K
DeepDiver-V2-7B	5.30	5.60	4.70	6.17	5.80	5.51	**25.9K**
DeepDiver-V2-38B	5.66	5.89	5.00	6.41	**6.18**	5.83	24.6K

Table 2: Automatic holistic evaluation results of long-form report writing.

	Rubric-based Score	Section-quality Score	Average Score
OpenAI o3 DeepResearch	**6.25**	**6.02**	**6.11**
Gemini-2.5-Pro Deep Research	5.57	5.87	5.74
Doubao Deep Research	5.74	5.57	5.64
DeepDiver-V2-7B	5.70	5.50	5.60
DeepDiver-V2-38B	5.96	5.87	5.91

Table 3: Automatic rubrics-based evaluation results of long-form report writing.

	Win	Tie	Lose
Rubric Evaluation	52	38	10
Holistic Evaluation	58	23	19

Table 4: Human evaluation of DeepDiver-V2-38B vs. OpenAI o3 DeepReseach.

Model	Supported	Uncertain	Refuted
OpenAI o3 DeepReseach	67.56%	23.96%	8.48%
DeepDiver-V2-7B	65.19%	23.66%	11.15%
DeepDiver-V2-38B	**68.09%**	23.07%	8.84%

Table 5: Verification results of factual claims.

8 Analysis

In this section, we analyze the impact of key design choices in DeepDiver-V2. We first examine the contribution of individual roles in the multi-agent framework by switching the models of each agent, evaluating their effectiveness across different tasks. For complex information-seeking QA, we further isolate the information seeker from the planner by providing the seeker with the original queries, instead of relying on the planner's task decomposition and status monitoring. This setup enables a direct comparison between single-agent and multi-agent configurations. For long-form report writing, we perform an automated evaluation of factuality using VeriScore (Song et al., 2024), providing a complementary measure of the reliability of DeepDiver-V2's generated content. Moreover, to complement the quantitative results, we present a case study highlighting the qualitative strengths of DeepDiver-V2 in long-form report generation.

8.1 Factuality Evaluation of Long-Form Reports

Human and automatic evaluations suggest that DeepDiver-V2 performs on par with proprietary agents in terms of coherence, diversity, and clarity. However, factuality and reliability remain especially critical for long-form generation. To assess robustness, we therefore compare the factual accuracy of reports produced by DeepDiver-V2 against those generated by o3-DeepResearch.

Setup Given that long-form generations often contain numerous factual claims, we follow VeriScore (Song et al., 2024) and SAFE (Wei et al., 2024) to verify the factual accuracy of claims in the reports. Specifically, we randomly sample 30 verifiable factual claims from each report. For each claim, we generate three relevant search queries and retrieve supporting evidence from the web. The retrieved snippets are then concatenated, and the LLM is prompted to determine whether the claim is correct based on the provided evidence.

Results *DeepDiver-V2-38B achieves slightly higher factual reliability than o3-DeepResearch, while even the smaller 7B variant maintains strong factual grounding and nearly matches o3's performance.* As presented in Table 5, DeepDiver-V2-38B and o3 DeepResearch demonstrate comparable performance, with approximately 67%–68% of claims being supported and only around 8% refuted. However, when the model size is reduced to 7B, the proportion of supported claims decreases to 65.19%. Notably, the 7B model retrieves sufficient external evidence, indicating that the reduced support rate is primarily due to increased hallucinations during evidence integration. This observation highlights that claim verification in long-form generation is limited not only by retrieval coverage but also by the model's reasoning and synthesis capabilities. Furthermore, smaller models are more susceptible to integration errors, even when retrieval quality remains comparable.

8.2 Role Switching

We investigate the impact of role deployment across tasks, with a particular focus on hybrid configurations that combine small and large LLMs. The goal is to assign models to roles where they are most effective:

identifying which agents in DeepDiver require the most capable LLMs, and which can be reliably supported by moderate-scale models. This analysis highlights how efficiency and performance can be balanced through strategic role switching.

Setup For information-seeking QA, we separate the *planner*—responsible for high-level reasoning (decomposing problems, tracking progress, assigning tasks)—from the *information seeker*, which executes evidence gathering and verification under the planner's instructions. For long-form writing, we partition the DeepDiver-V2 into two groups: material gathering and verification modules (e.g., planner + information seeker) and writing modules (i.e., writer + section writer). We then rotate the 38B and 7B Pangu models across roles to assess capacity sensitivity, identifying where large models are essential and where smaller models suffice.

Results of QA task *Executor capability primarily drives performance, while a smaller planner typically suffices for most tasks.* As shown in Table 6, substitute the 7B baselines with 38B agents consistently brings benefits. Specifically, upgrading only the executor improves by +9 (27.3 vs. 18.3) on BrowseComp-zh and +7 (14.0 vs. 7.0) on BrowseComp-en, whereas upgrading only the planner gives slightly smaller gains of +6.3 (24.6) and +3 (10.0).

Results of Writing task *Sub-agents responsible for writing play a more critical role in the long-form writing task*. As demonstrated in Table 7, replacing the writing sub-agents of the original 7B baseline with their corresponding 38B modules results in a combination that achieves writing performance comparable to the 38B baseline (5.84 vs. 5.87). However, substituting the information-gathering components (e.g., planner and information seeker) with their more advanced counterparts yields only marginal improvements (5.56 to 5.61). This finding underscores the pivotal role of the writing modules in determining overall performance.

	Paradigm	**BrowseComp-zh**	**BrowseComp-en-100**	**Xbench-DeepSearch**
DeepDiver-V1-7B	ReAct	11.4	3.0	19.0
WebSailor-7B	ReAct	14.2	-	34.3
WebSailor-32B	ReAct	25.2	-	53.3
DeepDiver-V2-7B-Info Seeker	ReAct	15.9	4.0	37.0
DeepDiver-V2-38B-Info Seeker	ReAct	26.3	10.0	52.0
DeepDiver-V2-7B	MAS	18.3	7.0	39.0
w/ 38B-Planner	MAS	24.6	10.0	-
w/ 38B-Info Seeker	MAS	27.3	14.0	-
DeepDiver-V2-38B	MAS	34.6	16.0	53.0

Table 6: Role switching experimental results on the BrowseComp-zh, BrowseComp-en and Xbench-DeepSearch.

	Holistic Score	**Rubrics-based Evaluation**		**Average**
		Rubric Score	Setion-Quality Score	
OpenAI o3 DeepResearch	5.75	6.25	6.02	5.93
Gemini-2.5-Pro Deep Research	5.90	5.57	5.87	5.82
Doubao Deep Research	5.56	5.74	5.57	5.60
DeepDiver-V2-7B	5.51	5.70	5.50	5.56
w/ 38B-Writer & Section Writer	5.80	5.97	5.80	5.84
w/ 38B-Planner & Info Seeker	5.56	5.76	5.57	5.61
DeepDiver-V2-38B	5.83	5.96	5.87	5.87

Table 7: Role switching experimental results on the long-form report writing task.

8.3 Isolation Testing of Executor

During RFT training of DeepDiver-V2, our planner-orchestrated framework decomposes complex problems into multiple sub-tasks or assigns a single task to several competitive sub-agents. This design yields key benefits: (1) The burden of complex tasks is offloaded to coordinated multi-agent execution, improving both

efficiency and effectiveness. (2) Each original problem is expanded into several related questions, enriching the diversity of training samples available to sub-agents. Consequently, a natural question arises: *to what extent can a single sub-agent perform on the original complex problems when tested in isolation, without assistance from the planner?*

Setup For both the 7B and 38B versions of DeepDiver-V2, we isolate the information seeker from the planner and evaluate performance on the benchmarks introduced in Section 7. To enable efficient validation, we randomly sample 100 instances each from BrowseComp-en, while for BrowseComp-zh and Xbench-DeepSearch we use the full test set.

Results *Collaborative training of the multi-agent system incidentally produces sub-agents that are themselves strong single agents.* This aligns with prior findings (Baker et al., 2020; Estornell et al., 2025; OpenAI et al., 2019), which show that agents trained cooperatively often develop into versatile individuals capable of high-level performance on their own. As shown in Table 6, while the primary objective of DeepDiver-V2 is to maximize cooperative ability for information-seeking tasks, the information seeker also emerges as a competent standalone agent for complex questions. Specifically, the 38B information seeker adapts seamlessly to solo problem solving, achieving a score of 26.3 on BrowseComp-zh and surpassing WebSailor-32B (25.2). Moreover, on Xbench-DeepResearch—which is comparatively easier than the BrowseComp benchmarks—the 38B information seeker alone performs on par with the full DeepDiver-V2-38B system. These results suggest that while multi-agent collaboration is essential for tackling extremely challenging problems, the resulting sub-agents emerge as efficient and effective solutions for simpler tasks.

8.4 Report Writing Case Study

The following case study illustrates the effectiveness of DeepDiver-V2-38B to long-form report writing. Our generated report exhibits a high degree of structural clarity, meticulously adhering to the requirements outlined in the writing query. A key distinguishing feature of this output is its extensive incorporation of factual data and descriptive details retrieved from external sources. This enables a level of granularity and accuracy that cannot be achieved by relying solely on the model's intrinsic knowledge. This observation highlights our model's advanced capabilities in information seeking and fact-based content generation.

Case Study: 安徽省南北分区早餐文化的多维解读

Writing Query

请撰写一份题为《江淮大省的味觉缩影：安徽省南北分区早餐文化的多维解读与社会意义》的研究报告，报告需全面覆盖以下 6 项具体内容：

1. 区域划分与地貌环境说明：阐释长江与淮河在自然地理上的分界作用，并说明其对食材获取、粮食种植与早餐形态的影响。
2. 代表性早餐案例研究：从皖北、江淮之间、皖南各选取至少8种具有地区标识度的早餐食品，详述制作流程、口味特点与消费场景。
3. 口味叙事与身份认同分析：探讨当地居民如何通过早餐表达地域归属、乡愁或社会阶层，结合访谈、田野笔记或二手文献进行论证，体现人类学与社会学视角。
4. 经济与产业链考察：梳理安徽早餐相关的种植、加工、流通、零售链条，评估产业规模、就业贡献与商业模式，提供数据支撑并揭示区域差异。
5. 早餐消费时段与外卖特征：利用平台数据或实地调研，对比合肥等主要城市早晨5-7时与9-10时订单占比，讨论城市化水平与就业结构对消费时段的塑形作用。
6. 传统与现代冲突探讨：从速食化、健康化、连锁化等趋势切入，评估其对传统早餐口味、技艺与消费方式的冲击及在地化应对方案，提出批判性见解和改进建议。

Generated Report (DeepDiver-V2-38B)

Report Content

```
# 标题：《江淮大省的味觉缩影：安徽省南北分区早餐文化的多维解读与社会意义》

### **第一章 摘要**
安徽省，作为地处中国南北交汇要冲的大省，其独特的地理区位与复杂多样的自然环境，深刻地塑造了省内南北迥异的饮食文化，其中早餐文化作为日常生活的缩影...

### **第二章 地理环境与区域划分对早餐文化的塑造作用**
```

2.1 淮河与长江的地理分界及其生态影响

淮河作为分界线，其南北两侧的气候差异显著。淮河以北地区，属于暖温带半湿润季风气候，冬季气温较低，降水相对较少；而淮河以南地区，则属于亚热带湿润季风气候，四季分明，雨热同期，降水充沛。这种气候上的分野，直接决定了南北两侧的农业类型和主要粮食作物。淮北平原地区，由于气候和土壤（尤其是广布的砂姜黑土）条件，更适合种植小麦、玉米、大豆等旱作作物。这种以麦、豆为主的农业结构，为皖北地区形成了以面食为核心的早餐文化奠定了物质基础，如各种烙饼、馒头、面条等成为主流选择...

2.2 地貌环境对食材获取与粮食种植的制约

安徽省的地貌环境呈现出极大的多样性，全省被划分为淮北平原、江淮丘陵、皖南山地三大自然区域，山地、丘陵、台地、平原的面积分别占全省土地总面积的15.3%、 14.0%、 13.0%和49.6%。这种复杂的地形起伏，加上土壤类型的多样，共同决定了不同区域农业生产的可能性，从而直接影响了当地居民食材的获取方式和早餐的构成...

2.3 气候条件对早餐形态的深层影响

热量条件是决定粮食作物种植类型和熟制的关键因素。安徽省的≥10℃活动积温在 4700℃·d 至 5400℃·d 之间，无霜期长达200至250天，这为一年两熟甚至三熟的种植制度提供了可能。在热量相对充足的沿江及江南地区，实行以双季稻为主的种植模式，早稻和晚稻的广泛种植，使得米饭、米粥、米饺等米制品成为当地居民早餐的主食选择。而在热量条件稍逊的江淮地区，则以冬小麦-水稻轮作为主的一熟稻或双熟制（冬小麦-一季稻）较为普遍。淮河以北的暖温带地区，≥10℃积温相对较低，主要实行冬小麦-大豆的一年两熟模式，小麦成为最主要的粮食作物，因此面食在皖北早餐中占据绝对主导地位...

其次，降水量的空间分布差异，深刻影响了食材的加工方式和早餐的口味偏好。安徽省年降水量呈现由北向南递增的趋势，北部约770-900毫米，南部可达1700毫米以上。淮北平原地区降水偏少，且季节分配不均，旱涝灾害频繁，这在一定程度上影响了作物的生长，也使得当地饮食文化中更注重保存和浓缩风味。例如，通过发酵制作的酱菜、豆制品...

第三章 安徽省南北分区代表性早餐案例研究

3.1 皖北地区：面食主导的北方早餐文化

皖北地区地处黄淮平原，历史上与中原文化联系紧密，其早餐文化深受北方饮食传统的影响，以面食为主导，口味偏重，讲究实惠与饱足感，呈现出一种"刚柔并济"的豪迈风格。这里的早餐不仅是果腹之物，更是一种充满烟火气的能量补给，承载着当地人披星戴月的奋斗精神。

淮南牛肉汤
淮南牛肉汤被誉为"江淮第一汤"，是皖北乃至整个安徽省最具代表性的早餐之一。这碗汤的精髓在于其浓郁的汤底和丰富的配料。制作时，需选用牛骨与牛肉，配以多种香料，经过数小时的慢火熬煮，直至汤色乳白，肉香浓郁。一碗地道的牛肉汤，通常会加入宽厚的千张丝、Q弹的粉丝、豆饼和切片的牛肉，汤头香浓，食材丰富。当地人认为，完美的吃法必须搭配刚出炉的油酥烧饼...

3.2 江淮地区：南北交融的过渡性早餐特色

江淮地区地处安徽中部，是南北文化的交汇地带，其早餐文化呈现显著的"南北通吃"和包容性。这里的早餐既有北方面食的厚重，也融入了南方的精致与细腻，形成了独特的过渡性风味。

合肥锅贴
作为省会城市，合肥的早餐选品繁多，但锅贴是最常见、最受本地人欢迎的基本款。合肥人对锅贴的要求极高，堪称"究极品鉴员"。一个地道的合肥锅贴，底儿必须酥脆，皮儿要薄而有韧性，馅儿则需多汁饱满。清晨，食客们等候在冒着热气的摊位前，等待锅盖掀开瞬间升腾的烟火气。通常，合肥人会将锅贴与一碗辣糊汤搭配食用，一口锅贴，一口热汤，轻松驱散困倦，精神抖擞地迎接新的一天...

3.3 皖南地区：徽州风味的精致早餐传统

皖南地区，其早餐文化深受徽州文化的影响，风格精致、鲜美，带有浓厚的江南风情和文人气息。这里的早餐不仅是为了填饱肚子，更是一种生活情趣和文化品味的体现。

芜湖小笼汤包
芜湖的小笼汤包是皖南早餐的标志性美食，也是城市的一张"名片"。这汤包源于开封灌汤包，经本地改良后，融入了徽菜的咸鲜与淮扬菜的精细。其特点是皮薄如纸，褶子多达28道，形如花瓣。最关键的在于其汤汁，由猪前腿肉与肉皮冻的黄金组合制成...

...

9 Conclusion

In this work, we introduced **DeepDiver-V2**, a planner-orchestrated multi-agent system designed to tackle complex information-seeking and long-form generation tasks. To enable more rigorous evaluation, we extended the WebPuzzle benchmark with more challenging and verifiable problems and introduced a new subset specifically for assessing agents' ability to produce long-form reports. To support robust and efficient training, we coupled our proposed planner-centric training algorithm with a custom in-house RL library that supports both online and offline SFT, RFT, and RL. Through extensive experiments, we showed that DeepDiver-V2 achieves competitive performance across both (1) challenging closed-ended information-seeking benchmarks and (2) open-ended long-form writing tasks. Our empirical analysis further revealed that multi-agent collaboration not only scales effectively to extremely difficult problems, but also yields emergent single-agent competence as a byproduct of collaborative training. Overall, our work demonstrates how LLM-based agents can communicate and collaborate efficiently, advancing the field of multi-agent training and offering new insights into the information-seeking capabilities of LLMs in real-world scenarios.

References

Anthropic. The "think" tool: Enabling claude to stop and think in complex tool use situations. https://www.anthropic.com/engineering/claude-think-tool, Mar. 2025. Accessed: 2025-08-29.

Y. Bai, J. Zhang, X. Lv, L. Zheng, S. Zhu, L. Hou, Y. Dong, J. Tang, and J. Li. Longwriter: Unleashing 10, 000+ word generation from long context llms. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=kQ5s9Yh0WI.

B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch. Emergent tool use from multi-agent autocurricula, 2020. URL https://arxiv.org/abs/1909.07528.

H. Chen, J. Qin, J. Guo, T. Yuan, Y. Yin, H. Zhen, Y. Wang, J. Li, X. Meng, M. Zhang, et al. Pangu light: Weight re-initialization for pruning and accelerating llms. *arXiv preprint arXiv:2505.20155*, 2025a.

H. Chen, Y. Wang, K. Han, D. Li, L. Li, Z. Bi, J. Li, H. Wang, F. Mi, M. Zhu, et al. Pangu embedded: An efficient dual-system llm reasoner with metacognition. *arXiv preprint arXiv:2505.22375*, 2025b.

D. C. Chiang and H. Lee. Can large language models be an alternative to human evaluations? In A. Rogers, J. L. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 15607–15631. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.870. URL https://doi.org/10.18653/v1/2023.acl-long.870.

DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Bao, H. Xu, H. Wang, H. Ding, H. Xin, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Wang, J. Chen, J. Yuan, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye, S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, S. Ye, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Zhao, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng,

X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y. Xu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

B. Doubao. Doubao, 2025. URL http://www.doubao.com/.

Y. Dubois, C. X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5fc47800ee5b30b8777fdd30abcaaf3b-Abstract-Conference.html.

A. Estornell, J.-F. Ton, M. F. Taufiq, and H. Li. How to train a leader: Hierarchical reasoning in multi-agent llms. *arXiv preprint arXiv:2507.08960*, 2025.

S. Gronauer and K. Diepold. Multi-agent deep reinforcement learning: a survey. *Artif. Intell. Rev.*, 55(2): 895–943, Feb. 2022. ISSN 0269-2821. doi: 10.1007/s10462-021-09996-w. URL https://doi.org/10.1007/s10462-021-09996-w.

S. Gu, T. Lillicrap, Z. Ghahramani, R. E. Turner, and S. Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic, 2017. URL https://arxiv.org/abs/1611.02247.

P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote. A survey of learning in multiagent environments: Dealing with non-stationarity. *CoRR*, abs/1707.09183, 2017. URL http://arxiv.org/abs/1707.09183.

S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, C. Wu, and J. Schmidhuber. Metagpt: Meta programming for A multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=VtmBAGCN7o.

Y. Huang, Y. Chen, H. Zhang, K. Li, M. Fang, L. Yang, X. Li, L. Shang, S. Xu, J. Hao, et al. Deep research agents: A systematic examination and roadmap. *arXiv preprint arXiv:2506.18096*, 2025.

M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, and T. Graepel. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, May 2019. ISSN 1095-9203. doi: 10.1126/science.aau6249. URL http://dx.doi.org/10.1126/science.aau6249.

Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig. Active retrieval augmented generation. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.495. URL https://doi.org/10.18653/v1/2023.emnlp-main.495.

L. Kraemer and B. Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomput.*, 190(C):82–94, May 2016. ISSN 0925-2312. doi: 10.1016/j.neucom.2016.01.031. URL https://doi.org/10.1016/j.neucom.2016.01.031.

P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem. CAMEL: communicative agents for "mind" exploration of large language model society. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA,*

December 10 - 16, 2023, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/a3621ee907def47c1b952ade25c67698-Abstract-Conference.html.

K. Li, Z. Zhang, H. Yin, L. Zhang, L. Ou, J. Wu, W. Yin, B. Li, Z. Tao, X. Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025a.

X. Li, G. Dong, J. Jin, Y. Zhang, Y. Zhou, Y. Zhu, P. Zhang, and Z. Dou. Search-o1: Agentic search-enhanced large reasoning models, 2025b. URL https://arxiv.org/abs/2501.05366.

X. Li, J. Jin, G. Dong, H. Qian, Y. Zhu, Y. Wu, J. Wen, and Z. Dou. Webthinker: Empowering large reasoning models with deep research capability. *CoRR*, abs/2504.21776, 2025c. doi: 10.48550/ARXIV.2504.21776. URL https://doi.org/10.48550/arXiv.2504.21776.

R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020. URL https://arxiv.org/abs/1706.02275.

A. Mahajan, T. Rashid, M. Samvelyan, and S. Whiteson. MAVEN: multi-agent variational exploration. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7611–7622, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/f816dc0acface7498e10496222e9db10-Abstract.html.

H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data, 2023. URL https://arxiv.org/abs/1602.05629.

M. Minsky. *The society of mind*. Simon & Schuster, Inc., USA, 1986. ISBN 0671607405.

OpenAI. Deep research system card, 2025a. URL https://cdn.openai.com/deep-research-system-card.pdf.

OpenAI. Introducing openai gpt-4.1, 2025b. URL https://openai.com/index/gpt-4-1/.

OpenAI. Introducing openai gpt-5, 2025c. URL https://openai.com/index/introducing-gpt-5/.

OpenAI. Introducing openai o3 and o4-mini, 2025d. URL https://openai.com/index/introducing-o3-and-o4-mini/.

OpenAI, :, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang. Dota 2 with large scale deep reinforcement learning, 2019. URL https://arxiv.org/abs/1912.06680.

S. J. Paech. Eq-bench: An emotional intelligence benchmark for large language models. *CoRR*, abs/2312.06281, 2023. doi: 10.48550/ARXIV.2312.06281. URL https://doi.org/10.48550/arXiv.2312.06281.

G. Papoudakis, F. Christianos, A. Rahman, and S. V. Albrecht. Dealing with non-stationarity in multi-agent deep reinforcement learning. *CoRR*, abs/1906.04737, 2019. URL http://arxiv.org/abs/1906.04737.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition, 2009. ISBN 0136042597.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

W. Shi, H. Tan, C. Kuang, X. Li, X. Ren, C. Zhang, H. Chen, Y. Wang, L. Shang, F. Yu, and Y. Wang. Pangu deepdiver: Adaptive search intensity scaling via open-web reinforcement learning, 2025. URL https://arxiv.org/abs/2505.24332.

H. Song, J. Jiang, Y. Min, J. Chen, Z. Chen, W. X. Zhao, L. Fang, and J.-R. Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.05592.

Y. Song, Y. Kim, and M. Iyyer. Veriscore: Evaluating the factuality of verifiable claims in long-form text generation. In Y. Al-Onaizan, M. Bansal, and Y. Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 9447–9474. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.FINDINGS-EMNLP.552. URL https://doi.org/10.18653/v1/2024.findings-emnlp.552.

P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. F. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In E. André, S. Koenig, M. Dastani, and G. Sukthankar, editors, *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 2085–2087. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018. URL http://dl.acm.org/citation.cfm?id=3238080.

A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente. Multiagent cooperation and competition with deep reinforcement learning, 2015. URL https://arxiv.org/abs/1511.08779.

M. Tan. Multi-agent reinforcement learning: independent versus cooperative agents. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning*, ICML'93, page 330–337, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. ISBN 1558603077.

K. Team, A. Du, B. Gao, B. Xing, C. Jiang, C. Chen, C. Li, C. Xiao, C. Du, C. Liao, C. Tang, C. Wang, D. Zhang, E. Yuan, E. Lu, F. Tang, F. Sung, G. Wei, G. Lai, H. Guo, H. Zhu, H. Ding, H. Hu, H. Yang, H. Zhang, H. Yao, H. Zhao, H. Lu, H. Li, H. Yu, H. Gao, H. Zheng, H. Yuan, J. Chen, J. Guo, J. Su, J. Wang, J. Zhao, J. Zhang, J. Liu, J. Yan, J. Wu, L. Shi, L. Ye, L. Yu, M. Dong, N. Zhang, N. Ma, Q. Pan, Q. Gong, S. Liu, S. Ma, S. Wei, S. Cao, S. Huang, T. Jiang, W. Gao, W. Xiong, W. He, W. Huang, W. Xu, W. Wu, W. He, X. Wei, X. Jia, X. Wu, X. Xu, X. Zu, X. Zhou, X. Pan, Y. Charles, Y. Li, Y. Hu, Y. Liu, Y. Chen, Y. Wang, Y. Liu, Y. Qin, Y. Liu, Y. Yang, Y. Bao, Y. Du, Y. Wu, Y. Wang, Z. Zhou, Z. Wang, Z. Li, Z. Zhu, Z. Zhang, Z. Wang, Z. Yang, Z. Huang, Z. Huang, Z. Xu, Z. Yang, and Z. Lin. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL https://arxiv.org/abs/2501.12599.

M. A. Team. Mirothinker: An open-source agentic model series trained for deep research and complex, long-horizon problem solving. https://github.com/MiroMindAI/MiroThinker, 2025a.

Q. Team. Qwq-32b: Embracing the power of reinforcement learning, 2025b. URL https://qwenlm.github.io/blog/qwq-32b/.

J. Wei, C. Yang, X. Song, Y. Lu, N. Hu, J. Huang, D. Tran, D. Peng, R. Liu, D. Huang, C. Du, and Q. V. Le. Long-form factuality in large language models. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/937ae0e83eb08d2cb8627fe1def8c751-Abstract-Conference.html.

J. Wei, Z. Sun, S. Papay, S. McKinney, J. Han, I. Fulford, H. W. Chung, A. T. Passos, W. Fedus, and A. Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL https://arxiv.org/abs/2504.12516.

T. D. Wilson. Models in information behaviour research. *Journal of Documentation*, 55(3):249–270, 1999. doi: 10.1108/EUM0000000007145. URL https://doi.org/10.1108/EUM0000000007145.

M. Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009. ISBN 0470519460.

J. Wu, B. Li, R. Fang, W. Yin, L. Zhang, Z. Tao, D. Zhang, Z. Xi, Y. Jiang, P. Xie, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025a.

Y. Wu, J. Mei, M. Yan, C. Li, S. Lai, Y. Ren, Z. Wang, J. Zhang, M. Wu, Q. Jin, and F. Huang. Writingbench: A comprehensive benchmark for generative writing. *CoRR*, abs/2503.05244, 2025b. doi: 10.48550/ARXIV.2503.05244. URL https://doi.org/10.48550/arXiv.2503.05244.

xAI. Grok 3 beta — the age of reasoning agents, 2025. URL https://x.ai/news/grok-3.

Xbench-Team. Xbench-deepsearch, 2025. URL https://xbench.org/agi/aisearch.

A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

A. Yang, B. Zhang, B. Hui, B. Gao, B. Yu, C. Li, D. Liu, J. Tu, J. Zhou, J. Lin, K. Lu, M. Xue, R. Lin, T. Liu, X. Ren, and Z. Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024b. URL https://arxiv.org/abs/2409.12122.

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations,*

ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

T. Yu, S. Zhang, and Y. Feng. Auto-rag: Autonomous retrieval-augmented generation for large language models, 2024. URL https://arxiv.org/abs/2411.19443.

L. Zheng, W. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Y. Zheng, D. Fu, X. Hu, X. Cai, L. Ye, P. Lu, and P. Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

P. Zhou, B. Leon, X. Ying, C. Zhang, Y. Shao, Q. Ye, D. Chong, Z. Jin, C. Xie, M. Cao, Y. Gu, S. Hong, J. Ren, J. Chen, C. Liu, and Y. Hua. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese, 2025. URL https://arxiv.org/abs/2504.19314.

Appendix

Discussion: Limitations and Extensions

Balance between Efficiency and Accuracy Despite the strong results on BrowseComp and the demonstrated quality of long-form report generation, DeepDiver-V2 incurs higher computational overhead than DeepDiver-V1. In particular, it requires more time for reasoning, evidence gathering, and verification, as well as increased tool-call usage. The inference time and number of rounds in DeepDiver-V2 adapt to problem difficulty, but can grow exponentially on extremely challenging tasks. While DeepDiver-V1 introduced search intensity scaling (SIS), DeepDiver-V2 magnifies this mechanism significantly, yielding more substantial performance gains but at the cost of greater inference time and higher tool-call expenditure. Such limitations constrain DeepDiver-V2's scalability, cost-efficiency, the practical applicability in real-world deployments, and the online RL training at scale.

Toolkits and Scaffold Support DeepDiver-V2 provides rich tool support compared to DeepDiver-V1, including local file operations, open-web search, and content generation. However, effective information-seeking agents often require access to additional tools such as web browsing, semantic search, sandboxed deployment, and code execution. These tools and corresponding capabilities enable more advanced verification and exploration within the agent framework. Future versions of DeepDiver should therefore aim to incorporate broader tool integration and scaffold support, enhancing generalization and strengthening the framework's ability to tackle diverse real-world tasks.