OpenSearch-VL: An Open Recipe for Frontier Multimodal Search Agents

Shuang Chen[1,2], **Kaituo Feng**[3], **Hangting Chen**[1], **Wenxuan Huang**[3], **Dasen Dai**[3]
Quanxin Shou[2,4], **Yunlong Lin**[3], **Xiangyu Yue**[3], **Shenghua Gao**[4], **Tianyu Pang**[1,†]

[1]Tencent Hunyuan [2]University of California, Los Angeles
[3]The Chinese University of Hong Kong [4]The University of Hong Kong
[†]Corresponding author.

Abstract

Deep search has become a crucial capability for frontier multimodal agents, enabling models to solve complex questions through active search, evidence verification, and multi-step reasoning. Despite rapid progress, top-tier multimodal search agents remain difficult to reproduce, largely due to the absence of open high-quality training data, transparent trajectory synthesis pipelines, or detailed training recipes. To this end, we introduce **OpenSearch-VL**, a fully open-source recipe for training frontier multimodal deep search agents with agentic reinforcement learning. First, we curated a dedicated pipeline to construct high-quality training data through Wikipedia path sampling, fuzzy entity rewriting, and source-anchor visual grounding, which jointly reduce shortcuts and one-step retrieval collapse. Based on this pipeline, we curate two training datasets, *SearchVL-SFT-36k* for SFT and *SearchVL-RL-8k* for RL. Besides, we design a diverse tool environment that unifies text search, image search, OCR, cropping, sharpening, super-resolution, and perspective correction, enabling agents to combine active perception with external knowledge acquisition. Finally, we propose a multi-turn fatal-aware GRPO training algorithm that handles cascading tool failures by masking post-failure tokens while preserving useful pre-failure reasoning through one-sided advantage clamping. Built on this recipe, OpenSearch-VL delivers substantial performance gains, with over 10-point average improvements across seven benchmarks, and achieves results comparable to proprietary commercial models on several tasks. We will release all data, code, and models to support open research on multimodal deep search agents.

1 Introduction

Multimodal deep search has emerged as a critical direction for multimodal large language models (MLLMs), enabling them to evolve from passive visual understanding systems into agents that actively search evidence, verify facts, and reason over knowledge-intensive visual queries [1, 2, 3]. However, frontier multimodal search agents remain difficult to reproduce, as their training data, code are often proprietary or insufficiently disclosed [4, 1, 5, 6]. As a result, the community still lacks a fully open recipe for building, analyzing, and improving strong multimodal search agents.

Among these missing components, high-quality training data is a central bottleneck. The strongest frontier systems are still largely dominated by well-funded commercial corporations [7, 8], where the data sources, filtering criteria, expert demonstrations, and tool-use trajectories are typically kept private. This makes it difficult to reproduce advanced multimodal search capabilities or systematically study which data properties are essential for agentic search behavior. The issue is even more pronounced in multimodal settings, where effective training data must capture image-grounded understanding, multi-hop retrieval, evidence verification, and long-horizon tool use rather than simple

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Table 1: The search-oriented tool suite integrated within **OpenSearch-VL**. The suite spans three complementary functions—*retrieval* for acquiring external information, *image enhancement* for improving low-quality visual inputs, and *attention & parsing* for focusing on and extracting content from specific regions. Detailed specifications of each tool are provided in Appendix G.

Tool	Description	Arguments	Tool Output
TEXTSEARCH	Web search with page reading and LLM summarization	Query + TopK	Query-focused passage summaries
IMAGESEARCH	Reverse image / visual entity search over the web	Image + TopK	Visual matches and related webpages
SHARPEN	Unsharp-masking based deblurring / detail enhancement	Image + Amount	Sharpened image
SUPERRESOLUTION	Deep super-resolution (EDSR) for low-resolution inputs	Image + Scale	High-resolution image
PERSPECTIVECORRECT	Auto perspective rectification of skewed documents	Image	Fronto-parallel image
CROP	Extract a user-specified rectangular region	Image + Coordinates	Cropped image
OCR	Structured document parsing with text and layout labels	Image + Flags	Text blocks with labels and reading order

visual question answering. Therefore, releasing high-quality training data is crucial for making frontier multimodal search agent research more transparent, reproducible, and accessible.

Beyond data, training multimodal search agents also poses unique challenges, especially when applying agentic reinforcement learning (agentic RL) [9, 10, 1] to long-horizon tool-use settings. Agentic search trajectories involve multiple rounds of reasoning, tool invocation, and observation integration, where a single malformed call, timeout, irrelevant query, or repeated failure can invalidate the remaining rollout. Simply discarding such trajectories wastes useful pre-failure reasoning, while training on the full rollout introduces noisy gradients from meaningless post-failure tokens. Another practical challenge is that real-world visual inputs are often imperfect, such as blurred photos, low-resolution thumbnails, skewed documents, and crowded screenshots. In these cases, searching alone is insufficient, and the agent must first crop, enhance, rectify, or parse the visual evidence before reliable search can begin. However, most existing multimodal search agents focus mainly on retrieval and do not jointly address robust visual pre-processing and failure-aware long-horizon RL.

In this work, we introduce **OpenSearch-VL**, a fully open recipe for training frontier multimodal deep search agents with agentic RL. Our recipe addresses the above challenges from data, tools, and training. First, we develop a dedicated data curation pipeline to build high-quality training data. Starting from the Wikipedia hyperlink graph, we sample multi-hop entity paths and convert them into multi-hop VQA instances by rewriting intermediate entities into fuzzy descriptions, followed by a carefully designed filtering mechanism. This design avoids single-hop image lookup shortcuts and encourages the agent to learn multi-hop search and reasoning behaviors. This pipeline yields two training datasets **SearchVL-SFT-36k** for SFT and **SearchVL-RL-8k** for agentic RL. Second, we build a tool environment that goes beyond retrieval-only multimodal agent. In addition to search, the agent is equipped with OCR, cropping, sharpening, super-resolution, and perspective correction, allowing it to handle imperfect visual inputs in real-world scenarios before querying external knowledge. Finally, we develop an agentic RL algorithm based on GRPO [11] for long-horizon multimodal tool use, where multi-step interactions often lead to cascading tool failures. To address this issue, we introduce fatal-aware token masking that removes invalid post-failure suffixes from optimization, while preserving useful pre-failure reasoning through one-sided advantage clamping. This enables the model to learn from partially successful trajectories without being affected by noisy gradients from failed rollouts. Together, these designs enable OpenSearch-VL to learn robust long-horizon search behavior over multimodal evidence in real-world scenarios.

Experiments across multimodal deep search benchmarks show that OpenSearch-VL consistently improves over strong baselines. For example, compared with the Qwen3-VL-30B-A3B [12] agentic baseline, our model improves the average score from 47.8 to 61.6, with large gains on VDR (+13.3) [13], MMSearch (+24.5) [14], FVQA (+10.2) [15], and InfoSeek (+16.2) [16]. Moreover, OpenSearch-VL achieves comparable or even better performance than proprietary commercial models on several benchmarks. In summary, our main contributions can be summarized as follows:

- We introduce **OpenSearch-VL**, a fully open recipe for training frontier multimodal deep search agents. We will release the training data, code, and models to provide an open foundation for reproducible research on multimodal agentic search.
- We build the key components required for training advanced multimodal search agents, including high-quality image-grounded multi-hop training data, a diverse tool environment, and a multi-turn fatal-aware GRPO algorithm.

- Extensive experiments demonstrate the effectiveness of our recipe. For example, our trained OpenSearch-VL-30B-A3B brings an average improvement of 13.8 points across 7 multimodal deep search benchmarks.

2 Preliminaries

Problem Formulation. Given an input image I_0 and a question q, the agent answers q by interleaving reasoning with tool calls over a diverse tool set $\mathcal{T} = \mathcal{T}_v \cup \mathcal{T}_s$, where $\mathcal{T}_v$ contains *visual* tools that transform or parse images and $\mathcal{T}_s$ contains *retrieval* tools that query external knowledge. At step l, the model conditions on the accumulated history

$$h_l = \big(\mathcal{I}_l,\ q,\ \mathbf{a}_{<l},\ \mathbf{o}_{<l}\big), \tag{1}$$

where $\mathcal{I}_l$, $\mathbf{a}_{<l}$, and $\mathbf{o}_{<l}$ denote the images, actions, and observations accumulated up to step l. The interaction unfolds as a multi-turn trajectory

$$\tau = \big\{(h_0, a_0, o_0),\ (h_1, a_1, o_1),\ \ldots,\ (h_{L-1}, a_{L-1}, o_{L-1}),\ (h_L, a_L)\big\}, \tag{2}$$

where the final step emits the answer without a subsequent observation. Following the ReAct [17] think-then-act convention, each action decomposes as $a_l = [z_l,\ c_l]$, where z_l is a reasoning trace, and c_l denotes a tool invocation for $l < L$ or the final response for $l = L$.

Unlike text-only formulations [18], our environment $\mathcal{E}$ returns *multimodal* observations: $\mathcal{T}_s$ yields textual passages and most $\mathcal{T}_v$ tools yield images that grow the visual context $\mathcal{I}_l = \{I_0\} \cup \{o_k : k < l,\ o_k \text{ is an image}\}$. The rollout is compactly written as $\tau \sim \pi_\theta(\cdot \mid I_0, q) \otimes \mathcal{E}$; a full formalisation is deferred to Appendix D.

Search Tools. *OpenSearch-VL* is equipped with a suite of tools covering three complementary functions: **retrieval** (TEXTSEARCH, IMAGESEARCH) for gathering external evidence, **image enhancement** (SHARPEN, SUPERRESOLUTION, PERSPECTIVECORRECT) for remedying low-quality inputs, and **attention and parsing** (CROP, OCR) for localizing and decoding fine-grained content. The suite combines lightweight offline primitives with online services backed by expert models, and is summarized in Table 1. Full specifications are deferred to Appendix G.

3 Dataset Curation

To equip the model with robust reasoning and tool-use capabilities, we design a scalable data curation pipeline (Figure 1) that synthesizes high-quality trajectories without manual human annotation. The pipeline proceeds in three stages—VQA construction, staged filtering and enhancement, and trajectory synthesis—yielding the final dataset used for the following stage training.

3.1 High-Quality VQA Construction

A central challenge for training multimodal search agents is the supply of questions that encourage non-trivial use of the diverse tool set $\mathcal{T}$. Directly prompting a VLM on an image tends to yield shallow, perception-level queries that can be resolved in a single forward pass [10, 1]. Building on this observation, we adopt a unified construction pipeline: we sample multi-hop trajectories over the Wikipedia hyperlink graph, synthesize textual QA pairs along each trajectory, and lift them into image-grounded VQA via answer-preserving fuzzy rewriting and source-anchored visual grounding. Compared with prior QA constructions [19, 20, 10], our pipeline (i) assigns each node on the sampled path an explicit functional role within the reasoning chain, and (ii) deliberately decouples the visual anchor from the answer entity, thereby suppressing single-shot retrieval shortcuts.

Wikipedia Path Sampling. We cast the Wikipedia [21] as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with articles as nodes and in-article hyperlinks as edges. From a seed $v_0 \in \mathcal{V}$, a constrained random walk of length $h \in \{2, 3, 4\}$ produces a path

$$P \ = \ \big(v_0 \xrightarrow{\rho_1} v_1 \xrightarrow{\rho_2} \cdots \xrightarrow{\rho_h} v_h\big), \tag{3}$$

where each relation ρ_j is induced by the hyperlink's anchor text. The walk skips (i) disambiguation and list pages, (ii) cycles, and (iii) hub nodes whose in-degree exceeds a threshold τ_{hub}; full thresholds, resampling heuristics, and additional filters are deferred to Appendix E.2. Each node on P is assigned a functional role: v_0 is the **anchor** (visual entry point, to be replaced by a visual referring expression), $v_1, \ldots, v_{h-1}$ are **bridge** nodes (intermediate entities with fuzzified names), and v_h is the **answer** node (source of the target attribute). These roles govern the rewriting and grounding stages below.

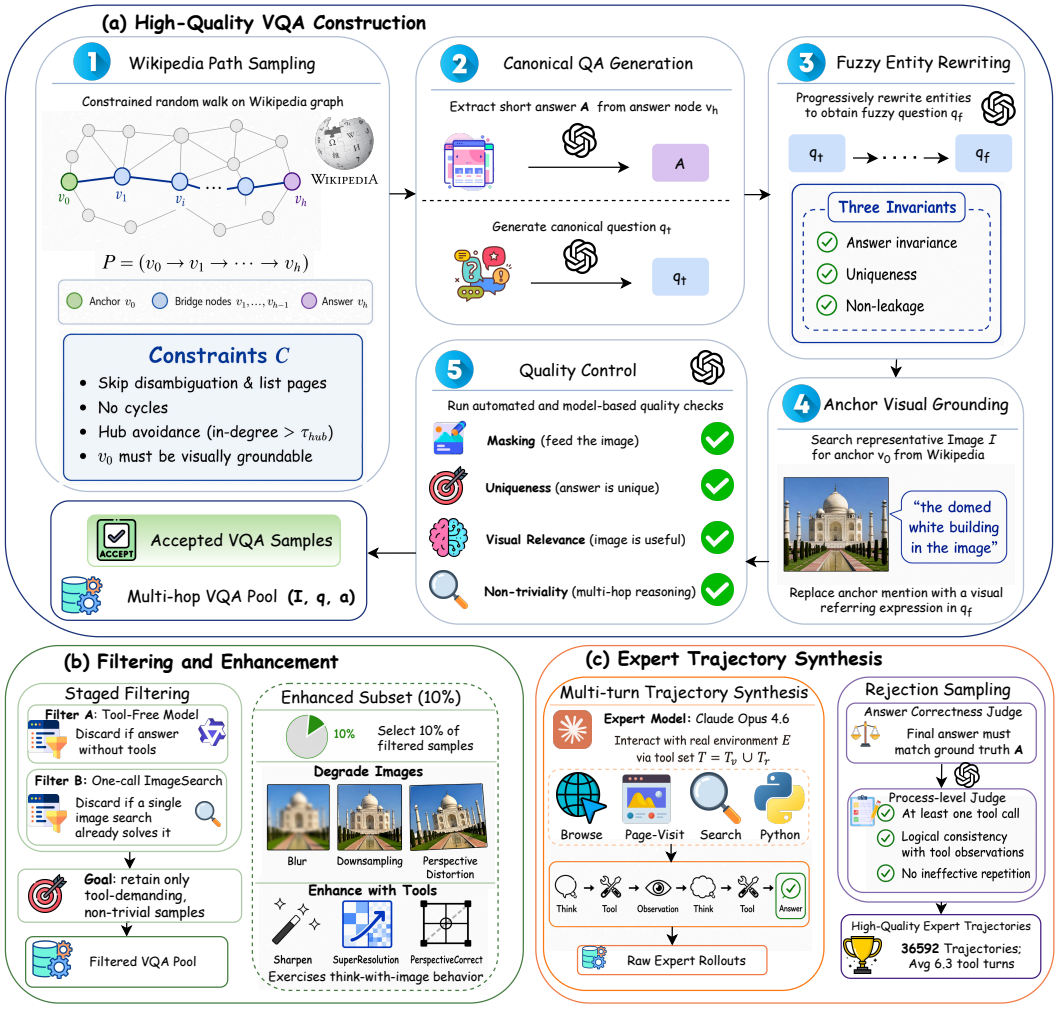

Figure 1: **Overview of the data curation pipeline.** (a) Starting from the English Wikipedia hyperlink graph, we construct high-quality multi-hop VQA instances by sampling constrained paths, generating canonical question–answer pairs, rewriting them into fuzzy questions, grounding anchor entities with representative images, and applying automated quality control. (b) We then perform staged filtering to retain only tool-demanding, non-trivial samples, and create an enhanced subset through image degradation and tool-based restoration to encourage think-with-image behavior. (c) Finally, we synthesize multi-turn expert trajectories in a real tool environment and apply rejection sampling with answer-correctness and process-level judges, yielding the final high-quality trajectories used for the following stage training.

We extract a short, unambiguous answer a from v_h and prompt GPT-4o [22] to synthesize a *canonical* question q_t that verbalizes P and references v_h only through the queried attribute (extraction details in Appendix E.2). The canonical q_t is not a training target but a manipulable object for rewriting.

Fuzzy Entity Rewriting. Preserving entity names in q_t enables the agent to short-circuit the chain with a single retrieval [20, 1]. We therefore progressively rewrite q_t into a fuzzy counterpart q_f while fixing a. Following the iterative style of Skywork-R1V4 [23], we rewrite *one entity at a time*, from the farthest bridge v_{h-1} toward v_0: each name is replaced by a relational or attribute-based descriptor drawn from the entity's Wikipedia context, and an LLM uniqueness evaluator verifies that the substitution still resolves to the intended entity conditional on the partially rewritten question. A

rewrite is accepted only when

$$\underbrace{a(q_f) = a(q_t)}_{\text{answer invariance}}, \qquad \underbrace{|\mathcal{R}(q_f)| = 1}_{\text{uniqueness}}, \qquad \underbrace{\left(\bigcup_{j=0}^{h} \text{aliases}(v_j)\right) \cap q_f = \emptyset}_{\text{non-leakage}}, \tag{4}$$

where $\mathcal{R}(q_f)$ denotes the set of entities compatible with q_f under the evaluator's world knowledge. We further interleave entity rewriting with occasional *answer obfuscation* [1] to avoid collapsing onto a stereotyped relational template.

Anchor-aware Visual Grounding. We retrieve a representative image I of the anchor v_0 from Wikimedia Commons or its Wikipedia infobox, filter candidates by CLIP similarity to a short textual description of v_0, and replace v_0 in q_f with a visual referring expression (e.g., *"the person in the image"*) to yield the final question q. Unlike prior QA-to-VQA conversions [10, 23] that ground on or near the answer entity, anchoring v_0 at the *source* of P substantially reduces single-hop shortcuts: the agent must first identify the visual anchor and then follow the intermediate textual relations before reaching a.

Each candidate triple (I, q, a) is gated by automatic checks for masking, uniqueness, and visual relevance, generalizing the selector/examiner protocol of WebWatcher [10] (full criteria in Appendix E.2); non-triviality is handled jointly with the staged filtering of Sec. 3.2. Instances passing these checks form the Wikipedia portion of our VQA pool, subsequently merged with open-source multimodal corpora before trajectory synthesis.

3.2 Filtering and Enhancement

Before trajectory synthesis, we consolidate the Wikipedia-derived VQA instances from Sec. 3.1 with three open-source multimodal corpora—LiveVQA [24], FVQA [15], and WebQA [25]—to broaden coverage across live entities, commonsense fact lookup, and open-web multi-hop reasoning. We then apply a two-stage difficulty filter using a frozen `Qwen3-VL-32B` [12]: first discarding examples answerable without tools, and then discarding examples solvable with a single IMAGESEARCH call. This removes samples that rely only on parametric knowledge, perceptual shortcuts, answer-coincident anchors, or one-hop bridge leakage, ensuring that retained instances genuinely require the intended visual-to-text search chain.

To further expose the agent to realistic visual imperfections, we randomly select 10% of the filtered VQA pool and apply controlled degradations—blur, downsampling, and perspective distortion—paired with the corresponding enhancement tools in $\mathcal{T}_v$ (SHARPEN, SUPERRESOLUTION, and PERSPECTIVECORRECT). This enhancement subset diversifies the training distribution and induces a *think-with-image* behavior: when the input image is unreliable, the policy learns to repair the visual evidence before initiating retrieval. Together, the filtered retrieval-heavy instances and the enhancement-required subset exercise both visual restoration and evidence acquisition within the unified tool environment.

3.3 Multi-turn Trajectory Synthesis

For each instance (I, q, a) that survives the filters of Sec. 3.2, we synthesize expert trajectories by rolling out `Claude Opus 4.6` [26] as the expert model against the real execution environment $\mathcal{E}$, prompted with the agent system prompt of Appendix F and free to invoke any tool in $\mathcal{T}$. We draw $K = 5$ independent rollouts per instance, each formatted as a multi-turn ReAct [17] trajectory aligned with Eq. 2. Then the raw rollouts are passed through a two-stage rejection cascade. The first stage discards any trajectory whose final answer disagrees with the ground truth a (adjudicated by the same LLM-as-judge [27] using `GPT-4o` [22], Appendix B.1). The surviving trajectories are then vetted by a `GPT-5.4` process-level judge on tool-use, logical consistency between reasoning and observations, and absence of ineffective repetition (full criteria in Appendix B.1).

Applying both stages to the full rollout corpus yields $\mathbf{36{,}592}$ high-quality expert trajectories with an average of $\mathbf{6.3}$ tool-invocation turns per trajectory, which together constitute the SFT corpus consumed in Sec. 4.1.

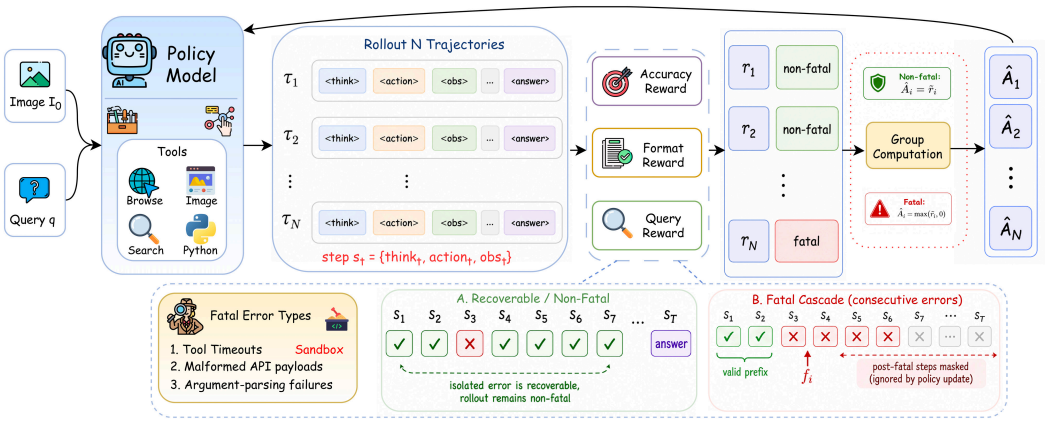

Figure 2: **Overview of the RL training pipeline.** Starting from a supervised fine-tuned model, we sample a group of multi-turn trajectories against the real environment $\mathcal{E}$. Each trajectory is evaluated by a composite reward combining final-task success (r_{acc}) and process-level search quality (r_{query}) along with a format check (r_{fmt}). To preserve valid reasoning in trajectories that eventually encounter fatal errors, we apply fatal-aware token masking to truncate the sequence and employ one-sided advantage clamping during policy optimization, preventing the suppression of viable early steps.

4 Training

We train OpenSearch-VL in two sequential stages. First, we perform supervised fine-tuning (SFT) to instill fundamental reasoning and tool-use behaviors; subsequently, we apply reinforcement learning (RL) via a multi-turn, search-augmented objective (Figure 2) to discover more effective exploration strategies.

4.1 Supervised Fine-Tuning

We perform SFT on a curated set of 36592 multi-turn expert trajectories (Section 3). Using the history h_l (Eq. 1) and the action decomposition $a_l = [z_l, c_l]$, by autoregressive factorisation the step-level action probability decomposes as $P_\theta(a_l \mid h_l) = P_\theta(z_l \mid h_l)\, P_\theta(c_l \mid h_l, z_l)$. Summing over all trajectories $i \in \{1, \ldots, N\}$ and steps $l \in \{1, \ldots, L_i\}$, the standard SFT objective can be equivalently written as

$$\max_\theta \sum_{i=1}^{N} \sum_{l=1}^{L_i} \left[\log P_\theta \left(z_l^{(i)} \mid h_l^{(i)} \right) + \log P_\theta \left(c_l^{(i)} \mid h_l^{(i)}, z_l^{(i)} \right) \right], \tag{5}$$

where tool observations o_l enter only as conditioning context and are excluded from the loss computation following the retrieved-token masking strategy of [18]. This provides a structured interpretation of the training signal, showing that it jointly supervises both the reasoning trace and the subsequent tool invocation (or terminal response) at each step.

4.2 Multi-Turn Search Fatal-Aware GRPO

While SFT provides a strong initialization for tool use, it remains bounded by the coverage of the demonstration trajectories and therefore cannot discover improved search strategies through exploration. We address this limitation with reinforcement learning, building on GRPO [28] and its search-augmented extension [18]. Our setting, however, differs from prior search-only formulations in three important respects: we optimize over a multimodal environment $\mathcal{E}$ with diverse tools rather than a text-only retriever $\mathcal{R}$; we use a composite reward that combines final-task success with process-level search quality; and we introduce fatal-aware masking together with one-sided advantage clamping to preserve useful supervision from partially successful trajectories.

During training, for each prompt (I_0, q) we sample a group of G multi-turn rollouts $\tau_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid I_0, q) \otimes \mathcal{E}$, for $i = 1, \ldots, G$.

Composite Multi-Turn Reward. Long-horizon tasks pose a sparse-reward challenge: outcome-only rewards miss credit for partially successful reasoning, while process-only rewards risk misalignment from the end goal. We therefore use a composite trajectory-level reward

$$r(\tau) \;=\; r_{\text{fmt}}(\tau) \;\cdot\; \big[\,\alpha\, r_{\text{acc}}(\tau) + (1-\alpha)\, r_{\text{query}}(\tau)\,\big] \tag{6}$$

where we set $\alpha = 0.8$ and combine three dimensions:

- **Format** $r_{\text{fmt}} \in [0,1]$: step-wise average of structural checks on `<think>`/action blocks, acting as a multiplicative gate against ill-formatted outputs.
- **Accuracy** $r_{\text{acc}} \in \{0,1\}$: We use `GPT-4o` [22] under an LLM-as-Judge protocol [27] to verify whether the final answer entails the ground truth, assigning zero reward to invalid responses.
- **Query Quality** $r_{\text{query}} \in [0,1]$: We use `GPT-5.4` [29], a proprietary frontier reasoning model, as the query-quality judge to score the relevance, diversity, and logical progression of issued search queries, providing dense learning feedback for unsuccessful trajectories when $r_{\text{acc}} = 0$.

Fatal-Aware Token Masking. In unconstrained multi-turn environments, agents frequently encounter "fatal" states—such as cascading tool-execution failures or infinite loops—after which subsequent reasoning becomes meaningless. Standard approaches either discard the entire trajectory [1] (wasting the valid early steps) or train on it blindly (injecting noise). We introduce a *fatal-aware* masking strategy to preserve the viable prefix.

We define the *fatal step index* f_i for trajectory τ_i as the earliest step where $K = 3$ consecutive tool-execution errors commence, with $f_i = L_i + 1$ if no such cascade occurs. We then extend the observation-token (generation) mask $M_{\text{gen}}(y_{i,t})$ to additionally zero out all tokens generated *after* the fatal step:

$$M(y_{i,t}) \;=\; M_{\text{gen}}(y_{i,t}) \;\cdot\; \mathbb{1}\big[\,s(t) < f_i\,\big], \tag{7}$$

where $s(t)$ maps token index t to its step index l. Crucially, the process rewards r_{fmt} and r_{query} are computed exclusively over the valid prefix $l < f_i$, ensuring the model is not penalised for structural collapse that occurs after the trajectory is deemed fatal.

All G trajectories—including fatal ones—contribute to the standard group-normalised reward $\widetilde{r}_i = (r(\tau_i) - \text{mean})/(\text{std} + \delta)$ to keep group statistics unbiased. However, directly using $\widetilde{r}_i$ for fatal trajectories is pathological: a sub-mean $\widetilde{r}_i$ would push the policy gradient to suppress the *viable prefix*, discouraging the valid reasoning before the error cascade. We therefore apply one-sided advantage clamping:

$$\hat{A}_i \;=\; \begin{cases} \widetilde{r}_i & \text{if } f_i = L_i + 1 \text{ (non-fatal)}, \\ \max(\widetilde{r}_i,\; 0) & \text{if } f_i \le L_i \text{ (fatal)}. \end{cases} \tag{8}$$

This clamping ensures that the valid prefix of a fatal trajectory is only ever *reinforced* if its partial reward exceeds the group mean, and otherwise receives zero gradient rather than an undeserved penalty. In this sense, it generalizes the hard-masking baseline [1] while recovering strictly more useful learning signal.

Integrating the fatal-aware mask $M_{i,t} \equiv M(y_{i,t})$ and the clamped advantage $\hat{A}_i$, our final GRPO objective over the multimodal environment $\mathcal{E}$ is formulated as:

$$\mathcal{J}(\theta) = \mathbb{E}_{\substack{(I_0,q)\sim\mathcal{D} \\ \{\tau_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|I_0,q;\,\mathcal{E})}} \left[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{\sum_t M_{i,t}} \sum_{t=1}^{|\tau_i|} M_{i,t} \min\Big(\rho_{i,t}(\theta)\hat{A}_i,\, \text{clip}_{1-\epsilon}^{1+\epsilon}(\rho_{i,t}(\theta))\,\hat{A}_i\Big) \right]. \tag{9}$$

where $\rho_{i,t}(\theta) = \pi_\theta(y_{i,t} \mid I_0, q, y_{i,<t}; \mathcal{E}) \,/\, \pi_{\theta_{\text{old}}}(y_{i,t} \mid I_0, q, y_{i,<t}; \mathcal{E})$ is the token-level importance ratio, and the standard $\beta\, D_{\text{KL}}[\pi_\theta \| \pi_{\text{ref}}]$ term is omitted from the display since it is identical to that of standard GRPO (Eq. 11). Relative to search-augmented GRPO [18], the differences (color-matched to Eq. 9) are threefold: the execution environment is generalized from $\mathcal{R}$ to $\mathcal{E}$, the generation mask M_{gen} is extended to the fatal-aware mask M, and the advantage is computed from the composite reward with one-sided clamping. Additional details and derivations are provided in Appendix B.

5 Experiments

We first describe our experimental setups as follows:

Models and Benchmarks. Our OpenSearch-VL is built on three Qwen3-VL variants [12]: *Qwen3-VL-8B-Instruct*, *Qwen3-VL-30B-A3B-Instruct*, and *Qwen3-VL-32B-Instruct*. For evaluation,

Table 2: Performance on multimodal knowledge-intensive QA and web-search benchmarks. **Bold** and underline mark the best and second-best score in each column.

Model	SimpleVQA	VDR	MMSearch	LiveVQA	BrowseComp-VL	FVQA	InfoSeek	Avg
Direct Reasoning								
GPT-4o [22]	51.7	1.7	18.7	28.1	5.5	48.0	52.9	29.5
GPT-5 [29]	61.6	**9.8**	35.1	44.4	**48.6**	54.4	**61.7**	45.1
Gemini-2.5-Flash [8]	57.9	6.2	30.4	51.0	37.1	47.7	44.1	39.2
Gemini-2.5-Pro [8]	**63.0**	8.0	**39.8**	**60.3**	43.1	**60.7**	46.9	**46.0**
Claude-4-Sonnet [7]	50.9	2.0	18.7	38.5	29.3	35.3	57.3	33.1
Claude-3.7-Sonnet [30]	42.7	4.6	21.1	38.0	32.3	36.7	54.8	32.9
Qwen3-VL-8B [12]	47.1	2.8	11.7	23.1	24.1	24.2	23.1	22.3
Qwen3-VL-30B-A3B [12]	53.2	3.8	18.7	42.7	29.6	34.7	26.4	29.9
Qwen3-VL-32B [12]	58.0	4.1	19.8	45.5	30.8	34.1	28.8	31.6
RAG Workflow								
GPT-4o [22]	**63.6**	4.5	49.1	40.1	13.4	**66.3**	59.5	42.4
GPT-5 [29]	55.9	**22.3**	**52.6**	**56.0**	**54.9**	62.6	**70.6**	**53.6**
Claude-3.7-Sonnet [30]	59.3	11.3	32.7	30.3	10.0	59.1	60.2	37.6
Qwen3-VL-8B [12]	62.3	7.3	47.3	39.3	29.3	53.6	46.1	40.7
Agentic Workflow								
DeepMMSearch-R1-7B [31]	55.8	–	–	–	–	–	47.5	–
Visual-ARFT-7B [32]	42.4	3.3	34.5	25.4	16.5	41.7	37.9	28.8
MMSearch-R1-7B[33]	57.4	2.9	53.8	48.4	20.9	58.4	55.1	42.4
DeepEyes-v2-7B [34]	59.4	7.8	63.7	–	–	60.6	51.1	–
WebWatcher-7B [10]	54.3	10.3	49.1	51.2	21.2	–	–	–
Qwen3-VL-8B [12]	52.0	17.0	37.4	50.6	27.9	58.7	50.3	42.0
SenseNova-MARS-8B [35]	61.7	19.4	**67.4**	56.2	35.1	67.1	61.7	52.7
OpenSearch-VL-8B(Ours)	**71.6**	**20.8**	64.5	**59.6**	**37.6**	**71.5**	**70.2**	**56.6**
Qwen3-VL-30B-A3B [12]	55.1	20.2	44.2	62.0	34.1	63.0	56.2	47.8
OpenSearch-VL-30B-A3B(Ours)	**74.9**	**33.5**	**68.7**	**67.4**	**41.1**	**73.2**	**72.4**	**61.6**
Qwen3-VL-32B [12]	58.7	23.1	53.9	45.5	35.1	61.2	58.5	48.0
WebWatcher-32B [10]	59.0	–	55.3	58.7	27.0	–	–	–
OpenSearch-VL-32B(Ours)	**76.2**	**33.8**	**72.3**	**70.5**	**43.8**	**74.7**	**74.8**	**63.7**

Table 3: **RL recipe ablation**, with deltas measured relative to Vanilla GRPO [18].

Method	SimpleVQA	InfoSeek	FVQA	Avg.
Qwen3-VL-8B	52.0	50.3	58.7	53.7
Qwen3-VL-8B + SFT only	66.1	62.4	65.3	64.6
+ Vanilla GRPO	68.8	66.5	67.4	67.6
+ GRPO w/ Hard Masking	68.3(-0.5)	67.9(+1.4)	66.9(-0.5)	67.7(+0.1)
+ GRPO w/ Fatal Masking only	69.7(+0.9)	68.3(+1.8)	69.2(+1.8)	69.1(+1.5)
+ Fatal Masking + One-sided Clamp **(Ours)**	**71.6**(+2.8)	**72.4**(+5.9)	**71.5**(+4.1)	**71.8**(+4.2)

we use seven knowledge-intensive benchmarks from our main results: SimpleVQA [36], VDR [13], MMSearch [14], LiveVQA [24], BrowseComp-VL [10], FVQA [15], and InfoSeek [16]. Together, they cover visual entity recognition, web evidence retrieval, multi-hop reasoning, and long-tail QA.

Baselines and Evaluation Metrics. We evaluate OpenSearch-VL against baseline types: Direct Reasoning, where the model answers from its parametric knowledge and visual perception alone; RAG Workflow, where external retrieval results are provided in-context but the reasoning remains single-pass; and Agentic Workflow, where the model autonomously interleaves reasoning with tool calls in a multimodal environment. We report Pass@1 on all seven benchmarks. Correctness is adjudicated by a *GPT-4o* judge that compares the model's final response with the reference answer. To ensure fair comparison across heterogeneous answer styles, we adopt the same evaluation protocol as VDR-Bench [13]; the full judge prompt is provided in Figure 10.

Datasets. Our SFT data consists of the **36K** multi-turn trajectories synthesized by the procedure in Sec. 3.3. For RL training, we randomly sample **8K** examples from the VQA pool after the staged filtering and enhancement process in Sec. 3.2, ensuring that these examples are disjoint from the VQA instances used to synthesize the SFT trajectories.

Implementation Details. OpenSearch-VL extends LlamaFactory [37] for agentic SFT and builds on rLLM [38] and VDR [1] for multi-turn tool-interleaved RL. All models are trained on Nvidia H20 GPUs, with detailed training configurations and hyperparameters reported in Appendix C.

Table 4: **SFT data pipeline ablation**, with deltas measured relative to the Full pipeline.

Settings	SimpleVQA	InfoSeek	FVQA	Avg.
Full pipeline (**Ours**)	**66.1**	**62.4**	**65.3**	**64.6**
w/o source-anchor grounding	53.6(-12.5)	54.5(-7.9)	51.2(-14.1)	53.1(-11.5)
w/o fuzzy entity rewriting	51.7(-14.4)	56.4(-6.0)	54.7(-10.6)	54.3(-10.3)
w/o staged filtering	57.6(-8.5)	55.2(-7.2)	56.3(-9.0)	56.4(-8.2)
w/o enhancement subset	64.9(-1.2)	61.7(-0.7)	63.2(-2.1)	63.3(-1.3)

5.1 Main Results

Table 2 reports the results on seven multimodal knowledge-intensive QA and web-search benchmarks. OPENSEARCH-VL exhibits a clear advantage over both direct-reasoning and RAG baselines across all scales, underscoring the necessity of an agentic loop for complex multimodal queries. Among 8B-scale agents, *OpenSearch-VL-8B* achieves the best average score of **56.6**, surpassing the previous strongest open 8B agent, *SenseNova-MARS-8B*, by **3.9** points on average. At larger scales, *OpenSearch-VL-30B-A3B* and *OpenSearch-VL-32B* further improve the average score to **61.6** and **63.7**, respectively; notably, *OpenSearch-VL-32B* outperforms strong proprietary direct-reasoning models such as *Gemini-2.5-Pro* and substantially exceeds the corresponding *Qwen3-VL* agentic baselines. These results demonstrate that our training recipe scales effectively from 8B to 32B and yields strong gains on both search-heavy and visually grounded benchmarks.

5.2 Ablation Study

We conduct ablations on the Qwen3-VL-8B model to validate the two design choices that define OPENSEARCH-VL: the data synthesis pipeline that produces tool-demanding multimodal trajectories, and the fatal-aware RL recipe that improves the policy beyond offline imitation.

Data Pipeline Ablation. Table 4 shows that each stage of our data synthesis pipeline contributes to the final performance. The full pipeline achieves the best average score of **64.6**, while removing source-anchor grounding, fuzzy entity rewriting, or staged filtering leads to large drops of 11.5, 10.3, and 8.2 points, respectively. These results indicate that effective training data must both prevent shortcut retrieval and preserve genuinely tool-demanding queries. Removing the enhancement subset causes a smaller but consistent decline (64.6 $\rightarrow$ 63.3 Avg.), suggesting that image-restoration trajectories mainly improve robustness rather than driving the core gains.

Training Recipe Ablation. Table 3 studies the effect of our RL recipe after the same Qwen3-VL-8B SFT initialization, using Search-R1-style vanilla GRPO [18] as the RL baseline. SFT improves the base model from 53.7 to 64.6 average accuracy, and vanilla GRPO further raises it to 67.6, showing the benefit of online exploration. The way fatal trajectories are handled is crucial: the hard-masking strategy of Vision-DeepResearch [1] brings almost no gain over vanilla GRPO [18, 11](67.6 $\rightarrow$ 67.7), while fatal masking improves the average to 69.1 by preserving valid pre-failure reasoning. Our full method with one-sided advantage clamping achieves the best score on every benchmark and reaches **71.8** average accuracy, a 4.2-point gain over vanilla GRPO. The training curves in Fig. 3 are consistent with this result: fatal-aware GRPO sustains longer tool-use trajectories while achieving higher batch accuracy, indicating that it encourages productive exploration rather than prematurely suppressing difficult rollouts.

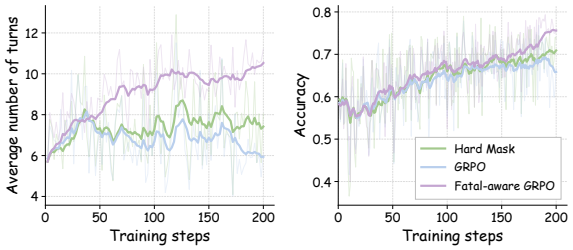

Figure 3: Training dynamics over the RL phase. **Left:** average number of turns per rollout. **Right:** batch-level accuracy. Fatal-aware GRPO sustains a higher number of turns *and* reaches a higher accuracy than vanilla GRPO [18, 11] and the Hard-Mask [1] baseline.

6 Related Work

Multimodal Agentic Search. Active search has reframed LLMs from static knowledge bases into agentic reasoners that retrieve information dynamically. Search-R1 [18] crystallized this shift, using RL to incentivize autonomous, multi-turn querying within the reasoning chain. The paradigm has

since migrated to multimodal domains: MMSearch-R1 [33] and Vision-DeepResearch [1, 39] embed visual retrieval into agent pipelines, while recent efforts [23, 40, 3, 2, 41] unify image manipulation with web search. These works share a fragile assumption—pristine visual inputs. On degraded or text-dense real-world images the "search-only" approach fails, since retrieval cannot repair fundamentally broken visual evidence.

Visual Perception and Retrieval. Multimodal agents are often bottlenecked not by retrieval logic but by the fidelity of initial perception [42, 43]. RAG frameworks such as VisRAG [44] preserve visual structure, yet treat the model as a passive observer that must "make do" with whatever is retrieved, and tool-augmented agents [45, 34, 10] add dynamic orchestration over toolsets that remain homogeneous and scenario-bound. We argue that robust reasoning requires *active perception*: the agent must intervene as well as search, autonomously invoking tools such as super-resolution or specialized OCR to remediate visual noise before reasoning over it.

Reinforcement Learning for Agentic Reasoning. Long-horizon, multi-tool environments pose substantial challenges for standard RL. GRPO [28] aligns reasoning trajectories effectively for language models [11, 46, 47, 48, 49, 50], but applying it to agentic rollouts with diverse tool interactions remains non-trivial. The primary difficulty is *cascading failures*: an early tool error renders the rest of the trajectory incoherent [51, 52, 53, 54], yet its post-failure tokens still inject noise into the policy gradient [55, 56, 57], and standard group normalization penalizes the valid reasoning prefixes of partially successful rollouts [1]. We therefore introduce a fatal-aware RL objective that prunes learning signals from post-failure states and applies one-sided advantage clamping to protect the gradients of constructive reasoning steps.

7 Limitations and Future Work

A non-trivial fraction of training instability traces to the tool environment $\mathcal{E}$ rather than the policy itself. The cascade of Serper, JINA Reader, and the Qwen3-32B summariser inside TEXTSEARCH, together with IMAGESEARCH, exhibits ranking drift, fetch failures, and occasional summarisation hallucinations that inflate reward variance even after fatal-aware masking; on-policy reliability estimation or a lightweight verifier over summaries is a natural remedy. The composite reward (Eq. 6) also relies on proprietary GPT-4o for both r_{acc} and r_{query}, which is neither cheap at scale nor stable across model versions, and r_{query} scores only textual queries while ignoring intermediate visual operations such as CROP or PERSPECTIVECORRECT. Replacing the LLM judges with open process reward models and extending dense rewards to the full visual action space $\mathcal{T}_v$ would improve reproducibility and credit assignment.

A second class of limitations concerns reproducibility. (i) Several components of $\mathcal{E}$—Serper, JINA Reader, the Polaris Lens API behind IMAGESEARCH, the PaddleX-backed OCR service, and the GPT-4o-based judges—are externally hosted and version-dependent, so exact reproduction of an individual benchmark cell can drift over time even under a frozen policy and decoding configuration. (ii) Pass@1 numbers in Table 2 come from a single run per setting, following the convention of recent multimodal deep-search benchmarks; multi-seed error bars are omitted because of the cost of rolling out 7 benchmarks across all 8B/30B-A3B/32B variants. To support open verification we will release the datasets, checkpoints, and the training, inference, and evaluation code under permissive open licenses. We regard (i)–(ii) as honest constraints of the current evaluation setting rather than evidence against the methodology, and flag them so downstream users can plan around them.

8 Conclusion

We present **OpenSearch-VL**, a fully open recipe for training multimodal deep search agents with agentic reinforcement learning. It combines a Wikipedia-based data curation pipeline that mitigates one-step retrieval shortcuts and yields two high-quality datasets, *SearchVL-SFT-36k* & *SearchVL-RL-8k*; a diverse tool environment spanning retrieval, image enhancement, and attention-and-parsing tools; and a multi-turn fatal-aware GRPO algorithm that preserves useful pre-failure reasoning through one-sided advantage clamping. OpenSearch-VL achieves over 10-point average gains across seven multimodal deep search benchmarks, remaining competitive with strong proprietary reasoning models on representative tasks such as VDR. We will release our data, code, models, and recipe to lower the reproducibility barrier for future research on multimodal deep search agents.

Acknowledgments and Disclosure of Funding

This work was supported by Tencent Hunyuan, which provided the computational resources, large-scale training infrastructure, and tool and search APIs used for all training and evaluation reported in this paper. We thank Tencent Hunyuan for this support. The authors received no other third-party support in connection with this work during the last 36 months.

Competing interests. A subset of the authors are affiliated with Tencent Hunyuan, which develops and releases multimodal foundation models. Beyond this affiliation and the support described above, the authors declare no financial relationships outside the submitted work that could be perceived as influencing it.

References

[1] Wenxuan Huang, Yu Zeng, Qiuchen Wang, Zhen Fang, Shaosheng Cao, Zheng Chu, Qingyu Yin, Shuang Chen, Zhenfei Yin, Lin Chen, et al. Vision-deepresearch: Incentivizing deepresearch capability in multimodal large language models. *arXiv preprint arXiv:2601.22060*, 2026.

[2] Kaituo Feng, Manyuan Zhang, Shuang Chen, Yunlong Lin, Kaixuan Fan, Yilei Jiang, Hongyu Li, Dian Zheng, Chenyang Wang, and Xiangyu Yue. Gen-searcher: Reinforcing agentic search for image generation. *arXiv preprint arXiv:2603.28767*, 2026.

[3] Shuang Chen, Quanxin Shou, Hangting Chen, Yucheng Zhou, Kaituo Feng, Wenbo Hu, Yi-Fan Zhang, Yunlong Lin, Wenxuan Huang, Mingyang Song, et al. Unify-agent: A unified multimodal agent for world-grounded image synthesis. *arXiv preprint arXiv:2603.29620*, 2026.

[4] Bytedance Seed. Seed2.0 model card, 2026.

[5] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[6] Kimi Team. Kimi k2.5: Visual agentic intelligence, 2026.

[7] Anthropic Team. System card: Claude opus 4 & claude sonnet 4. https://www-cdn.anthropic.com/6d8a8055020700718b0c49369f60816ba2a7c285.pdf, 2025.

[8] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[9] Kaixuan Fan, Kaituo Feng, Manyuan Zhang, Tianshuo Peng, Zhixun Li, Yilei Jiang, Shuang Chen, Peng Pei, Xunliang Cai, and Xiangyu Yue. Exploring reasoning reward model for agents. *arXiv preprint arXiv:2601.22154*, 2026.

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

[11] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

[12] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[13] Yu Zeng, Wenxuan Huang, Zhen Fang, Shuang Chen, Yufan Shen, Yishuo Cai, Xiaoman Wang, Zhenfei Yin, Lin Chen, Zehui Chen, Shiting Huang, Yiming Zhao, Yao Hu, Philip Torr, Wanli Ouyang, and Shaosheng Cao. Vision-deepresearch benchmark: Rethinking visual and textual search for multimodal large language models. *preprint*, 2026.

[14] Dongzhi Jiang, Renrui Zhang, Ziyu Guo, Yanmin Wu, Pengshuo Qiu, Pan Lu, Zehui Chen, Guanglu Song, Peng Gao, Yu Liu, et al. Mmsearch: Unveiling the potential of large models as multi-modal search engines. In *The Thirteenth International Conference on Learning Representations*.

[15] Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Fvqa: Fact-based visual question answering. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2413–2427, 2017.

[16] Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? *arXiv preprint arXiv:2302.11713*, 2023.

[17] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

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

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

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

[21] Wikipedia. https://www.wikipedia.org/.

[22] OpenAI Team. Gpt-4o system card, 2024.

[23] Yifan Zhang, Liang Hu, Haofeng Sun, Peiyu Wang, Yichen Wei, Shukang Yin, Jiangbo Pei, Wei Shen, Peng Xia, Yi Peng, et al. Skywork-r1v4: Toward agentic multimodal intelligence through interleaved thinking with images and deepresearch. *arXiv preprint arXiv:2512.02395*, 2025.

[24] Mingyang Fu, Yuyang Peng, Benlin Liu, Yao Wan, and Dongping Chen. Livevqa: Live visual knowledge seeking. *arXiv preprint arXiv:2504.05288*, 2025.

[25] Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. Webqa: Multihop and multimodal qa, 2022.

[26] Anthropic Team. Claude opus 4.6 system card. https://www-cdn.anthropic.com/14e4fb01875d2a69f646fa5e574dea2b1c0ff7b5.pdf, 2026.

[27] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025.

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

[29] OpenAI. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/, 2025.

[30] Anthropic Team. Claude 3.7 sonnet system card. https://www-cdn.anthropic.com/9ff93dfa8f445c932415d335c88852ef47f1201e.pdf, 2025.

[31] Kartik Narayan, Yang Xu, Tian Cao, Kavya Nerella, Vishal M Patel, Navid Shiee, Peter Grasch, Chao Jia, Yinfei Yang, and Zhe Gan. Deepmmsearch-r1: Empowering multimodal llms in multimodal web search. *arXiv preprint arXiv:2510.12801*, 2025.

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

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

[34] Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. Deepeyesv2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025.

[35] Yong Xien Chng, Tao Hu, Wenwen Tong, Xueheng Li, Jiandong Chen, Haojia Yu, Jiefan Lu, Hewei Guo, Hanming Deng, Chengjun Xie, et al. Sensenova-mars: Empowering multimodal agentic reasoning and search via reinforcement learning. *arXiv preprint arXiv:2512.24330*, 2025.

[36] Xianfu Cheng, Wei Zhang, Shiwei Zhang, Jian Yang, Xiangyuan Guan, Xianjie Wu, Xiang Li, Ge Zhang, Jiaheng Liu, Yuying Mai, et al. Simplevqa: Multimodal factuality evaluation for multimodal large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4637–4646, 2025.

[37] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics.

[38] Sijun Tan, Michael Luo, Colin Cai, Tarun Venkat, Kyle Montgomery, Aaron Hao, Tianhao Wu, Arnav Balyan, Manan Roongta, Chenguang Wang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. rllm: A framework for post-training language agents, 2025. Notion Blog.

[39] Kartik Narayan, Yang Xu, Tian Cao, Kavya Nerella, Vishal M Patel, Navid Shiee, Peter Grasch, Chao Jia, Yinfei Yang, and Zhe Gan. Deepmmsearch-r1: Empowering multimodal llms in multimodal web search. *arXiv preprint arXiv:2510.12801*, 2025.

[40] Huanjin Yao, Qixiang Yin, Min Yang, Ziwang Zhao, Yibo Wang, Haotian Luo, Jingyi Zhang, and Jiaxing Huang. Mm-deepresearch: A simple and effective multimodal agentic search baseline. *arXiv preprint arXiv:2603.01050*, 2026.

[41] Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. Websailor: Navigating super-human reasoning for web agent, 2025.

[42] Jiankun Zhang, Shenglai Zeng, Kai Guo, Xinnan Dai, Hui Liu, Jiliang Tang, and Yi Chang. Fix before search: Benchmarking agentic query visual pre-processing in multimodal retrieval-augmented generation. *arXiv preprint arXiv:2602.13179*, 2026.

[43] Yana Wei, Liang Zhao, Kangheng Lin, En Yu, Yuang Peng, Runpei Dong, Jianjian Sun, Haoran Wei, Zheng Ge, Xiangyu Zhang, et al. Perception in reflection. *arXiv preprint arXiv:2504.07165*, 2025.

[44] Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, et al. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *arXiv preprint arXiv:2410.10594*, 2024.

[45] Mingyang Song, Haoyu Sun, Jiawei Gu, Linjie Li, Luxin Xu, Ranjay Krishna, and Yu Cheng. Adareasoner: Dynamic tool orchestration for iterative visual reasoning, 2026.

[46] Shuang Chen, Yue Guo, Zhaochen Su, Yafu Li, Yulun Wu, Jiacheng Chen, Jiayu Chen, Weijie Wang, Xiaoye Qu, and Yu Cheng. Advancing multimodal reasoning: From optimized cold start to staged reinforcement learning. *arXiv preprint arXiv:2506.04207*, 2025.

[47] Shuang Chen, Yue Guo, Yimeng Ye, Shijue Huang, Wenbo Hu, Haoxi Li, Manyuan Zhang, Jiayu Chen, Song Guo, and Nanyun Peng. Ares: Multimodal adaptive reasoning via difficulty-aware token-level entropy shaping. *arXiv preprint arXiv:2510.08457*, 2025.

[48] Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms. *arXiv preprint arXiv:2503.21776*, 2025.

[49] Kaituo Feng, Manyuan Zhang, Hongyu Li, Kaixuan Fan, Shuang Chen, Yilei Jiang, Dian Zheng, Peiwen Sun, Yiyuan Zhang, Haoze Sun, et al. Onethinker: All-in-one reasoning model for image and video. *arXiv preprint arXiv:2512.03043*, 2025.

[50] Wenbo Hu, Xin Chen, Yan Gao-Tian, Yihe Deng, Nanyun Peng, and Kai-Wei Chang. Openvl-thinkerv2: A generalist multimodal reasoning model for multi-domain visual tasks, 2026.

[51] Sri Vatsa Vuddanti, Aarav Shah, Satwik Kumar Chittiprolu, Tony Song, Sunishchal Dev, Kevin Zhu, and Maheep Chaudhary. Paladin: Self-correcting language model agents to cure tool-failure cases, 2025.

[52] Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where llm agents fail and how they can learn from failures, 2025.

[53] Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning, 2026.

[54] Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization, 2025.

[55] Haoyu Liu and Le Xiao. Re-grpo: Leveraging hard negative cases through large language model guided self training. *Neurocomputing*, page 132543, 2025.

[56] Wenlong Deng, Yushu Li, Boying Gong, Yi Ren, Christos Thrampoulidis, and Xiaoxiao Li. On group relative policy optimization collapse in agent search: The lazy likelihood-displacement, 2026.

[57] Anisha Garg, Claire Zhang, Nishit Neema, David Bick, Ganesh Venkatesh, and Joel Hestness. Corpo: Adding a correctness bias to grpo improves generalization, 2026.

[58] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[59] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

Appendix

Appendix Contents

A Preliminary of Reinforcement Learning **15**

A.1 Standard GRPO 15

A.2 GRPO with Search Engine 16

B Multi-Turn Search Fatal-Aware GRPO Details **17**

B.1 Reward Design Formalization 17

B.2 Fatal Step Detection Logic 17

B.3 Advantage Derivation with One-Sided Clamping 19

C Implementation Details **20**

C.1 SFT Training Configuration 21

C.2 RL Training Configuration 21

D Extended Problem Formulation **22**

E Data Curation Details **25**

E.1 Path Sampling Hyperparameters 25

E.2 Running Example: `Australia_Zoo` 25

E.3 Counterfactual Design Choices 26

F System Prompts **27**

G Tool Definition and Usage **27**

H Case Study **30**

A Preliminary of Reinforcement Learning

This section details the reinforcement learning (RL) preliminaries underpinning our multi-turn training objective. We first review the standard Group Relative Policy Optimization (GRPO) algorithm [28], which operates on single-turn generations. Subsequently, we describe its extension to search-augmented rollouts, as introduced by Search-R1 [18], wherein the policy interleaves generation with calls to an external search engine. These formulations serve as the direct precursors to the multi-turn, multi-tool objective employed by OpenSearch-VL (see Sec. 4.2).

A.1 Standard GRPO

GRPO [28] is an actor-critic variant of Proximal Policy Optimization (PPO) [58] that eliminates the reliance on a learned value function. Instead of maintaining a separate critic model to estimate a baseline, GRPO derives its advantage estimate by comparing multiple responses sampled for the same prompt, collectively defining a *group*. This parameter-efficient design is particularly advantageous in settings where only a scalar outcome-level reward is available, reducing memory overhead and mitigating the instability of training a dense token-level value function.

Formally, given a prompt distribution $\mathcal{D}$ and a reference policy $\pi_{\theta_{\text{old}}}$, GRPO samples a group of G candidate responses $\{o_1, o_2, \ldots, o_G\} \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$ for each prompt $q \sim \mathcal{D}$. Each response is evaluated via a reward function $r_i = r(q, o_i)$. To compute the advantages, GRPO standardizes these

rewards within the local group:

$$\hat{A}_{i,t} \;=\; \widetilde{r}_i \;=\; \frac{r_i - \mathrm{mean}(\{r_j\}_{j=1}^{G})}{\mathrm{std}(\{r_j\}_{j=1}^{G})}, \quad \text{for all tokens } t = 1, \ldots, |o_i|. \tag{10}$$

By this formulation, all constituent tokens of a given response o_i are assigned a uniform advantage scalar. The policy parameters θ are subsequently optimized by maximizing the clipped surrogate objective:

$$\begin{aligned} \mathcal{J}_{\text{GRPO}}(\theta) \;=\; \mathbb{E}_{q\sim\mathcal{D},\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \Bigg\{ \\ \min\Big(\rho_{i,t}(\theta)\,\hat{A}_{i,t},\;\; \mathrm{clip}\big(\rho_{i,t}(\theta),\,1-\epsilon,\,1+\epsilon\big)\,\hat{A}_{i,t}\Big) \\ -\;\beta\,\mathbb{D}_{\text{KL}}\big[\pi_\theta \,\|\, \pi_{\text{ref}}\big] \Bigg\}\Bigg], \end{aligned} \tag{11}$$

where the importance sampling ratio is defined as

$$\rho_{i,t}(\theta) \;=\; \frac{\pi_\theta(o_{i,t} \mid q,\, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q,\, o_{i,<t})}. \tag{12}$$

Here, ϵ represents the probability ratio clipping hyperparameter, and β modulates the Kullback–Leibler (KL) divergence penalty against a fixed reference policy π_{ref}. In contrast to standard PPO—which subsumes the KL penalty directly into the reward signal and necessitates a parameterized value network to approximate $\hat{A}_{i,t}$ [58]—GRPO enforces the KL regularization explicitly within the loss landscape. By deriving $\hat{A}_{i,t}$ strictly from empirical group statistics (Eq. 10), GRPO naturally aligns with the comparative structure of preference-based reward modeling.

A.2 GRPO with Search Engine

Search-R1 [18] extends the GRPO framework from unimodal, single-turn generation to an interleaved, search-augmented generative process. Rather than sampling an isolated response $o_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid q)$, Search-R1 generates a multi-step rollout comprising both policy-emitted tokens and external evidence retrieved from a search engine $\mathcal{R}$. This search-augmented sampling distribution is denoted as:

$$o_i \;\sim\; \pi_{\theta_{\text{old}}}(\cdot \mid q; \mathcal{R}) \;=\; \pi_{\theta_{\text{old}}}(\cdot \mid q) \bigotimes \mathcal{R}, \tag{13}$$

where $\bigotimes$ represents the interleaving operator: whenever the policy emits a search command, the retrieved documents are appended to the conditioning context, and the policy resumes generation conditioned on this expanded prefix.

To prevent the optimizer from erroneously updating parameters based on exogenous environmental tokens, Search-R1 utilizes a masking function that filters out non-generated tokens. Defining $M_{\text{gen}}(y_{i,t}) \in \{0, 1\}$ as an indicator variable where $M_{\text{gen}}(y_{i,t}) = 1$ if $y_{i,t}$ is explicitly generated by the policy, the corresponding Search-R1 GRPO objective becomes:

$$\begin{aligned} \mathcal{J}^{\mathcal{R}}_{\text{GRPO}}(\theta) \;=\; \mathbb{E}_{q\sim\mathcal{D},\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q;\mathcal{R})} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \frac{1}{\sum_{t=1}^{|o_i|} M_{\text{gen}}(y_{i,t})} \sum_{\substack{t=1 \\ M_{\text{gen}}(y_{i,t})=1}}^{|o_i|} \Bigg\{ \\ \min\Big(\rho^{\mathcal{R}}_{i,t}(\theta)\,\hat{A}_{i,t},\;\; \mathrm{clip}\big(\rho^{\mathcal{R}}_{i,t}(\theta),\,1-\epsilon,\,1+\epsilon\big)\,\hat{A}_{i,t}\Big) \\ -\;\beta\,\mathbb{D}_{\text{KL}}\big[\pi_\theta \,\|\, \pi_{\text{ref}}\big] \Bigg\}\Bigg], \end{aligned} \tag{14}$$

featuring the environment-conditioned importance ratio:

$$\rho^{\mathcal{R}}_{i,t}(\theta) \;=\; \frac{\pi_\theta(y_{i,t} \mid q,\, y_{i,<t};\, \mathcal{R})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid q,\, y_{i,<t};\, \mathcal{R})}. \tag{15}$$

Relative to standard GRPO (Eq. 11), this adaptation introduces two pivotal modifications. First, the expectation is calculated with respect to the interleaved distribution $\pi_{\theta_{\text{old}}}(\cdot \mid q; \mathcal{R})$, ensuring the causal conditioning history incorporates all previously retrieved external evidence. Second, the objective is normalized strictly by the count of generated tokens, $\sum_t M_{\text{gen}}(y_{i,t})$, restricting gradient calculations

to policy-authored positions. Equivalently masking the KL divergence ensures the model is not penalized for distributional divergence over environmental observations. These mechanisms adapt GRPO to tool-in-the-loop architectures and establish the theoretical foundation for our multimodal environment $\mathcal{E}$ formulation (Sec. 4.2).

B Multi-Turn Search Fatal-Aware GRPO Details

Building upon the foundations established in Appendix A, this section details the mechanics of our multi-turn, search-augmented RL objective introduced in Sec. 4.2. We formalize the constituent elements of our composite reward function, articulate the detection logic for fatal execution cascades, and rigorously derive the advantage estimation process incorporating one-sided clamping.

B.1 Reward Design Formalization

The composite trajectory-level reward (Eq. 6) is structured to balance algorithmic formatting, terminal accuracy, and process-level search quality. We define each component as follows:

Format Reward r_{fmt}. This acts as a deterministic, algorithmic prior to enforce structural integrity. We define $r_{\text{fmt}}(\tau) = \frac{1}{L+1}\sum_{l=0}^{L} r_{\text{fmt}}^{(l)}$, where the step-wise indicator $r_{\text{fmt}}^{(l)} = 1$ if step l correctly emits a contiguous `<think>`$\cdots$`</think>` block immediately followed by either a `<tool_call>`$\cdots$`</tool_call>` block (for intermediate steps $l < L$) or a `<response>`$\cdots$`</response>` block (for the terminal step $l = L$). Conversely, $r_{\text{fmt}}^{(l)} = 0$ for any structural violation. Crucially, steps that precipitate tool-execution errors (e.g., malformed API arguments) are assigned $r_{\text{fmt}}^{(l)} = 0$. By operating as a multiplicative gate in the composite reward formulation, r_{fmt} systematically drives the overall return of structurally degraded trajectories toward zero.

Accuracy Reward r_{acc}. This terminal outcome metric assesses the fidelity of the agent's final resolution. A GPT-4o [22] judge verifies semantic equivalence between the agent's terminal output and the ground-truth annotation, assigning $r_{\text{acc}} = 1$ for a match and $r_{\text{acc}} = 0$ otherwise. **Convention for Truncated Trajectories.** For trajectories that are aborted by the fatal-state condition prior to emitting a terminal `<response>`, we deterministically assign $r_{\text{acc}} = 0$. This is not an evaluative judgment but a structural guarantee: in the absence of a terminal answer, correctness is strictly undefined and conservatively zeroed out. This convention ensures that the outcome signal remains mathematically well-defined for group-relative advantage estimation, irrespective of the trajectory's completion status.

Query Quality Reward r_{query}. To counteract the inherent sparsity of the terminal accuracy signal—particularly critical in long-horizon interactions—we employ a process-level heuristic. We use GPT-4o [22] to evaluate the cumulative sequence of search queries on a continuous $[0, 1]$ scale. The evaluation rubric encompasses four dimensions: (i) the semantic relevance of the issued queries to the initial prompt, (ii) the logical progression and iterative refinement of queries across successive turns, (iii) the signal-to-noise ratio within the retrieved payloads, and (iv) the effective cross-modal complementary utilization of image and text retrieval tools. For trajectories designated as fatal, the judge restricts its evaluation strictly to the valid pre-fatal prefix (i.e., steps $l < f_i$), ensuring the agent is credited for sound early-stage reasoning despite subsequent collapse.

B.2 Fatal Step Detection Logic

Unconstrained interactions with external environments inevitably yield "fatal" states—irrecoverable error cascades rendering subsequent reasoning invalid. We articulate the procedure for computing the fatal step index f_i, which drives the fatal-aware token mask (Eq. 7).

For a given trajectory τ_i of length L_i, we instantiate a stateful error counter n_{err} initialized to zero. At each sequential step $l \in \{0, 1, \ldots, L_i\}$, the counter transitions according to:

$$n_{\text{err}}^{(l)} = \begin{cases} n_{\text{err}}^{(l-1)} + 1 & \text{if step } l \text{ triggers a tool-execution error,} \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

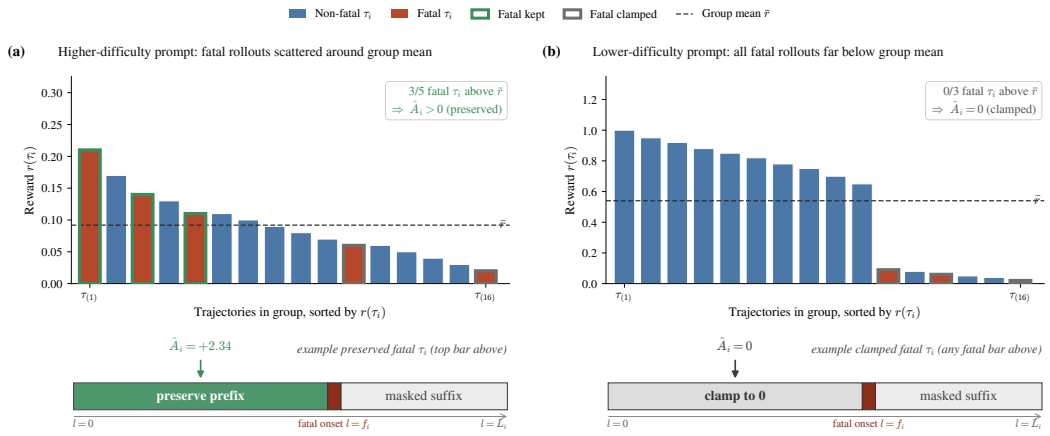

Figure 4: **Fatal-aware masking with one-sided clamping on two illustrative rollout groups.** Each panel shows a group of G=16 rollouts (non-fatal in blue, fatal in brick red) together with a cartoon of the resulting token-level update on one representative fatal trajectory τ_i. Bar outlines mark fatal rollouts whose pre-clamp score $\widetilde{r}_i$ exceeds the group mean $\bar{r}$ (green, preserved) or falls below it (dark grey, clamped). **(a) Higher-difficulty prompt.** When $\bar{r}$ is low, several fatal prefixes already beat it: $\widetilde{r}_i > 0$ and $\hat{A}_i = \widetilde{r}_i$, so gradients flow only through the viable prefix (tokens up to the fatal onset f_i) while the post-failure suffix is hard-masked – the partial reasoning that reached the failure point is *reinforced* even though the trajectory itself is unsolvable. **(b) Lower-difficulty prompt.** When most rollouts succeed, every fatal reward falls far below $\bar{r}$ and one-sided clamping sets $\hat{A}_i = \max(\widetilde{r}_i, 0) = 0$, degenerating to pure hard-masking and avoiding any suppression of the possibly-valid prefix.

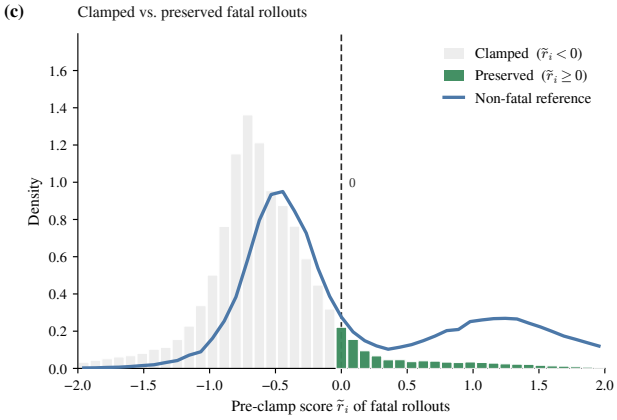

Figure 5: **Aggregate distribution of clamped and preserved fatal rollouts.** Pre-clamp group-normalized scores $\widetilde{r}_i$ aggregated over 10,000 groups (G=16; 47,978 fatal rollouts in total). 91.8% of fatal rollouts fall on the negative side of the clamp threshold (mean $\overline{\widetilde{r}}_- = -0.68$) and are zeroed out by $\hat{A}_i = \max(\widetilde{r}_i, 0)$; the remaining 8.2% are preserved (mean $\overline{\widetilde{r}}_+ = +0.57$) and overlap with the positive mode of the non-fatal reference density (blue line: pre-clamp scores of all non-fatal rollouts from the same groups). Preservation is therefore driven by the same group-normalized score that GRPO already computes, and the preserved fatal prefixes sit in the same score regime as stronger non-fatal behaviors – not an ad-hoc heuristic.

The fatal step index is subsequently defined as the earliest step where the error threshold is breached:

$$f_i = \min\{l : n_{\text{err}}^{(l)} = K\}. \tag{17}$$

In our implementation, we set $K = 3$. If the threshold is never reached, the trajectory is classified as non-fatal, denoted by setting $f_i = L_i + 1$.

This formulation embodies two critical design properties. First, the explicit reset condition ($n_{\text{err}}^{(l)} = 0$) ensures that *isolated* transient errors—which are ubiquitous and often recoverable in realistic web

environments—do not prematurely abort the trajectory. Second, the conservative threshold ($K = 3$) dictates that a trajectory is only deemed fatal following three *consecutive* failures, affording the autoregressive policy a robust opportunity to self-correct prior to gradient masking. Tool-execution errors triggering this counter encompass timeouts, malformed API payloads, and argument-parsing failures; these are flagged deterministically by the execution sandbox, independent of any learned heuristic.

B.3 Advantage Derivation with One-Sided Clamping

We rigorously detail the advantage estimation protocol underpinning the objective function in Eq. 9.

Step 1: Unbiased Group Statistics. For a sampled group of G rollouts originating from the identical prompt (I_0, q), the empirical group mean and standard deviation are computed over the composite rewards:

$$\mu_{\mathcal{G}} \;=\; \mathrm{sg}\left(\frac{1}{G}\sum_{j=1}^{G} r(\tau_j)\right), \qquad \sigma_{\mathcal{G}} \;=\; \mathrm{sg}\left(\sqrt{\frac{1}{G}\sum_{j=1}^{G}\big(r(\tau_j)-\mu_{\mathcal{G}}\big)^2}\right). \tag{18}$$

Here, $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. Consequently, $\mu_{\mathcal{G}}$ and $\sigma_{\mathcal{G}}$ act as static normalizing constants during backpropagation. Gradients flow exclusively through the importance sampling ratios evaluated on the unmasked, policy-generated tokens.

Step 2: Universal Group Normalization. Each trajectory within the group is assigned a standardized reward:

$$\widetilde{r}_i \;=\; \frac{r(\tau_i)-\mu_{\mathcal{G}}}{\sigma_{\mathcal{G}}+\delta}, \tag{19}$$

where $\delta > 0$ provides numerical stability. Crucially, all G trajectories—expressly including those truncated by the fatal condition ($f_i \le L_i$)—are incorporated into the calculation of $\mu_{\mathcal{G}}$ and $\sigma_{\mathcal{G}}$. This deliberate inclusion ensures that fatal trajectories actively shape the group-level baseline, anchoring the relative performance ranking across the entire sampled cohort.

Step 3: Fatal-Aware One-Sided Clamping. The standardized rewards are mapped to the final advantage estimates $\hat{A}_i$ via a piecewise clamping function:

$$\hat{A}_i \;=\; \begin{cases} \widetilde{r}_i & \text{if } f_i = L_i{+}1 \quad \text{(non-fatal trajectory)},\\ \max\big(\widetilde{r}_i,\ 0\big) & \text{if } f_i \le L_i \quad \text{(fatal trajectory)}. \end{cases} \tag{20}$$

Step 4: Gradient Dominance over Hard-Masking. We now formalize the informal claim that fatal-aware clamping strictly dominates the hard-masking baseline of [1] in gradient informativeness. Let

$$g_i^{\text{ours}} \;=\; \frac{1}{\sum_t M_{i,t}} \sum_{t=1}^{|\tau_i|} M_{i,t}\,\nabla_\theta \log \pi_\theta\big(y_{i,t} \mid h_{s(t)}\big)\,\hat{A}_i \tag{21}$$

denote the (un-clipped) per-trajectory contribution to $\nabla_\theta \mathcal{J}(\theta)$ under our scheme, and let g_i^{hard} denote the corresponding contribution under the hard-masking baseline, which discards every fatal trajectory in full so that $g_i^{\text{hard}} = 0$ whenever $f_i \le L_i$. We compare the two on the support of fatal trajectories.

Proposition 1 (Dominance over hard-masking)**.** *Fix any fatal trajectory τ_i ($f_i \le L_i$). Then:*

1. *If $\widetilde{r}_i < 0$, then $g_i^{ours} = g_i^{hard} = 0$.*
2. *If $\widetilde{r}_i \ge 0$, then $g_i^{hard} = 0$, while g_i^{ours} coincides with the search-augmented GRPO gradient (Eq. 14) of τ_i restricted to its viable prefix $\{t : s(t) < f_i,\ M_{gen}(y_{i,t}) = 1\}$, evaluated at the non-negative advantage $\widetilde{r}_i$.*

Consequently, g_i^{ours} is weakly informative-dominant over g_i^{hard}: it never propagates gradient through the post-fatal suffix, never penalises the viable prefix, and strictly extracts positive reinforcement on prefixes whose group-normalised return exceeds the baseline.

Proof. By Eq. 8, $\widetilde{r}_i < 0$ implies $\hat{A}_i = 0$, so the surrogate term and its gradient vanish identically in Eq. 21; combined with $g_i^{\text{hard}} = 0$ by definition of hard-masking, this establishes (i). For $\widetilde{r}_i \ge 0$,

$\hat{A}_i = \widetilde{r}_i$, while the fatal-aware mask $M_{i,t} = M_{\text{gen}}(y_{i,t}) \cdot \mathbb{1}[s(t) < f_i]$ retains exactly the policy-generated tokens of the viable prefix and zeros out (a) environment-emitted tokens and (b) all post-fatal tokens. On this support the per-token integrand of Eq. 21 reduces token-by-token to that of Eq. 14 with the same importance ratio $\rho_{i,t}$ and advantage $\widetilde{r}_i \geq 0$, so the per-token surrogates—and hence their gradients—agree. Since $g_i^{\text{hard}} = 0$ by definition, this establishes (ii). □

In aggregate, the one-sided clamp safely ignores invalid credit assignment while selectively harvesting positive reinforcement from prematurely truncated yet high-quality exploratory rollouts.

Bias of Group Statistics under Clamping. The asymmetric definition of $\hat{A}_i$ deserves an honest accounting. We compute $(\mu_{\mathcal{G}}, \sigma_{\mathcal{G}})$ in Step 1 over *all* G rollouts—including fatal ones—so the standardised score $\widetilde{r}_i$ is zero-mean within the group; Step 3 then maps any negative $\widetilde{r}_i$ on a fatal trajectory to zero. Letting $\mathcal{F} \subseteq \{1, \ldots, G\}$ index the fatal subset of the group,

$$\frac{1}{G}\sum_{i=1}^{G} \hat{A}_i \;=\; \underbrace{\frac{1}{G}\sum_{i=1}^{G} \widetilde{r}_i}_{=0} \;+\; \underbrace{\frac{1}{G}\sum_{i\in\mathcal{F}} \max\big(0,\, -\widetilde{r}_i\big)}_{=\, b_{\mathcal{G}} \geq 0}, \tag{22}$$

so the clamp induces a non-negative bias $b_{\mathcal{G}}$ relative to the zero-mean GRPO baseline, in exchange for non-zero gradient on viable prefixes. We argue this trade-off is benign in our regime for two reasons. (a) *Concentration of fatal rollouts in the lower mode.* Figure 5 shows that 91.8% of fatal rollouts have $\widetilde{r}_i < 0$ and are clamped to zero, while the residual 8.2% that survive sit in the same score regime as competitive non-fatal rollouts; $b_{\mathcal{G}}$ is therefore dominated by samples whose viable prefix is empirically high-quality, rather than by indiscriminate inflation of low-quality fatals. (b) *Preservation of relative ranking.* Because $b_{\mathcal{G}}$ shifts only the fatal subset upward and never displaces non-fatal advantages, the within-group ordering between non-fatal trajectories—which is the actual signal GRPO exploits—remains intact. We therefore interpret $b_{\mathcal{G}}$ as a controlled positive bias that buys back gradient on prematurely truncated yet promising prefixes; replacing the clamp with an unbiased estimator (e.g., a separate baseline computed only over non-fatal rollouts, or a doubly-robust correction for $b_{\mathcal{G}}$) is an interesting alternative we leave for future work.

Empirical Visualization of the Two Cases. Figure 4 visualizes how $\hat{A}_i = \max(\widetilde{r}_i, 0)$ interacts with per-group difficulty on two representative rollout groups. In the higher-difficulty case, most fatal trajectories still emit a coherent prefix before hitting a tool error; a fraction of them beat the group mean and contribute a positive gradient to the prefix tokens only. In the lower-difficulty case, every fatal trajectory is dominated by the successful non-fatal rollouts, and clamping to zero prevents a noisy negative signal from pushing the policy *away* from what may actually be a valid prefix. Figure 5 shows that this split is not anecdotal: aggregated over $10{,}000$ groups, the vast majority of fatal trajectories sit on the negative side of the pre-clamp score and are safely zeroed out, while the small preserved tail is distributionally close to the right mode of the non-fatal reference. One-sided clamping thus recovers a principled credit signal from failed tool-call trajectories without amplifying their inherent noise.

C Implementation Details

Our agentic SFT pipeline builds on LLAMAFACTORY [37], which we extend with multi-turn, tool-interleaved data collators and a Qwen3-VL-aware vision/text packing scheme so that interleaved image observations produced by visual tools (CROP, SHARPEN, SUPERRESOLUTION, PERSPECTIVECORRECT, OCR) can be consumed verbatim by the policy. Our RL pipeline builds jointly on RLLM [38] and VISION-DEEPRESEARCH [1]: from the former we adopt the asynchronous agent rollout architecture (decoupled trajectory generation, replay, and policy-update workers), and from the latter we adopt the multimodal trajectory abstraction. On top of these two, we implement a Qwen3-VL chat-template renderer, an interleaved image-token re-alignment routine for variable-length visual observations, and an asynchronous multi-turn rollout engine that supports tool-call interruption and resumption. The resulting pipeline is what we use to train all OpenSearch-VL variants in this paper.

All stages of OPENSEARCH-VL are trained on Nvidia H20 GPUs. Agentic SFT takes roughly 2 days for the 8B dense model and 4 days for the 30B-A3B MoE model on 256 H20s (32 nodes $\times$ 8 GPUs);

Category	Hyperparameter	Value / Setting
Model	Base Model	Qwen3-VL-8B-Instruct (32B, 30B-A3B)
	Image Max Pixels	262,144 ($\approx 512 \times 512$)
	Video Max Pixels	16,384 ($\approx 128 \times 128$)
	Trust Remote Code	True
Method	Finetuning Type	Full
	Vision Tower Frozen	False
	MM Projector Frozen	False
	DeepSpeed Stage	ZeRO-3
	Mixed Precision	bfloat16
Dataset	Total Samples	36592
	Template	`qwen3_vl`
	Cutoff Length	32,000 tokens
	Preprocessing Workers	16
	Dataloader Workers	4
Training	Batch Size per Device	1
	Gradient Accumulation Steps	1
	Effective Batch Size	256 ($= 1 \times 1 \times 256$ GPUs)
	Gradient Checkpointing	True
	Learning Rate	2.0×10^{-5}
	Epochs	8
	LR Scheduler	cosine
	Warmup Ratio	0.1
Infrastructure	Total GPUs	256 (32 nodes $\times$ 8)
	Orchestration	Ray + DeepSpeed ZeRO-3
	Placement Strategy	PACK
	Resources per Worker	1 GPU
Logging / I O	Logging Steps	5
	Checkpoint Save Steps	400
	Plot Loss	True
	Report Backend	TensorBoard

Table 5: Agentic SFT training configuration and hyperparameters for OPENSEARCH-VL. All three model sizes (8B dense, 32B dense, 30B-A3B MoE) use the identical recipe; only the *Base Model* row differs at launch time. Hyperparameters not exercised in our pipeline—e.g. held-out evaluation split, LoRA/adapter settings, or layer-freezing schedules—are omitted.

the subsequent multi-turn fatal-aware GRPO stage runs for approximately 200 optimization steps over 10 days on 64 H20s (8 nodes $\times$ 8 GPUs). The complete per-stage hyperparameter configurations are listed in the following subsections (SFT in Table 5; RL in Table 6).

C.1 SFT Training Configuration

Table 5 reports the full hyperparameter configuration used for the agentic supervised fine-tuning stage across all three OPENSEARCH-VL variants. The three model sizes (8B dense, 32B dense, 30B-A3B Mixture-of-Experts) share identical training hyperparameters—only the base checkpoint path differs—so we present them in a single consolidated table. All runs use full-parameter finetuning (including the vision tower and multi-modal projector) with DeepSpeed ZeRO-3 and Ray-based orchestration over 256 GPUs (32 nodes $\times$ 8 GPUs).

C.2 RL Training Configuration

Table 6 reports the key hyperparameters for the multi-turn fatal-aware GRPO stage across the two OPENSEARCH-VL variants we RL-finetune: **8B dense** (`Qwen3-VL-8B-Instruct`) and **30B-A3B MoE** (`Qwen3-VL-30B-A3B-Instruct`). Both runs use the same async SGLang rollout engine and

the same Megatron-parallel actor, with a shared algorithm template (RLOO leave-one-out advantage under the GRPO group-relative objective, low-variance KL controller, no critic). We list only the fields set explicitly in our launch scripts; framework defaults (e.g. optimizer betas, reward-model paths, checkpoint-resume flags) are omitted. Rows whose value is identical for both variants are written once; rows that differ are split into two cells.

D Extended Problem Formulation

This section provides a detailed formalisation of the visual deep search process that is only sketched in Sec. 2. We build up the formulation layer by layer, starting from the task setup, proceeding through trajectory structure and action/observation semantics, and ending with a compact interleaved-generation notation that serves as the conditioning object for both supervised and reinforcement learning.

An instance of the visual deep search task is formalized as a pair (I_0, q), comprising an initial image I_0 and a natural-language query q. The agent, instantiated as a multimodal policy π_θ parameterized by θ, is equipped with a predefined tool set $\mathcal{T}$. Its objective is to synthesize a terminal response $\hat{a}$ that accurately resolves q conditioned on I_0 and any subsequently retrieved evidence. Unlike classical single-turn Visual Question Answering (VQA), the policy is not constrained to immediate generation; instead, it iteratively interacts with $\mathcal{T}$, dynamically grounding its reasoning on the intermediate observations returned by the tools.

Heterogeneous Tool Set. A defining architectural choice of OPENSEARCH-VL is the *heterogeneity* of its tool set $\mathcal{T}$. We formalize a bipartite structure based on operational modality:

$$\mathcal{T} = \mathcal{T}_v \cup \mathcal{T}_s. \tag{23}$$

Here, $\mathcal{T}_v$ denotes the set of *visual* tools (CROP, SHARPEN, SUPERRESOLUTION, PERSPECTIVECORRECT, OCR) that ingest image data to produce either transformed visual artifacts or structured textual representations (as in OCR). Conversely, $\mathcal{T}_s$ denotes the set of *retrieval* tools (TEXTSEARCH, IMAGESEARCH) that interface with external web resources to yield textual evidence. This heterogeneity renders our observation space fundamentally richer and more complex than that of text-centric search agents (e.g., Search-R1 [18]), whose execution environment is typically confined to a singular text-based search engine $\mathcal{R}$.

Following the sequential decision-making paradigm articulated by AdaReasoner [45], the interaction between the policy π_θ and the execution environment unfolds as a discrete-time, multi-turn trajectory:

$$\tau = \big\{(h_0, a_0, o_0),\ (h_1, a_1, o_1),\ \ldots,\ (h_{L-1}, a_{L-1}, o_{L-1}),\ (h_L, a_L)\big\}, \tag{24}$$

where h_l, a_l, and o_l denote the history, the agent's action, and the environment's observation at step l, respectively. The terminal step L emits the final answer $\hat{a}$ and naturally lacks a subsequent observation. This progression induces a Markovian transition sequence over the expanded history space:

$$h_0 \xrightarrow{(a_0,o_0)} h_1 \xrightarrow{(a_1,o_1)} h_2 \xrightarrow{(a_2,o_2)} \cdots \xrightarrow{(a_{L-1},o_{L-1})} h_L \xrightarrow{a_L} \hat{a}. \tag{25}$$

We intentionally denote the conditioning state as history h_l rather than a compressed state s_l to underscore the non-Markovian nature of the raw generative process: the autoregressive policy continuously attends to the entire uncompressed, interleaved multimodal context.

Adopting the ReAct [17] think-then-act framework, we factorize each action into a *reasoning* component and a *control* component:

$$a_l = [\, z_l,\ c_l \,]. \tag{26}$$

Here, z_l represents a latent chain-of-thought enclosed within `<think>` $\cdots$ `</think>` tags, and c_l constitutes a structured executable command. For intermediate steps ($l < L$), c_l is a tool invocation (t, args) parameterized by $t \in \mathcal{T}$ and enclosed within `<tool_call>` $\cdots$ `</tool_call>`. For the terminal step ($l = L$), c_l is the final response $\hat{a}$ encapsulated by `<response>` $\cdots$ `</response>`. This unified parametrization of c_l streamlines the learning objective by eliminating terminal-step edge cases, directly facilitating the chain-rule factorization central to our SFT formulation (Sec. 4.1):

$$P_\theta(a_l \mid h_l) = P_\theta(z_l \mid h_l)\, P_\theta(c_l \mid h_l, z_l). \tag{27}$$

Consequent to the heterogeneity of $\mathcal{T}$, the observation space $\mathcal{O}$ is intrinsically multimodal. Given a control command c_l, the execution environment $\mathcal{E}$ deterministically routes the invocation to the

Category	Hyperparameter	8B (dense)	30B-A3B (MoE)
Model	Base Model	`Qwen3-VL-8B-Instruct`	`Qwen3-VL-30B-A3B-Instruct`
	Training Dtype	bfloat16	
Data	Train Batch Size (prompts)	256	
	Val Batch Size	64	512
	Max Prompt Length	4,096 tokens	
	Max Response Length	70,000 tokens	
	Data Seed	3407	
Rollout	Engine	SGLang (async mode)	
	Rollout Tensor Parallel	4	
	# Samples per Prompt (n)	8	16
	GPU Mem. Utilization	0.85	0.65
	Train / Val Temperature	0.7 / 0.7	
	Train / Val Top-p	1.0 / 0.95	
	Top-k	-1 (disabled)	
Policy (Actor)	Strategy	Megatron-LM	
	Tensor Parallel (TP)	4	
	Pipeline Parallel (PP)	2	
	Context Parallel (CP)	8	4
	Expert Parallel / ETP	—	8 / 1
	PPO Mini-batch Size	64	128
	PPO Max Token Len / GPU	74,576	74,596
	Micro-batch Size / GPU	1	
	Dynamic Batch Size	True	
	Param / Optim / Grad Offload	CPU	
	Gradient Checkpointing	Full recompute, uniform (1 layer)	
Optim. / Loss	Actor LR	1×10^{-6}	
	PPO Clip Ratio (high)	0.28	
	Entropy Coefficient	0.0	
	Use KL Loss	False	
	KL Loss / Controller Coef	1×10^{-3} / 1×10^{-3}	
	Loss Aggregation	`seq-mean-token-sum`	
Algorithm	Advantage Estimator	RLOO (within GRPO objective)	
	KL Type	low-variance KL	
	Fatal-aware Masking	True (unknown + error)	
Tool-Agent	# Parallel Tasks	256	
	# Parallel Tool Calls	2,048	
	Stepwise Advantage	False	
Trainer	Cluster	8 nodes $\times$ 8 GPUs = 64 GPUs	
	Save / Test Freq (steps)	50 / 10	1 / 10
	Total Epochs	100	5
	Critic Warmup	0 (critic-free)	
MoE-only (30B-A3B)	Router Dtype	—	fp32
	Auxiliary-Loss Coef	—	0.01
	Router Z-loss Coef	—	1×10^{-3}
	Permute Fusion	—	True

Table 6: Key RL training hyperparameters for the multi-turn fatal-aware GRPO stage of OPENSEARCH-VL. The 8B dense run and the 30B-A3B MoE run share the same algorithm template and rollout/actor stack; values that differ between the two (response budget per prompt, mini-batch size, context/expert parallelism, MoE-specific routing coefficients, save cadence and total epochs) are shown side-by-side. Entries marked "—" are not applicable (e.g. no expert parallelism in the dense 8B model). Framework-default fields and cluster-network environment variables are omitted.

appropriate backend, yielding:

$$o_l \;=\; \mathcal{E}(c_l, h_l) \;\in\; \begin{cases} \mathcal{O}^{\text{img}}, & \text{if } c_l \text{ invokes } t \in \mathcal{T}_v \setminus \{\text{OCR}\}, \\ \mathcal{O}^{\text{txt}}, & \text{if } c_l \text{ invokes } t \in \mathcal{T}_s \cup \{\text{OCR}\}. \end{cases} \tag{28}$$

Thus, $\mathcal{O} = \mathcal{O}^{\text{img}} \cup \mathcal{O}^{\text{txt}}$. Specifically, spatial and enhancement primitives (CROP, SHARPEN, SUPER-RESOLUTION, PERSPECTIVECORRECT) emit image tensors, whereas retrieval and parsing tools (TEXTSEARCH, IMAGESEARCH, OCR) return textual sequences. Unlike the unimodal constraints of Search-R1 [18], our agent must seamlessly condition its generation on freshly materialized visual evidence. This piecewise definition necessitates the progressive accumulation of visual context and introduces substantial optimization complexities during RL, as the policy must adapt to a dynamically expanding visual input distribution.

To accommodate newly generated visual artifacts, the agent's visual context is accumulated progressively. We define the active image set at step l as:

$$\mathcal{I}_l \;=\; \{I_0\} \cup \big\{\, o_k \;:\; k < l,\; o_k \in \mathcal{O}^{\text{img}} \,\big\}. \tag{29}$$

We enforce a *monotonically growing* constraint on $\mathcal{I}_l$: historical visual observations are strictly preserved, enabling the policy to cross-reference multi-hop visual transformations (e.g., comparing a localized CROP against its SUPERRESOLUTION-enhanced counterpart). In practice, $\mathcal{I}_l$ is materialized as an interleaved image-text prompt prefix natively processed by the underlying Qwen3-VL architecture.

Integrating the aforementioned components, the context history at step l is rigorously defined as the tuple:

$$h_l \;=\; \big(\mathcal{I}_l,\; q,\; a_{<l},\; o_{<l}\big), \tag{30}$$

where $a_{<l} = (a_0, \ldots, a_{l-1})$ and $o_{<l} = (o_0, \ldots, o_{l-1})$. The policy models the joint probability of the trajectory via standard autoregressive factorization:

$$\pi_\theta(\tau \mid I_0, q) \;=\; \prod_{l=0}^{L} P_\theta(a_l \mid h_l) \;=\; \prod_{l=0}^{L} P_\theta(z_l \mid h_l)\, P_\theta(c_l \mid h_l, z_l). \tag{31}$$

Notably, o_l is excluded from the generative probability mass, as it is an exogenous variable yielded by the environment $\mathcal{E}$; it influences the trajectory likelihood exclusively by modulating the subsequent conditioning histories $h_{l'}$ for $l' > l$.

To streamline the notation for our reinforcement learning objectives, we collapse the explicitly iterative structure of τ into a compact interleaved representation. Extending the formulation of Search-R1 [18] from a unimodal search engine $\mathcal{R}$ to our multimodal execution environment $\mathcal{E}$, we define the rollout distribution as:

$$\tau \;\sim\; \pi_\theta(\cdot \mid I_0,\, q) \bigotimes \mathcal{E}. \tag{32}$$

Here, the operator $\bigotimes$ signifies the strict interleaving of autoregressive token generation (yielding $a_l = [z_l, c_l]$) and deterministic environmental feedback (yielding $o_l = \mathcal{E}(c_l, h_l)$). This concise notation forms the basis of the expectation operators in our GRPO formulation (Sec. 4.2).

During both SFT and RL, optimization gradients must be restricted strictly to tokens emitted by the policy. For *textual observations* (originating from $\mathcal{T}_s$ and OCR), we define a masking indicator $M_{\text{gen}}(y_t) = 1$ if token y_t is constituent to a model-generated action $a_l = [z_l, c_l]$, and $M_{\text{gen}}(y_t) = 0$ if y_t belongs to an observation span o_l. Consequently, all objective functions—including the SFT negative log-likelihood and the GRPO policy gradient—are masked by $M_{\text{gen}}(y_t)$. For *image-valued observations* (from $\mathcal{T}_v \setminus \{\text{OCR}\}$), the artifacts are injected directly into the visual backbone and inherently bypass the token-level loss calculation. This masking protocol, inspired by Search-R1 [18], is critical for training stability, as textual serializations of search results and OCR parses are characteristically noisy and structurally divergent from the policy's intrinsic generative distribution.

Building upon this formulation, our optimization pipeline is strictly bifurcated. In the SFT stage (Sec. 4.1), we maximize the log-likelihood $\sum_{i,l} \log P_\theta(a_l^{(i)} \mid h_l^{(i)})$ over a corpus of high-quality expert trajectories, leveraging the chain-rule factorization of $[z_l, c_l]$. In the RL stage (Sec. 4.2), we optimize the expected return $\mathbb{E}_{\tau \sim \pi_\theta(\cdot|I_0,q) \otimes \mathcal{E}}\,[r(\tau)]$ via a multi-turn adaptation of GRPO, where gradients are propagated strictly through the policy-generated sequences within the multimodal rollout.

E Data Curation Details

This section complements Sec. 3.1 with the full operational specification of the multi-hop VQA construction pipeline and traces the pipeline end-to-end through a concrete seed page. All statistics below are computed on the `2025-05-01` snapshot of English Wikipedia.

E.1 Path Sampling Hyperparameters

We fix $\tau_{\text{hub}} = 10{,}000$ on the in-degree measured against the snapshot. Any candidate node whose incoming count exceeds τ_{hub} is skipped, rejecting roughly the top 0.03% of all article nodes. This cut-off excludes continent-, country-, and century-level pages (e.g. `United_States`, `Queensland`, `21st_century`, `English_language`) for which the uniqueness invariant in Eq. 4 is routinely violated, while preserving the long tail of entity pages that carry substantive semantic content.

Path length distribution. Path lengths are sampled as $h \sim \text{Categorical}(\{2, 3, 4\}; (0.4, 0.4, 0.2))$. Shorter paths are favoured in order to keep the downstream rollout horizons tractable; the upper bound $h = 4$ is set empirically, as walks longer than four hops rarely survive the uniqueness and non-leakage checks of Eq. 4 without heavy resampling.

Beyond the three rules stated in the main text, the walk additionally skips:

1. any title containing the substring `(disambiguation)`, or beginning with `List of`, `Outline of`, `Index of`, or `Timeline of`;
2. all non-article namespaces, i.e. `Template:`, `Category:`, `File:`, `User:`, `Help:`, `Portal:`, `Wikipedia:`;
3. redirect pages: each outgoing link is first dereferenced to its target article, and the exclusion rules above are applied to the dereferenced target rather than the surface link.

Seeds are drawn by stratified sampling across five coarse domains—*Person*, *Building/Place*, *Location (non-hub)*, *Organism*, and *Artifact*—to balance the representation of visually groundable categories. A node is eligible as a seed iff it (i) exposes an infobox; (ii) links to at least one Wikimedia Commons image of resolution no smaller than 512×512; and (iii) has in-degree in $[50, \tau_{\text{hub}}]$, so that the seed is neither a dead end nor a hub.

A walk rooted at a fixed seed is retried up to 10 times whenever it (i) hits a hub or a filtered namespace, (ii) fails to produce any descriptor for some bridge v_j that survives the LLM uniqueness evaluator, or (iii) terminates at a node v_h whose infobox contains no attribute meeting the six-token length bound. Seeds that exhaust all 10 attempts are dropped from the final pool.

E.2 Running Example: `Australia_Zoo`

To make the five stages of Sec. 3.1 concrete, we trace the pipeline end-to-end on the seed page `Australia_Zoo` with $h = 2$.

The outgoing links of `Australia_Zoo` are first partitioned by the exclusion rules of Appendix E.1. Representative entries of each partition are:

- *Retained* (entity pages, in-degree below τ_{hub}): `Bob_Irwin`, `Steve_Irwin`, `Terri_Irwin`, `Bindi_Irwin`, `Robert_Irwin`, `The_Crocodile_Hunter`, `Wildlife_Warriors`, `Beerwah,_Queensland`, `Rosedale,_Queensland`, `Angkor_Wat`, ...
- *Hub-rejected* (in-degree $> \tau_{\text{hub}}$): `Queensland`, `Brisbane`, `Australia`, `United_States`, `Zoo`.

A random walk with $h = 2$ then samples

$$\underbrace{\texttt{Australia_Zoo}}_{v_0\ \text{(anchor)}} \xrightarrow{\rho_1 = \text{“managed by”}} \underbrace{\texttt{Steve_Irwin}}_{v_1\ \text{(bridge)}} \xrightarrow{\rho_2 = \text{“spouse of”}} \underbrace{\texttt{Terri_Irwin}}_{v_2\ \text{(answer)}}.$$

From the lead paragraph of v_2, we extract the attribute *date of Australian citizenship*, yielding the short answer $a =$ “20 November 2009” (3 tokens). GPT-4o [22] is then prompted with the full path

and its relations to synthesize the canonical question

$$q_t = \text{“On what date did } \textit{Terri Irwin}\text{, the wife of } \textit{Steve Irwin}\text{—the man who took over management of } \textit{Australia Zoo} \text{ in 1991—become an Australian citizen?”}$$

Rewriting proceeds from the farthest bridge $v_1 = \texttt{Steve_Irwin}$ toward v_0. Table 7 lists the descriptor candidates proposed by GPT-4o [22] from v_1's Wikipedia context and the verdicts returned by the uniqueness evaluator. The first accepted descriptor is retained (ties broken uniformly at random). The

Descriptor for v_1	**Uniqueness verdict**
“the man who took over management of [v_0] in 1991”	✓ unique
“the wildlife documentary host killed by a stingray in 2006”	✓ unique
“the son of the founders of [v_0]”	× – Bob and Lyn had three children (Joy, Steve, Mandy)

Table 7: Descriptor candidates for the bridge node $v_1 = \texttt{Steve_Irwin}$ and the verdicts returned by the LLM uniqueness evaluator.

accepted descriptor produces the fuzzy form

$$q_f = \text{“On what date did the wife of the man who took over management of Australia Zoo in 1991 become an Australian citizen?”}$$

All three invariants of Eq. 4 are satisfied: a is preserved, the descriptor combined with “wife of” resolves uniquely to Terri Irwin, and q_f contains no surface form or alias of any node along P (e.g. neither `Steve Irwin`, `Terri Irwin`, nor `The Crocodile Hunter` appears in q_f).



Figure 6: Representative image I for the anchor $v_0 = \texttt{Australia_Zoo}$.

We then retrieve $K = 8$ candidate images from Wikimedia Commons under the query `Australia_Zoo` and rank them by CLIP [59] cosine similarity to the canonical description *“Australia Zoo entrance”*. The top-ranked image (Fig. 6, similarity 0.34) is selected as I. Replacing the anchor mention in q_f with a visual referring expression yields the final VQA instance

$$(I,\ q,\ a) = \big(I,\ \text{“On what date did the wife of the man who took over management of } \textit{the zoo in the image} \text{ in 1991 become an Australian citizen?”},\ \text{“20 November 2009”}\big).$$

The instance passes all four checks of Sec. 3.1: *masking* (q contains no entity name or alias from $\{\texttt{Australia Zoo}, \texttt{Steve Irwin}, \texttt{Terri Irwin}\}$); *uniqueness* (a GPT-4o judge given only q returns exactly one consistent answer); *visual relevance* (CLIP similarity 0.34 exceeds our 0.28 threshold); and *non-triviality*, verified jointly under the filtering process of Sec. 3.2.

E.3 Counterfactual Design Choices

We probe the necessity of the three key design choices in Sec. 3.1 by counterfactually relaxing each one on the running example above.

Anchor = Answer. Replacing I with a photograph of Terri Irwin and asking for her citizenship date reduces the task to a single reverse-image lookup: IMAGESEARCH on the new I returns the Wikipedia page of Terri Irwin directly, from which the citizenship date is read off in one step. The multi-hop structure collapses, independently of how the question is phrased.

No fuzzing. Retaining the canonical q_t as the final question—i.e. skipping the fuzzy-rewriting stage—exposes all entity names in plain text, and a single TEXTSEARCH(“Terri Irwin Australian citizenship date”) suffices to recover a. The image I becomes decorative rather than load-bearing.

No hub avoidance. If the walk were allowed to admit the hub `Queensland` as a bridge, a natural descriptor such as “the zoo located in [v_j]” would fail the uniqueness check—thousands of zoos satisfy this relation—forcing either unsuccessful resampling or a contrived descriptor that itself

leaks the identity of `Australia_Zoo`. Hub avoidance (Appendix E.1) removes this failure mode at sampling time rather than relying on the downstream filters to catch it.

F System Prompts

This section provides the complete system prompts used in OPENSEARCH-VL. The agent system prompt (Figure 7) is shared across inference and SFT data collection, and is paired with the machine-readable tool schema (Figure 11) that is injected into the model context as `<tools>...</tools>` for OpenAI-style function calling. The two reward judge prompts (Figures 8 and 9) are used during RL training to compute r_{acc} and r_{query}, respectively. For final benchmark reporting, we additionally adopt a GPT-4o judge prompt (Figure 10) that is aligned with the evaluation protocol of Vision-DeepResearch [1], so that our numbers remain directly comparable to prior multimodal deep-research work.

G Tool Definition and Usage

This section details the search-oriented tools integrated within OPENSEARCH-VL. For each tool, we articulate its core functionality, formalize its input–output signature, describe its backend implementation, and contextualize its operational role within visual search trajectories. The tool set is partitioned into three functional modalities: *Retrieval*, *Image Enhancement*, and *Attention & Parsing*. Retrieval tools interface with the open web via remote APIs; image enhancement tools operate as lightweight, deterministic local primitives; and attention & parsing tools combine local spatial priors with remote document-understanding services.

Retrieval Tools

- **TEXTSEARCH**
  - **Functionality:** A composite textual retrieval mechanism comprising three pipelined stages: (i) a Serper-backend search query to retrieve top-k candidate URLs; (ii) a JINA Reader invocation to fetch and normalize the HTML payloads into clean Markdown; and (iii) a Qwen3-32B summarization pass that distills a query-focused 2–4 sentence synopsis per document. The terminal output is a structured list of `[Passage` i`] (title, url, summary)` objects.
  - **Operational Role:** Functions as the primary "read the web" primitive when surface-level snippets are insufficient. It is indispensable for resolving complex, multi-hop knowledge queries and retrieving long-tail factual evidence that necessitates reasoning over entire paragraphs.
- **IMAGESEARCH**
  - **Functionality:** A visual-entity and reverse-image search tool powered by the Polaris Lens API. It accepts a publicly routable image URL and returns a structured JSON payload encompassing visually similar images, recognized entities, related domain URLs, and textual captions associated with the query image.
  - **Operational Role:** Serves as the critical bridge transmuting purely visual queries into retrievable textual entities. When the agent determines that resolving a query hinges on identifying an unknown visual entity (e.g., a landmark, logo, or public figure), it invokes IMAGESEARCH to extract external semantic grounding, which can subsequently be cross-referenced via TEXTSEARCH.

Image Enhancement Tools

- **SHARPEN**
  - **Functionality:** A deterministic deblurring operator implemented via OpenCV Unsharp Masking. Parameterized by an input image and an optional sharpening intensity α (default 1.5), it computes the enhanced image $I_{\text{out}} = (1+\alpha)\, I - \alpha\, G_\sigma * I$, where G_σ is a Gaussian blur kernel.
  - **Operational Role:** Deployed when the input image exhibits pervasive blur or soft edge gradients, which frequently degrade downstream optical character recognition (OCR) or object detection. As a computationally cheap, side-effect-free preprocessing step, the agent can heuristically apply SHARPEN prior to reinvoking perceptual tools.

Agent System Prompt (Inference / SFT Data Collection)

You are an advanced **Visual Investigation Agent**. Answer user questions with maximum precision by proactively using image-processing and retrieval tools.

Core Philosophy: "Verify, Don't Guess"

1. **Tool-First Mindset**: small text ⇒ `crop`; blurry ⇒ `sharpen`; tilted ⇒ `perspective_correct`. Never rely on the internal encoder when a tool gives a sharper view.
2. **Chain Your Tools**: non-trivial queries usually require a pipeline, e.g. `perspective_correct` → `crop` → `layout_parsing`.
3. **External Validation**: whenever the answer depends on facts not purely visible in the pixels, you *must* call `text_search`.

1. Tool Calling Format

Function signatures are provided inside `<tools>...</tools>`. Emit tool calls as one JSON object inside `<tool_call>{"name": <f>, "arguments": <args>}</tool_call>`, one call per turn.

2. Your Toolbox

{Tool List}

For each of the seven tools the production prompt specifies its *trigger* (when to call), *params* (JSON schema), and *output* (how to consume the return value). The tools fall into three families: *visual perception* (`crop`, `layout_parsing`); *image enhancement* (`perspective_correct`, `super_resolution`, `sharpen`); and *knowledge retrieval* (`text_search`, `image_search`; image search must be followed by text search).

3. Thinking Protocol

Before any action, emit a `<think>` block with: (i) *analyse request*; (ii) *assess image quality*—legibility, geometry, and target size, mapping any deficiency to the corresponding enhancement tool; (iii) *identify information gaps*—retrieval needed?; (iv) *formulate plan*—commit to a single next action.

Critical reminders: (a) when `layout_parsing` returns text, treat it as ground truth—do *not* fall back to visual guessing; (b) after `image_search`, always call `text_search` for factual detail; (c) `text_search` summaries are already query-focused—trust them and cite the returned URLs.

4. Workflow Recipes

- *Unreadable document*: `perspective_correct` → `sharpen` → `layout_parsing`.
- *Dense chart*: `crop` (region of interest) → `layout_parsing`.
- *Entity identification*: `image_search` → `text_search` (mandatory follow-up).

5. Output Rules

1. **Single action per turn**; wait for its result before the next.
2. **Think first**: never emit `<tool_call>` without a preceding `<think>`.
3. **Image refs**: initial image is `img_1`; each tool output yields `img_2`, `img_3`, ...; always operate on the latest best version.
4. **Final answer**: emit `<response>...</response>` once evidence suffices.

6. Execution Example (tool-use turn)

```
<think> Invoice img_1 is skewed; correct perspective first. </think>
<tool_call>{"name": "perspective_correct", "arguments": {"image":
"img_1"}}</tool_call>
```

Figure 7: Condensed agent system prompt used during both inference and SFT trajectory collection. The placeholder **{Tool List}** stands in for the per-tool description block of the production prompt; concrete *trigger*/*params*/*output* content for each of the seven tools is reproduced in Figure 11 (machine-readable schema) and Table 1 (human-readable summary). Relative to the original prompt used in our codebase, we condense the three long "critical reminder" paragraphs into a single line, drop redundant execution examples, and merge per-tool workflow sub-rules into the core philosophy; no behavioural rule is removed.

- **SUPERRESOLUTION**
  - **Functionality:** A deep-learning-based upscaling tool employing the EDSR architecture via OpenCV's `dnn_superres` module. It accepts an image and a discrete scale factor (default $\times 4$), emitting a high-resolution reconstruction. For robustness, if the EDSR weights are unavailable in the deployment environment, it gracefully degrades by returning the original image.
  - **Operational Role:** Crucial for mitigating resolution bottlenecks, particularly on tightly cropped patches or inherently low-fidelity inputs (e.g., thumbnails). By synthesizing high-frequency details, it substantially elevates the reliability and extraction accuracy of subsequent OCR invocations.

Accuracy Reward Judge Prompt (r_{acc})

You are an impartial judge evaluating whether a deep research report contains the correct answer.
[Question] {question}
[Correct Answer] {reference_answer}
[Deep Research Report] {assistant_answer}
Task: Determine if the deep research report contains the correct answer anywhere in its content.
Instructions:
1. Read through the entire research report carefully
2. Look for the correct answer anywhere in the report (it may be embedded in paragraphs, tables, or sections)
3. Check if the information in the report is consistent with the correct answer
4. The answer does NOT need to be in a specific format or labelled as "final answer"
5. Provide your reasoning
6. Answer with "yes" if the report contains the correct answer, "no" if it doesn't or contradicts it

Output format: correct: [yes/no] reasoning: [your explanation]

Figure 8: The GPT-4o judge prompt used to compute the accuracy reward $r_{\text{acc}} \in \{0, 1\}$ during RL training.

Query-Quality Reward Judge Prompt (r_{query})

You are an impartial judge evaluating the quality and utility of an agent's search trajectory.
[Original Question] {question}
[Ground Truth Answer] {ground_truth}
[Agent's Final Answer] {prediction}
[Search Trajectory] {trajectory_summary}
[Evaluation Criteria] Evaluate the overall utility of the agent's search queries and retrieved results:
1. **Image search utility**: Did image searches retrieve visual evidence that genuinely supports answering the question? Were the images relevant, or just noise?
2. **Text search utility**: Did text searches find relevant textual information? Were queries well-formed and targeted?
3. **Query progression**: Did the queries show logical progression—refining, narrowing, or covering different aspects? Or did they repeat / drift aimlessly?
4. **Complementarity**: Did image and text searches complement each other, providing evidence that one modality alone couldn't supply?
5. **Evidence vs. noise ratio**: What fraction of retrieved results actually contained useful evidence versus irrelevant content?

Score the overall query utility from 0.0 to 1.0: - 0.0: No useful information retrieved; all searches irrelevant or failed - 0.3: Mostly noise with occasional marginally relevant results - 0.5: Mixed—some useful evidence found but significant noise or inefficiency - 0.7: Good search strategy; majority of results were relevant - 1.0: Excellent—targeted, efficient queries that retrieved highly relevant evidence
Output format (strictly follow): score: [a single float between 0.0 and 1.0] reasoning: [brief explanation in 2–3 sentences]

Figure 9: The GPT-4o judge prompt used to compute the query-quality reward $r_{\text{query}} \in [0, 1]$ during RL training.

- **PERSPECTIVECORRECT**
  - **Functionality:** An automated perspective-rectification primitive. It executes a Canny edge detection pipeline on the grayscale projection, extracts the maximal quadrilateral contour, and computes a four-point perspective transform to warp the image into a fronto-parallel plane. If a reliable quadrilateral cannot be established, it falls back to the original image alongside a diagnostic warning.
  - **Operational Role:** Addresses the pervasive domain shift of real-world captures, such as skewed photographs of documents, receipts, or screens. Rectifying the image geometry dramatically improves the bounding-box precision and text-recognition fidelity of downstream OCR parsing.

Attention and Parsing Tools

- **CROP**

Benchmark Evaluation Judge Prompt (GPT-4o)

You are an impartial judge evaluating whether a deep research report contains the correct answer.
[Question] {question}
[Correct Answer] {correct_answer}
[Deep Research Report] {response}
Task: Determine if the deep research report contains the correct answer anywhere in its content.
Instructions:
1. Read through the entire research report carefully
2. Look for the correct answer anywhere in the report (it may be embedded in paragraphs, tables, or sections)
3. Check if the information in the report is consistent with the correct answer
4. The answer does NOT need to be in a specific format or labelled as "final answer"
5. Provide your reasoning
6. Answer with "yes" if the report contains the correct answer, "no" if it doesn't or contradicts it

Output format: correct: [yes/no] reasoning: [your explanation]

Figure 10: GPT-4o judge prompt used for *benchmark* evaluation of OPENSEARCH-VL and all baselines. We deliberately keep this prompt aligned with the evaluation protocol released by Vision-DeepResearch [1], so that reported accuracies are directly comparable across systems. Although structurally similar to the RL accuracy-reward judge (Figure 8), this prompt is applied *post-hoc* to final agent trajectories rather than as a training signal, and uses the field names (`question`, `correct_answer`, `response`) consumed by our evaluation script.

  - **Functionality:** A deterministic spatial-attention primitive. Parameterized by an image and a bounding box coordinate tuple (x, y, w, h), it extracts and isolates the specified rectangular sub-region, writing the artifact to disk for subsequent reference.
  - **Operational Role:** Models the human cognitive mechanism of foveating onto a dense sub-region within a cluttered scene (e.g., isolating a single chart in a multi-panel figure). It is the canonical mechanism for the policy to suppress peripheral noise prior to passing a clean, localized patch to downstream tools such as OCR, SUPERRESOLUTION, or IMAGESEARCH.

- **OCR**
  - **Functionality:** A layout-aware optical character recognition service backed by a remote PaddleX infrastructure. It consumes a base64-encoded image alongside optional flags for chart-recognition and document-orientation classification. It emits a hierarchical list of detected text blocks (categorized into titles, body text, footnotes, etc.), yielding explicit `block_label`, `block_content`, and a reading-order-reconstructed `formatted_text`.
  - **Operational Role:** Serves as the agent's primary "read the image" capability. Beyond plain character recognition, its preservation of the document's logical layout is indispensable for queries dependent on structural hierarchy (e.g., distinguishing a figure caption from a section title). It is optimally invoked following upstream enhancements (PERSPECTIVECORRECT, SHARPEN, SUPERRESOLUTION) or spatial isolation (CROP).

H Case Study

Figure 12 presents a representative end-to-end trajectory of **OpenSearch-VL** on a knowledge-intensive visual question ("*In what year did this bridge open?*"), whose answer is recoverable neither from the encoder's parametric knowledge nor from any single retrieval call. The agent first dispatches a CROP on the roadside signage to foveate on the most diagnostic sub-region, forwards the cropped patch to IMAGESEARCH to identify the structure as the *Kessock Bridge*, and then issues a targeted TEXTSEARCH that corroborates the opening year as **1982**. The trace exemplifies the compositional "verify, don't guess" behavior that the visual–retrieval action space $\mathcal{T}_v \cup \mathcal{T}_s$, the query-quality reward r_{query}, and the fatal-aware masking scheme are jointly designed to incentivize: tools are chained in an order that progressively grounds the question, and the rollout terminates as soon as cross-modal evidence converges.

Tool Definitions (OpenAI-compatible JSON schema, injected as <tools>)

```
{
  "tools": [
    {
      "type": "function",
      "function": {
        "name": "crop",
        "description": "Crop a region from an image; trigger when target covers < 30% of the
     frame.",
        "parameters": {
          "type": "object",
          "properties": {
            "image":  {"type": "string",  "description": "Image reference, e.g. 'img_1'"},
            "x":      {"type": "integer", "description": "Top-left X coordinate"},
            "y":      {"type": "integer", "description": "Top-left Y coordinate"},
            "width":  {"type": "integer", "description": "Width of the crop region"},
            "height": {"type": "integer", "description": "Height of the crop region"}
          },
          "required": ["image", "x", "y", "width", "height"]
        }
      }
    },
    {
      "type": "function",
      "function": {
        "name": "text_search",
        "description": "Serper + JINA Reader + Qwen3-32B summarisation over top-k web pages.",
        "parameters": {
          "type": "object",
          "properties": {
            "q":     {"type": "string",  "description": "Search query keywords"},
            "hl":    {"type": "string",  "description": "Language code", "default": "en"},
            "top_k": {"type": "integer", "description": "#results to summarise", "default": 5}
          },
          "required": []
        }
      }
    },
    // ---- Remaining tools follow the same schema; full specs in the tool-suite table ----
    { "function": { "name": "layout_parsing",      "required": []         } },
    { "function": { "name": "perspective_correct", "required": ["image"]  } },
    { "function": { "name": "super_resolution",    "required": ["image"]  } },
    { "function": { "name": "sharpen",             "required": ["image"]  } },
    { "function": { "name": "image_search",        "required": ["url"]    } }
  ]
}
```

Figure 11: Machine-readable tool schema produced by our agent codebase and injected into the model context as an OpenAI-style <tools> block. Two representative tools (crop for visual perception and text_search for knowledge retrieval) are shown in full; the remaining five follow the same schema and are collapsed to their name/required fields—their complete specifications are listed in Table 1. At inference time the agent emits tool calls inside <tool_call>{"name": ..., "arguments": ...}</tool_call> conforming to the parameters schema.



User Question: In which year did this bridge officially open?

Model Responses: **<think>** The user is asking about when the bridge in the image officially opened. I can see a cable-stayed bridge crossing what appears to be a body of water, with a road sign visible in the foreground. The road sign shows directions to Perth, Aberdeen, Inverness, and Fort William, which suggests this is in Scotland.
Let me first try to read the road sign more clearly to identify the bridge, then search for information about when it opened. I'll crop the road sign area to get better text recognition. **</think>**
<tool_call> {"name": "crop", "arguments": {"image": "img_1", "x": 280, "y": 280, "width": 200, "height": 150}} **</tool_call>**

Tool Output: **<observation>** Image cropped successfully. New image ID: img_3. **</observation>**

Model Responses: **<think>** **The cropped image shows part of the road sign**, but I need to get a better view of the text. Let me try to use layout parsing on this cropped image to extract the text, or I could use image search to identify the bridge directly. Given that I can see this is a cable-stayed bridge in Scotland with signs pointing to Perth, Aberdeen, Inverness, and Fort William, this is likely the Kessock Bridge near Inverness. Let me use image search to confirm the bridge's identity. **</think>**
<tool_call> {"name": "image_search", "arguments": {"url": "img_1"}} **</tool_call>**



cropped image

Tool Output: **<observation>** Tool execution result: {"summary": "<think> Okay, let's see. The user wants me to extract and summarize only the relevant "title" and "source" information from the image search results provided. They mentioned to remove all irrelevant info and keep only the essential details ... So the answer would be a list"} **</observation>**

Model Responses: **<think>** The image search result mentions "**Kessock Bridge**" in the summary. This is a cable-stayed bridge near Inverness, Scotland. Now I need to search for specific information about when the Kessock Bridge officially opened. **</think>**
<tool_call> {"name": "text_search", "arguments": {"q": "Kessock Bridge opening year when did it open", "hl": "en", "top_k": 5}} **</tool_call>**

Tool Output: **<observation>** Tool execution result: [Passage 1] Title: Kessock Bridge, URL: **https://en.wikipedia.org/wiki/Kessock_Bridge**
Summary: <think> Okay, the user is asking about the opening year of the Kessock Bridge ... Let me check the provided webpage content. It replaced a centuries-old ferry service and marked a major transport improvement for the Highlands. The bridge's 40th anniversary was celebrated in 2022, aligning with its **1982 opening year**. **</observation>**

Final Model Response: **<think>** The text search results clearly confirm that the Kessock Bridge officially opened on August 6, 1982. Multiple sources (Wikipedia, Transport Scotland, BBC, and Facebook) all confirm this date. The bridge was officially opened by The Queen Mother, and it celebrated its 40th anniversary in 2022, which aligns with the 1982 opening year.
I now have sufficient information to answer the user's question. **</think>**
<response> The bridge shown in the image is the **Kessock Bridge,** a cable-stayed bridge crossing the Beauly Firth near Inverness, Scotland. The Kessock Bridge officially opened on **6 August 1982**. It was formally inaugurated by The Queen Mother. The bridge replaced a centuries-old ferry service and celebrated its 40th anniversary in 2022.
\boxed{1982} **</response>**

Figure 12: **Case study of OpenSearch-VL.** Given an image-based question about the opening year of a bridge, the model first inspects visual evidence and crops the road sign to obtain finer-grained location cues. It then uses image search to identify the bridge as the Kessock Bridge and issues a targeted text search to verify its official opening date. The retrieved evidence confirms that the bridge opened in 1982, leading to the final answer. This example illustrates how interleaved visual inspection, image retrieval, and textual evidence acquisition can resolve knowledge-intensive visual questions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Sec. 1 state three concrete contributions—a multi-hop visual QA construction pipeline (Sec. 3), a heterogeneous tool environment (Sec. 2; Appendix G), and a multi-turn fatal-aware GRPO algorithm (Sec. 4.2; Appendix B)—together with the empirical gain of more than 10 average points across 7 benchmarks. These claims are supported by Table 2 in Sec. 5 and the ablations in Tables 4–3; we deliberately scope our claims to the seven evaluation benchmarks and the three Qwen3-VL backbones tested, and do not claim improvements on tasks outside this evaluation suite.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A dedicated "Limitations and Future Work" section is provided in Sec. 7, where we explicitly discuss the reliability of the external tool environment (Serper/JINA Reader/Qwen3-32B summariser, ranking drift, fetch failures), the proprietary nature and cost of the GPT-4o-based r_{acc} and r_{query} judges, and the fact that r_{query} currently scores only textual queries while ignoring visual operations in $\mathcal{T}_v$.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper is primarily empirical, but Appendix B.3 (Proposition 1) states and proves a single informative-dominance result for fatal-aware clamping over the hard-masking baseline of [1], with all assumptions (the per-trajectory gradient form of Eq. 21, the fatal-aware mask $M_{i,t}$ of Eq. 7, and the group-normalised, one-sided-clamped advantage of Eq. 8) made explicit in the statement and used in the proof. Appendix B.3 additionally provides an explicit accounting (Eq. 22) of the non-negative bias $b_{\mathcal{G}}$ that the clamp introduces relative to a zero-mean GRPO baseline. The remaining algorithmic derivations of GRPO with fatal-aware masking (Sec. 4.2) and its precursors (Appendix A) are stated as numbered equations together with all required assumptions (e.g., stop-gradient on group statistics, the consecutive-error threshold $K{=}3$).

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Sec. 2 specifies the data pipeline, tool environment, and the multi-turn fatal-aware GRPO objective; Appendices B–G provide the reward formalisation, fatal-detection logic, system prompts (Figs. 7–10), tool schemas, and per-stage hyperparameter tables (Tables 5, 6). Sec. 5 reports the base checkpoints, dataset sizes (SearchVL-SFT-36k, SearchVL-RL-8k), the 7 evaluation benchmarks, and the GPT-4o evaluation protocol aligned with VDR-Bench. We note that exact reproduction additionally depends on third-party services (Serper, JINA Reader, Polaris Lens, GPT-4o) whose responses can drift over time; this caveat is acknowledged in Sec. 7.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may

be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will release the SearchVL-SFT-36k and SearchVL-RL-8k training datasets, the OpenSearch-VL checkpoints (8B, 30B-A3B, 32B), and the training, inference, and evaluation code (extending LlamaFactory [37], rLLM [38], and Vision-DeepResearch [1]; see Appendix C) under permissive open licenses, together with README files and environment-setup instructions. The construction algorithms, system and judge prompts (Figs. 7–10), tool schemas (Fig. 11, Appendix G), and per-stage hyperparameters (Tables 5, 6) needed to re-derive the main results are documented in the paper and its appendix, so that the main results can be re-derived from the description alone; artifact URLs accompany the public release.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 5 states the base models, benchmarks, baselines, evaluation metric (Pass@1 with a GPT-4o judge), and dataset composition. Appendix C reports compute infrastructure, while Tables 5 and 6 list the full SFT and multi-turn fatal-aware GRPO hyperparameter configurations (optimizer, learning rates, group size G, KL coefficient β, fatal threshold K=3, reward mixing α=0.8, etc.) for every model variant.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Following the convention of recent multimodal deep-research benchmarks (e.g., VDR-Bench, MMSearch, BrowseComp-VL), we report deterministic Pass@1 with a fixed GPT-4o judge prompt and greedy/low-temperature decoding, so each cell of Table 2 corresponds to a single run. We do not report error bars because each multi-turn agent rollout already involves ≈ 6 tool calls per query and re-running 7 benchmarks across all 8B/30B-A3B/32B variants and baselines for multiple seeds is computationally prohibitive. The training-dynamics figure (Fig. 3) shows per-batch curves that already aggregate over many prompts, providing an indirect indication of variance.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: Appendix C reports that all stages run on Nvidia H20 GPUs, with agentic SFT taking $\sim$ 2 days for the 8B dense model and $\sim$ 4 days for the 30B-A3B MoE model on 256 H20s (32×8), and the multi-turn fatal-aware GRPO stage running for $\sim$ 200 optimization steps over $\sim$ 10 days on 64 H20s (8×8). Tables 5 and 6 additionally specify per-GPU batch sizes, parallelism configuration, and rollout-engine settings so that total compute can be reconstructed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We have reviewed the NeurIPS Code of Ethics and the research conforms to it: training data is constructed from the publicly available English Wikipedia snapshot together with already-released multimodal benchmarks (LiveVQA, FVQA, WebQA), no human-subject studies or crowdsourcing are involved, the double-blind review process was respected throughout submission and review, and we discuss the limitations of relying on proprietary LLM judges (Sec. 7).

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: On the positive side, OpenSearch-VL is designed as an end-to-end documented recipe whose data-construction pipeline, training procedure, and tool environment are described in full in the paper and its appendix; we will release the corresponding training data, training/inference code, and model checkpoints publicly, with the goal of lowering the entry barrier to frontier multimodal deep-search research and reducing reliance on proprietary commercial systems—a motivation made explicit in Sec. 1 and revisited in Sec. 8. On the negative side, more capable web-search agents could in principle be misused for automated information harvesting or targeted re-identification of public figures from images; we mitigate this by training only on already-public Wikipedia/benchmark content, by relying on standard search APIs that the agent invokes rather than constructing private indexes, and by inheriting the safety alignment of the underlying Qwen3-VL backbones.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released artifacts pose limited misuse risk beyond what already exists in the underlying open Qwen3-VL backbones and standard web-search APIs. The training datasets (SearchVL-SFT-36k, SearchVL-RL-8k) are derived from public Wikipedia and previously released multimodal QA benchmarks; they are filtered to remove disambiguation/list pages and hub entities (Sec. 3.1, Appendix E.1) and contain no scraped private content, faces curated for identification, or generative image content. We are therefore not introducing a new high-risk asset that requires gated release.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets are cited at first use: base models Qwen3-VL [12]; training-data sources Wikipedia [21], LiveVQA [24], FVQA [15], WebQA [25]; benchmarks SimpleVQA, VDR-Bench, MMSearch, BrowseComp-VL, InfoSeek; expert models GPT-4o [22] and Claude Opus [26]; tool back-ends (CLIP [59], EDSR via OpenCV, Serper, JINA Reader, PaddleX, Polaris Lens API); and training frameworks LlamaFactory [37], rLLM [38], and Vision-DeepResearch [1]. We use each asset within the terms of its public license/API agreement (e.g., Wikipedia/Wikimedia Commons under CC BY-SA, Qwen3-VL under its

respective open license, paid commercial APIs for GPT-4o/Claude/Serper/PaddleX) and the accompanying release will reproduce these license notices.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: This paper introduces four new assets—the SearchVL-SFT-36k and SearchVL-RL-8k training datasets, the OpenSearch-VL model checkpoints (8B / 30B-A3B / 32B), and the agent training/inference/evaluation codebase. Their construction is documented in detail in Sec. 3 and Appendix E (path sampling, fuzzy rewriting, anchor grounding, filtering, and trajectory synthesis), the tool environment in Appendix G, the system and judge prompts in Figs. 7–10, and the training configurations in Appendix C. We will release these assets publicly under permissive open licenses, accompanied by per-asset documentation (dataset/model cards, license notices, README files, setup instructions, and example samples).

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This work does not involve crowdsourcing or research with human subjects. All training data is synthesized automatically from the Wikipedia hyperlink graph and existing public benchmarks (Sec. 3), with quality control performed by LLM-based judges rather than human annotators.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: External LLMs are core methodological components and are explicitly disclosed throughout the paper: GPT-4o [22] is used to synthesise canonical questions, propose fuzzy descriptors, and serve as the answer-correctness (r_{acc}) and query-quality (r_{query}) judges (Sec. 3.1, Sec. 4.2, Appendix B.1; full prompts in Figs. 8–10); Claude Opus 4.6 [26] is used as the expert rollout model for SFT trajectory synthesis (Sec. 3.3); a frozen Qwen3-VL-32B is used as a difficulty filter and Qwen3-32B as the TEXTSEARCH summariser (Sec. 3.2, Appendix G). LLMs were also used for incidental writing and editing assistance, which does not affect the core scientific contributions.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.