SCALING VERIFIABLE ENVIRONMENTS FOR LONG-HORIZON WORK AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Work agents operate over digital artifacts to execute professional knowledge-intensive work, requiring training environments that support long-horizon interaction and trustworthy verification. However, hand-crafted environments incur prohibitive engineering overhead that prevents environment scaling, whereas synthesis methods sacrifice workspace complexity, realism, or grounded verifiability. To bridge this gap, we introduce WORKFORGE, a scalable synthesis framework for constructing verifiable work-agent environments from real-world resources. Starting from expert workflows, WORKFORGE first identifies the resources, decisions, and deliverables required by each workflow. It then retrieves relevant real-world files and organizes them into a workspace. WORKFORGE inspects the workspace to extract concrete, checkable facts about its content. These factual anchors fix which task types the workspace can support and how their outcomes can be verified. Therefore, WORKFORGE derives each task's instructions, solution plan, and complementary programmatic and semantic verifiers directly from these factual anchors, keeping verification traceable to observable workspace evidence. Furthermore, we construct 16.7K verifiable environments across 40 professional domains, with workspaces collectively covering 60 file types. Post-training Qwen3.5-35B-A3B-Base improves GDPVal from 45.5 to 73.6 and APEX Score from 5.0 to 21.3, while enabling Qwen3.5-27B to achieve highly competitive performance and outperform strong competitors. Our analyses confirm the efficacy of the proposed method and reveal consistent scaling behaviors across both data volume and interaction horizons.

1 INTRODUCTION

Autonomous agents mark a fundamental paradigm shift in large language models, transitioning foundation models from passive conversational assistants to active digital workers capable of executing complex, multi-step professional knowledge work (Drouin et al., 2024; Boisvert et al., 2024; Huang et al., 2024; Patwardhan et al., 2026; Vidgen et al., 2026). Agent assistants such as Claude Code and Codex illustrate this paradigm through iterative exploration, tool orchestration, and execution over digital workspaces (Anthropic, 2025; OpenAI, 2025).

Unlike conversational or API-calling routines, professional workflows center on concrete deliverables: users specify objectives, and the solution path is embedded within the workspace (Tang et al., 2026). Producing these deliverables requires agents to untangle multiple file dependencies and make decisions in authentic workspace artifacts (Zhou et al., 2024; Trivedi et al., 2024; Zhang et al., 2026a; Xi et al., 2026a). Cultivating agents capable of such long-horizon workflow fundamentally demands agent environments that couple authentic operational friction with trustworthy, verifiable supervision (Dou et al., 2025; Zhang et al., 2025; Sun et al., 2026; Xi et al., 2026b).

While expert-curated environments offer high fidelity, their prohibitive overhead severely impedes environment scaling (Zhou et al., 2024; Trivedi et al., 2024). Recent work addresses this bottleneck through automated environment synthesis (Song et al., 2026; Tu et al., 2026; Cheng et al., 2026; Fan et al., 2026; Zhao et al., 2026; Bai et al., 2026). However, these methods rely on synthetic environments or task-specific mock resources (Song et al., 2026; Tu et al., 2026; Bai et al., 2026), yielding toy workspaces with limited file diversity and cross-file dependencies. This reduces the exploration demands that characterize real professional work in Figure 1. Likewise, evaluation criteria

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

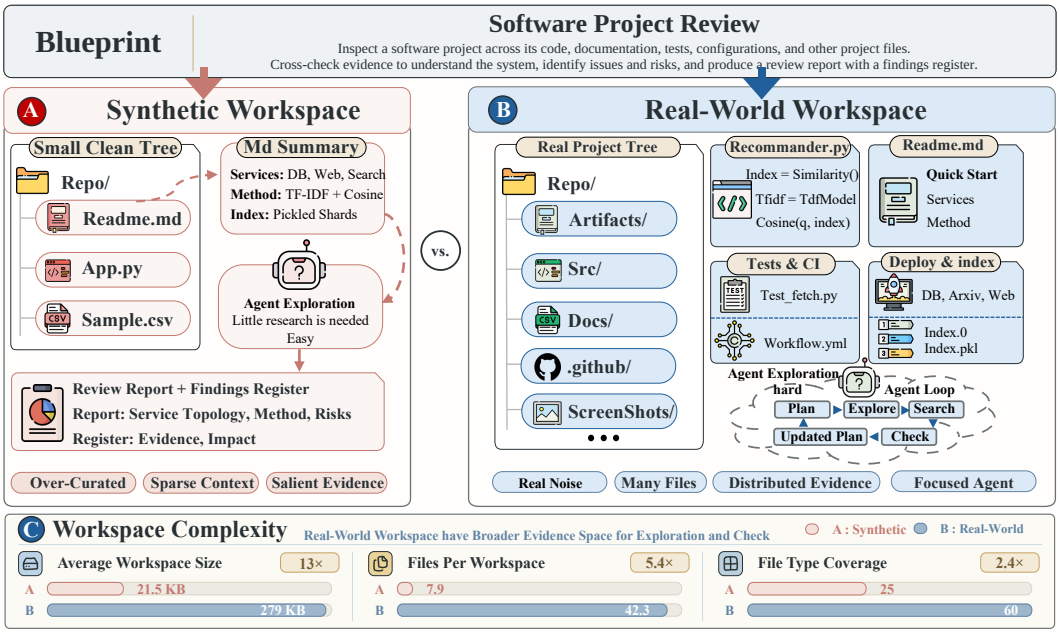

Figure 1: **Comparison of synthetic and realistic workspaces.** Realistic workspaces are substantially more complex: they contain more files, span a broader range of file types, and distribute task-relevant evidence throughout the directory hierarchy. In contrast, synthetic workspaces are highly curated, with most task-relevant information concentrated in a small set of identifiable files.

not grounded in observable workspace facts risk introducing hallucinated or otherwise unsupported requirements, leading to unreliable judges (Zhang et al., 2026b; Shi et al., 2026).

To address this challenge, we introduce WORKFORGE, a scalable framework for constructing complex, verifiable work-agent environments from real-world digital resources. WORKFORGE is motivated by a simple premise: professional tasks and their verification are inseparable from the workspaces in which they are performed. Workspace determines what an agent can explore, while observable workspace facts determine what can be asked and reliably verified. To operationalize this premise, WORKFORGE formalizes expert workflows as task blueprints, retrieves relevant resources, and profiles the resulting workspaces to recover file roles, dependencies, and factual anchors. These anchors jointly condition task generation and verifier design, keeping evaluation traceable to the workspace: programmatic checks verify deterministic requirements, while semantic rubrics assess open-ended deliverables against the same facts. Autonomous solvers then execute these tasks to yield long-horizon trajectories for post-training.

WORKFORGE instantiates 16.7K verifiable environments across 40 professional domains, with workspaces comprising real artifacts, rich file types and cross-file dependencies. Fine-tuning Qwen3.5-35B-A3B-Base (Qwen Team, 2026) on the resulting trajectories improves performance across five benchmarks, including gains of 28.1 points on GDPval and 16.3 points on APEX. Controlled studies display significant benefits with increasing volume and trajectory length, and confirm the efficacy of the proposed method. Our contributions are as follows:

- We introduce WORKFORGE, a scalable framework that constructs verifiable environments from real-world resources, grounding task specifications and verification rubrics in audited workspaces.
- We instantiate 16.7K agentic environments across 40 professional domains, featuring complex filesystem and long-horizon solver trajectories.
- Empirical experiments demonstrate the efficacy of our method. Notably, fine-tuning on WORKFORGE data yields substantial gains across five diverse agent benchmarks, improving GDPVal by 28.1 points and APEX by 16.3 points.

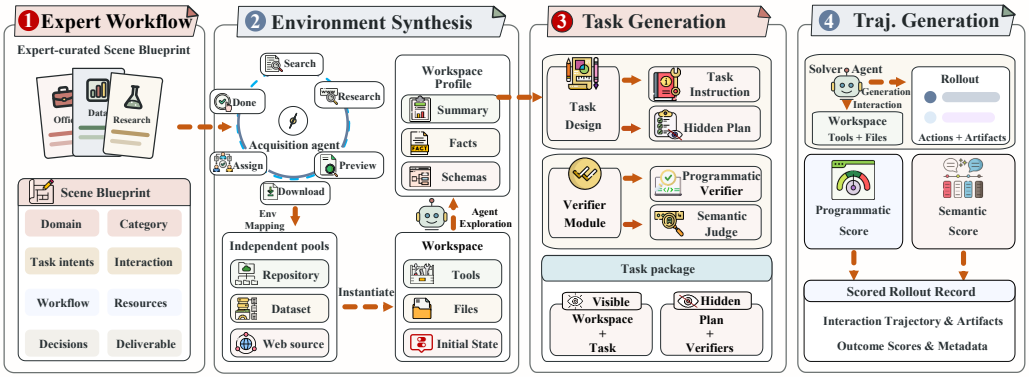

Figure 2: **Overview of WORKFORGE pipeline.** Conditioned on professional scene blueprints, WORKFORGE acquires and instantiates real-world resources into authentic workspaces, derives tasks and complementary verifiers, and collects scored trajectories through interaction.

2 WORKFORGE

To scale agent environments without compromising authenticity or verifiability, we introduce WORKFORGE, a framework for translating professional workflows into realistic, verifiable agent environments. WORKFORGE first distills domain expertise into reusable workflow specifications. It then instantiates each specification as a workspace populated with realistic artifacts and analyzes the resulting workspace evidence to ground both task construction and verification. Agents execute these tasks to produce scored trajectories for training. Figure 2 summarizes this four-stage pipeline.

2.1 WORKFLOW SPECIFICATION

To generate sophisticated professional tasks, we engage domain experts to distill their workflows into reusable *Scene Blueprints*, as these workflows identify dependencies between initial conditions, domain distinctive operations, and final deliverables. We define a collection of M blueprints as

$$\mathcal{B} = \{b_j\}_{j=1}^{M}, \qquad b_j = (\mathcal{W}_j, \mathcal{O}_j, \mathcal{R}_j, \mathcal{D}_j)\,, \tag{1}$$

where $\mathcal{W}_j$ specifies the workflow and its context; $\mathcal{O}_j$ defines the expected deliverable; $\mathcal{R}_j$ specifies required sources and their dependencies; and $\mathcal{D}_j$ records domain specific decisions.

For instance, a teacher can distill the workflow of testing students on a unit and providing targeted explanations into a scene blueprint. The blueprint specifies how to compare student responses against a standard answer and use teaching materials to group errors. It thus serves as a reusable seed for constructing workspaces and generating tasks with different assessment and teaching materials.

Each blueprint serves as a seed that connects expert intent to task generation. Its resource requirements guide source acquisition. Simultaneously, its workflow, decision points, and deliverable guide task generation using evidence from the instantiated workspace.

2.2 ENVIRONMENT SYNTHESIS

Environment synthesis integrates two modules, environment mapping and environment profiling. The former one maps each scene blueprint to different resource realizations. Complementing the mapping process, the latter one represents each environment's professional context, resource characteristics, and observable facts. For a scene blueprint b_j, the stage produces

$$\mathrm{Env}(b_j) = \{(W_{jk}, P_{jk})\}_{k=1}^{K_j}, \tag{2}$$

where K_j is the number of retained environments, W_{jk} is a workspace, and P_{jk} is its profile.

Environment Mapping. Environment mapping expresses the one-to-many relationship between a scene blueprint and its concrete environments. We acquire distinct resource pools $\{R_{jk}\}_{k=1}^{K_j}$, each

containing the files of a repository and defining a resource mapping as

$$W_{jk} = \phi_{R_{jk}}(b_j), \tag{3}$$

where $\phi_{R_{jk}}$ realizes b_j using the selected resource pool. An acquisition agent iteratively searches for sources, reads reference pages, downloads resources, and previews their contents to assess relevance and associate artifacts with the abstract source requirements $\mathcal{R}_j$. Each workspace uses one non-empty pool and retains its file organization.

Environment Profiling. Environment profiling defines the description accompanying each environment as an integral part of synthesis. We represent this profile as

$$P_{jk} = (U_{jk}, \mathcal{F}_{jk}, A_{jk}), \qquad \mathcal{F}_{jk} = \left\{ f_{jk}^{(\ell)} \right\}_{\ell=1}^{L_{jk}}, \tag{4}$$

where U_{jk} summarizes the domain, file roles, and potential task directions; $\mathcal{F}_{jk}$ contains workspace facts; and A_{jk} contains each artifact schemas and statistics. In brief, the profile describes what makes each environment suitable for particular professional tasks and supplies factual anchors for task construction and subsequent solution verification.

2.3 Task Generation

Task generation produces executable task packages with aligned task requirements and verification criteria. Task Design combines the scene blueprint and environment profile to specify the professional objective, required deliverables, and a hidden solution plan. Verifier Design translates these requirements into rubrics and complementary programmatic and semantic verifiers.

Task Design. Given a scene blueprint b_j and the profile P_{jk}, the task designer jointly generates the instruction q_{jk} which agent can view and hidden solution plan h_{jk}:

$$(q_{jk}, h_{jk}) = (f_{\text{task}}(b_j, P_{jk}), f_{\text{plan}}(b_j, P_{jk})), \tag{5}$$

Together, the instruction and plan form an executable task grounded in the concrete contents of the real-world workspace. Outputs with invalid paths or any unsupported components are regenerated.

Verification Module. We verify key factual claims in h_{jk} against the workspace W_{jk} and record the resulting evidence as $\widehat{F}_{jk}$. Along with the profile facts and task constraints, this evidence defines an atomic rubric set $\mathcal{V}_{jk} = \mathcal{V}_{jk}^{\text{prog}} \cup \mathcal{V}_{jk}^{\text{sem}}$. After execution, programmatic criteria check the final artifacts Y_{jk}:

$$S_{jk}^{\text{prog}} = \frac{1}{|\mathcal{V}_{jk}^{\text{prog}}|} \sum_{v \in \mathcal{V}_{jk}^{\text{prog}}} \mathbb{I}[v(W_{jk}, Y_{jk}) = 1]. \tag{6}$$

Semantic criteria and the task instruction q_{jk} configure the LLM judge G_{jk}^{sem}, which assesses open-ended requirements using criterion-specific scoring guides and normalizes each score $s_v \in [0, 10]$:

$$S_{jk}^{\text{sem}} = \frac{1}{|\mathcal{V}_{jk}^{\text{sem}}|} \sum_{v \in \mathcal{V}_{jk}^{\text{sem}}} \frac{s_v}{10}. \tag{7}$$

Both verifiers are applied after execution, and their scores are retained separately. The resulting task package separates visible inputs from hidden solution plan and verifiers:

$$\mathcal{T}_{jk} = \left(\underbrace{(W_{jk}, q_{jk})}_{\text{visible}}, \underbrace{\left(P_{jk}, h_{jk}, \widehat{F}_{jk}, \mathcal{V}_{jk}, G_{jk}^{\text{prog}}, G_{jk}^{\text{sem}} \right)}_{\text{hidden}} \right). \tag{8}$$

Table 1: Key statistics of WORKFORGE from task, workspace and trajectory three dimensions.

Statistic	Value
Task	
# Task number	16,665
Avg. Rubrics	21.9
Workspace	
Avg. Files	42.3
Avg. Size	279.0 KB
# File Types	60
Trajectory	
Avg. Steps	31.8
Avg. Tool Calls	33.8
Avg. Tokens	76,485

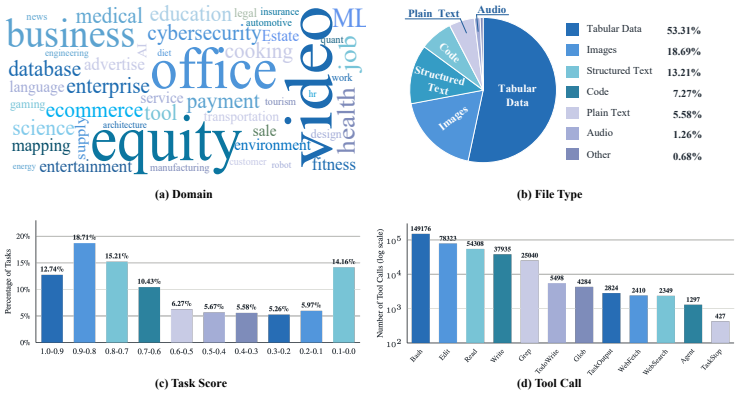

Figure 3: **Distributions of WORKFORGE.** Statistics across (a) Professional domains, (b) File types, (c) Task scores, and (d) Tool calls.

2.4 TRAJECTORY GENERATION

Trajectory generation turns static task package into observed agent behavior. For each $\mathcal{T}_{jk}$, a solver agent π_θ receives only the visible pair (W_{jk}, q_{jk}) and interacts with the workspace until it submits its deliverables or reaches the execution limit. We denote this process as

$$(\tau_{jk}, Y_{jk}) = \text{Rollout}(\pi_\theta; q_{jk}, W_{jk}), \tag{9}$$

where τ_{jk} records the ordered messages and tool interactions, and Y_{jk} denotes the output artifacts.

After execution, the graders G_{jk}^{prog} and G_{jk}^{sem} evaluate the final workspace and Y_{jk}, producing the programmatic and semantic scores defined in Eq. 6 and 7. We retain the complete execution as a scored trajectory record

$$z_{jk} = \left(\tau_{jk}, Y_{jk}, S_{jk}^{\text{prog}}, S_{jk}^{\text{sem}}, \mu_{jk}\right), \qquad \mathcal{D}_{\text{rollout}} = \{z_{jk}\}_{j,k}, \tag{10}$$

where μ_{jk} contains the model configuration. Collectively, the four stages transform professional workflows encoded in scene blueprints into independently grounded task packages and interaction trajectories paired with verifiable execution outcomes. Table 1 reports the corpus-level scalar statistics, while Figure 3 summarizes its domain coverage, file types, task score, and tool call distribution. Algorithm 1 in Appendix B summarizes the integral procedures from scene blueprints to trajectories.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Baselines. We evaluate models trained on WORKFORGE-16K against three groups of baselines. **Frontier Proprietary Models:** Claude Opus 5.0 (Anthropic, 2026), Claude Opus 4.8, GPT-5.6 Sol (OpenAI, 2026), and Gemini 3.7 Flash (Doshi, 2026). **Open-source Models (>100B):** Qwen3-Coder (Qwen Team, 2025), Qwen3.5-397B-A17B (Qwen Team, 2026), DeepSeek-V3.2, DeepSeek-V4-Flash, DeepSeek-V4-Pro (DeepSeek-AI et al., 2026), Kimi-K2.5 (Kimi Team, 2026), GLM-5.2 (Z.ai, 2026), MiniMax-M3 (MiniMax, 2026b), and MiniMax-M2.7 (MiniMax, 2026a). **Open-source Models (<100B):** OpenThinkerAgent-32B (Raoof et al., 2026), ClawGym-30A3 (Bai et al., 2026), EnvScaler-Qwen3-8B (Song et al., 2026), Nemotron-Terminal-32B (Pi et al., 2026), and Arctic-AWM-14B (Wang et al., 2026a).

Benchmarks. We evaluate work-agent capability on five popular benchmarks. **WildClaw-Bench** (Ding et al., 2026) and **Claw-Eval** (Ye et al., 2026) emphasize long-horizon execution, multi-file interaction, and reliable tool use under repeated trials. **GDPval** (Patwardhan et al., 2026) and **APEX-Agents (APEX)** (Vidgen et al., 2026) focus on professional deliverables and workplace tool orchestration, while **Agents' Last Exam (ALE)** (Sun et al., 2026) tests extended autonomous problem solving with verifiable outcomes. Additional evaluation details are provided in Appendix C.4.

Table 2: Performance (%) of agent capability on WildClaw, ClawEval, GDPval, APEX, and Agent's Last Exam. The best results are in **bold** and the second-best results are underlined within each group. Additional evaluation details are provided in Appendix C.4.

Model	WildClaw		ClawEval		GDPVal	APEX		ALE	
	Score	Pass@1	Score	Pass^3	Score	Score	Pass@1	Score	Pass@1
Frontier Proprietary Models									
Claude Opus 5.0	**64.5**	**32.4**	**82.3**	**74.2**	**91.0**	**58.5**	**42.7**	**48.3**	**26.7**
Claude Opus 4.8	64.0	30.5	81.5	72.1	89.1	46.2	34.0	46.2	24.8
GPT-5.6 Sol	53.5	29.5	77.3	68.9	90.1	51.8	37.5	**48.3**	**26.7**
Gemini 3.7 Flash	52.6	18.1	69.2	60.0	86.1	51.1	36.9	46.9	25.7
Open-source Models (>100B)									
Qwen3-Coder	32.7	15.2	37.3	23.4	67.7	11.4	2.7	21.1	5.7
Qwen3.5-397B-A17B	38.8	17.1	60.8	53.0	79.7	27.4	12.9	31.1	11.4
DeepSeek-V3.2	28.2	15.2	58.3	49.2	74.2	17.6	6.3	26.3	9.5
DeepSeek-V4-Flash	59.2	27.6	**74.4**	**66.4**	83.6	46.3	30.0	42.3	22.9
DeepSeek-V4-Pro	**60.7**	**31.4**	70.4	61.6	**88.4**	**48.5**	**32.1**	45.3	23.2
Kimi-K2.5	41.1	14.3	56.7	44.4	78.5	24.1	11.0	32.5	11.4
GLM-5.2	58.2	28.6	73.1	63.8	85.2	45.7	30.4	**47.9**	**24.8**
MiniMax-M3	56.8	28.6	70.2	55.6	86.8	41.2	26.0	34.6	17.1
MiniMax-M2.7	41.2	16.2	60.8	48.7	81.9	25.3	11.5	28.4	9.5
Open-source Models (<100B)									
OpenThinker-32B	**16.7**	5.7	31.2	17.2	21.2	**6.8**	**3.1**	2.0	0.0
ClawGym-30A3	14.9	6.7	**35.4**	**19.4**	**35.0**	2.5	0.4	**9.1**	**2.9**
EnvScaler-8B	13.7	3.8	18.7	8.2	15.0	4.9	0.8	0.7	0.0
Nemotron-Terminal-32B	8.8	2.9	12.9	1.8	5.8	1.8	0.6	1.2	0.0
Agent World Model 14B	13.1	**7.6**	26.0	16.4	24.5	3.6	0.8	1.1	0.8
Ours									
Qwen3.5-35B-A3B-Base	18.6	7.6	35.4	19.9	45.5	5.0	0.6	11.4	3.8
+ WorkForge-A3B	29.9	20.0	53.3	41.8	73.6	21.3	9.2	28.7	12.4
Qwen3.5-27B	30.5	18.1	61.9	52.0	79.4	25.7	12.7	32.7	14.3
+ WorkForge-27B	**38.5**	**23.8**	**65.4**	**55.7**	**82.4**	**29.9**	**16.3**	**33.9**	**17.1**

Implementation Details. We apply supervised fine-tuning on Qwen3.5-35B-A3B-Base and Qwen3.5-27B. Both models are trained for five epochs with a global batch size of 64 and a maximum sequence length of 256K tokens. We adopt a learning rate of 1×10^{-5} with cosine decay and 5% warmup. More implementation illustrations are in Appendix C.

3.2 Main Results

Superior performance of WorkForge over other baselines. Table 2 reports the capacity of general agents across state-of-the-art competitors. We make several key observations: (1) WorkForge delivers consistently competitive results across all benchmarks, establishing strong performance. Notably, WorkForge-A3B achieves the highest GDPval Score among prior specialized sub-100B baselines, i.e., 73.6, surpassing the previous SOTA, Clawgym-30A3, by a substantial margin of 38.6 points. Similar trends are observed on WildClaw, ClawEval, APEX, and ALE. (2) Importantly, these leads do not shrink under strict pass rate: WorkForge-A3B reaches 12.4% Pass@1 on ALE, compared with 2.9% of ClawGym-30A3 which rarely complete these challenge tasks. (3) WorkForge-A3B also delivers a substantial improvement over pretrained model Qwen3.5-35B-A3B-Base. These gains reflect WorkForge's design principle: solving each task requires the agent to explore complex, real-world file structures and integrate evidence across multiple files into its deliverables. This capability generalizes well on challenging benchmarks.

WorkForge-27B advances capability toward the frontier. WorkForge-27B demonstrates that the supervision produced by our pipeline complements existing agent post-training, improving Qwen3.5-27B across all five benchmarks, with WildClaw, APEX, and GDPval Scores rising by 8.0, 4.2, and 3.0 points, respectively. Notably, on GDPval, WorkForge-27B reaches 82.4, surpassing Qwen3.5-397B-A17B (79.7) despite nearly 15× fewer total parameters, and narrowing the gap to frontier systems such as Claude Opus 4.8 (89.1) and GPT-5.6 Sol (90.1).

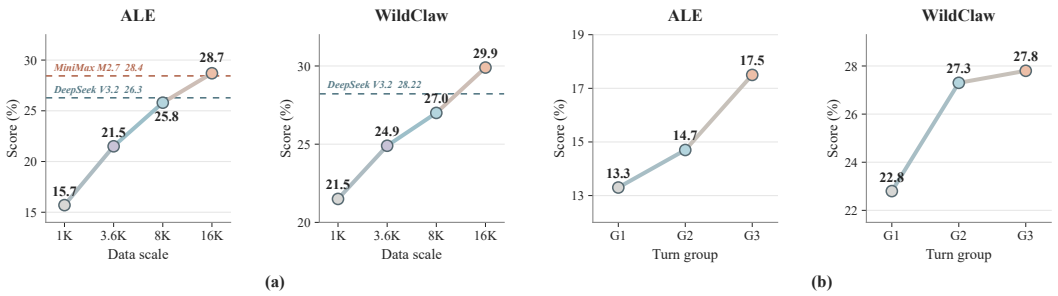

Figure 4: Scaling analysis of WORKFORGE. Performance on ALE and WildClaw with increasing (a) Training data volume and (b) Interaction length.

3.3 ABLATION STUDIES

Trajectory Harness. We further compare models trained on trajectories generated by two agent harnesses, Claude Code (Anthropic, 2025) and OpenClaw (Steinberger, 2026). In Figure 5, the model trained on Claude Code trajectories achieves a higher ALE score when evaluated with Claude Code (24.2 vs. 16.1). On WildClaw, this ordering reverses: the model trained on OpenClaw trajectories leads by 6.2 points under the OpenClaw evaluation harness. The OpenClaw-trained model also performs better on APEX, although its advantage narrows to 2.7 points under the official ReAct Toolbelt. These results indicate a degree of harness-specific transfer: aligning the training and evaluation harnesses is associated with improved performance, potentially because the model acquires harness-specific tool-use conventions in addition to task knowledge.

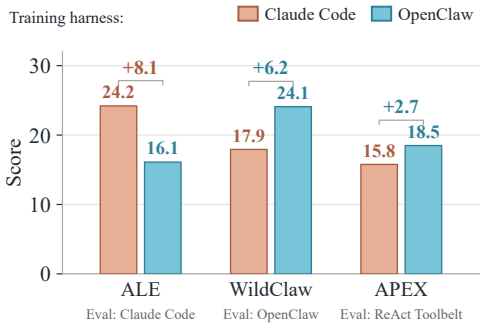

Figure 5: **Effect of trajectory harness.** We compare two training groups, each using 8k trajectories generated by Claude Code or OpenClaw, respectively.

Verification Mechanism. Recall that WORKFORGE evaluates task deliverables with two components: programmatic criteria check exact values and observable workspace states, and LLM-based criteria assess content quality. To isolate their effects on trajectory selection, we construct three 1K-trajectory training sets from the same pool of candidate tasks. The programmatic-only and LLM-only variants select trajectories whose corresponding score is at least 0.7, while the hybrid variant applies the same threshold to the equally weighted mean of the two scores. All variants use the same training-set size and fine-tuning configuration. As shown in Table 3, the hybrid strategy outperforms either variant. This indicates that combining the two verification signals yields effective supervision.

Table 3: Ablation of verification-based trajectory selection. Each variant uses trajectories selected by the semantic score, the programmatic score, or their combination.

	ALE		APEX
	Score	Pass@1	Score
LLM only	15.62	4.8	11.18
Programmatic	14.89	4.8	10.92
Hybrid	**22.11**	**11.4**	**15.39**

Workspace Authenticity. We instantiate matched workspaces from the same scene blueprints and vary only whether their resources are retrieved from the web or synthesized by an LLM. Figure 1 reveals a sharp structural gap: workspaces built from real-world resources contain 5.4× more files and 2.4× more file types, with their average footprint increasing from 21.5 KB to 279 KB. Real-world acquisition therefore enlarges both the scale and heterogeneity of the workspace, providing a richer artifact and evidence space for task construction and agent exploration.

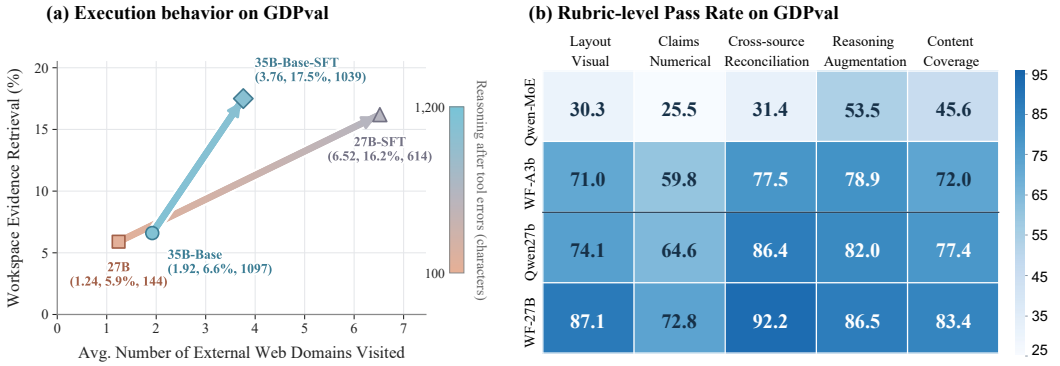

Figure 6: Execution behavior and rubric-level performance on GDPval. (a) The x-axis presents the mean number of distinct external web domains visited per task; the y-axis presents evidence retrieval within the workspace as a percentage of tool calls. Arrows connect each base model to its fine-tuned counterpart; color records reasoning length after tool errors. (b) Pass rate across five rubric categories. Details about rubric categories are described in Table 11.

3.4 Scaling Analysis

Scaling with Data Volume. To investigate how training data volume affects work-agent performance, we train Qwen3.5-35B-A3B-Base on 1K, 3.6K, 8K, and 16K trajectories under the same recipe. As shown in Figure 4 (a), performance improves monotonically with data volume. The model trained on 16K trajectories achieves scores of 28.7 on ALE and 29.9 on WildClaw, outperforming its 1K-trajectory counterpart by 13.0 and 8.4 points, respectively. At this scale, the model also surpasses DeepSeek V3.2 (26.3) and MiniMax M2.7 (28.4) on ALE, as well as DeepSeek V3.2 (28.2) on WildClaw. Increasing data volume exposes the model to a broader range of workspace structures, cross-file dependencies, and execution paths. Broader coverage promotes transferable capabilities for workspace exploration and deliverable production.

Scaling with Interaction Horizon. Beyond data volume, we explore how interaction horizon shapes agent behavior. We construct three subsets of 1K Claude Code trajectories, grouped by trajectory length: 0–20 turns (G1), 21–40 turns (G2), and 41–60 turns (G3). As shown in Figure 4 (b), performance increases with trajectory length on both benchmarks. The model trained on G3 achieves scores of 17.5 on ALE and 27.8 on WildClaw, exceeding its G1 counterpart by 4.2 and 5.0 points, respectively. This improvement is accompanied by more sustained interaction with the environment. On WildClaw, the G3-trained model averages 24.0 productive tool calls, compared with 11.7 for the G1-trained model, and performs 38.3 distinct actions, compared with 17.0. Table 13 further shows that the G3-trained model uses a broader range of tool types and invokes workspace skills more frequently. These results suggest that longer training trajectories expose the model to more complete sequences of dependency tracing, evidence gathering, and deliverable construction, thereby promoting the sustained interaction required in complex workspaces.

3.5 Behavior Analysis

We analyze the trajectory behavior of WorkForge-27B. Figure 6 (a) reveals a shift toward exploration behavior: the fine-tuned model visits 6.52 external web domains per task (5.3× the base model's average) and lifts targeted evidence retrieval from 5.9% to 16.2% of tool calls. After a failed tool call, reasoning length grows from 144 to 614 characters (over fourfold), reflecting deliberate diagnosis and replanning rather than immediate retries. These shifts carry through to final deliverables: Figure 6 (b) shows category-score gains of 4.5–13 points across five GDPval-derived categories. Consistent with this shift toward exploration behavior, the largest gains appear in formatting and visual presentation (+13 pp) and numerical and calculation accuracy (+8.2 pp), which rely on thorough investigation and careful verification.

4 Related Work

Agent Environment Synthesis. Automated environment construction expands the supply of executable settings for agent training. EnvScaler and Agent World Model synthesize tool interfaces and stateful backends (Song et al., 2026; Wang et al., 2026a), while Agent-World incorporates real-service metadata and external resources (Dong et al., 2026). Skill and intent-driven pipelines construct tasks and supporting workspaces for terminal and general-assistant workflows (Cheng et al., 2026; Bai et al., 2026). NexForge compiles capability requirements into task directives, then retrieves or constructs the resources needed for execution (Zhao et al., 2026). WorkForge instead instantiates and profiles realistic workspaces before specifying concrete tasks. Preserving repository and dataset organization grounds professional tasks in heterogeneous materials, existing file relationships, and observable evidence.

Long-Horizon Agent Data Synthesis. Agentic data synthesis expands multi-step execution through tool interactions (Prabhakar et al., 2025; Xu et al., 2025), task composition (Shi et al., 2025; Xie et al., 2025), and exploration-based task discovery (Ramrakhya et al., 2025; Chen et al., 2026; Wang et al., 2026b). Nemotron-Terminal studies generation, filtering, and long-context training (Pi et al., 2026), while OpenThoughts-Agent examines task sources and diversity to train OpenThinkerAgent-32B (Raoof et al., 2026). Executable-state alignment (Shi et al., 2026) and rubric-based artifact assessment (Starace et al., 2025; Zhuge et al., 2025) connect trajectory quality to task outcomes. WorkForge generates long-horizon supervision through sustained exploration, cross-file reasoning, and artifact production in content-rich workspaces. Task requirements and verification criteria share the same workspace evidence, enabling complementary programmatic and semantic evaluation to select trajectories for agent post-training.

5 Conclusion, Limitations and Future Work

Conclusion We introduce WorkForge, a scalable framework for constructing complex, verifiable work-agent environments from real-world resources. Its core principle is to use observable workspace facts as shared anchors for task construction and outcome verification. Starting from expert workflows, WorkForge retrieves and profiles real-world resources, then derives task specifications, solution plans, and complementary programmatic and semantic verifiers from the resulting factual anchors. The pipeline instantiated 16.7K environments across 40 professional domains, which improves the capability of Qwen3.5-35B-A3B-Base, including gains of 28.1 points on GDPval and 16.3 points on APEX. Our analysis indicates significant gains with increasing data volume and trajectory length, while confirming the efficacy of verification mechanism.

Limitations and Future Work WorkForge depends on expert workflows and accessible realistic resources, limiting comparable grounding in proprietary or sensitive domains. Because workspace profiles, tasks, and verifiers are constructed automatically, their reliability depends on the completeness and accuracy of workspace profiling. Moreover, the current verifiers emphasize final deliverables, receive limited expert calibration, and do not directly assess intermediate agent behavior. Future work should broaden workflow and resource coverage while developing verification that is aware of processing.

AI Use Statement

Generative AI tools were used to assist and polish manuscript writing. They were also used within WorkForge to generate synthetic task packages and training trajectories, as detailed in the method and experimental sections. The authors review all AI-assisted outputs and take responsibility for the final content of this work.

Ethics Statement

This paper introduces a scalable framework for constructing complex, verifiable work-agent environments from real-world digital resources. The methods and findings presented are intended solely for research purposes; our study does not pertain to malicious applications, unintended uses, or issues related to fairness, privacy, security, crowdsourcing, or human subject research.

REFERENCES

Anthropic. Claude 3.7 sonnet and claude code, February 2025. URL https://www.anthropic.com/news/claude-3-7-sonnet.

Anthropic. Introducing Claude Opus 5, July 2026. URL https://www.anthropic.com/news/claude-opus-5.

Fei Bai, Huatong Song, Shuang Sun, Daixuan Cheng, Yike Yang, Chuan Hao, Renyuan Li, Feng Chang, Yuan Wei, Ran Tao, Bryan Dai, Jian Yang, Wayne Xin Zhao, and Ji-Rong Wen. ClawGym: A scalable framework for building effective claw agents, 2026. URL https://arxiv.org/abs/2604.26904.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault Le Sellier De Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. WorkArena++: Towards compositional planning and reasoning-based common knowledge work tasks. In *Advances in Neural Information Processing Systems*, volume 37, pp. 5996–6051, 2024. doi: 10.52202/079017-0195.

Aili Chen, Chi Zhang, Junteng Liu, Jiangjie Chen, Chengyu Du, Yunji Li, Ming Zhong, Qin Wang, Zhengmao Zhu, Jiayuan Song, Ke Ji, Junxian He, Pengyu Zhao, and Yanghua Xiao. DIVE: Scaling diversity in agentic task synthesis for generalizable tool use, 2026. URL https://arxiv.org/abs/2603.11076.

Zihao Cheng, Hongru Wang, Zeming Liu, Xinyi Wang, Xiangrong Zhu, Yuhang Guo, Wei Lin, Jeff Z. Pan, and Yunhong Wang. Terminal-World: Scaling terminal-agent environments via agent skills, 2026. URL https://arxiv.org/abs/2605.20876.

DeepSeek-AI et al. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Shuangrui Ding, Xuanlang Dai, Long Xing, Shengyuan Ding, Ziyu Liu, JingYi Yang, Penghui Yang, Zhixiong Zhang, Xilin Wei, Xinyu Fang, Yubo Ma, Haodong Duan, Jing Shao, Jiaqi Wang, Dahua Lin, Kai Chen, and Yuhang Zang. WildClawBench: A benchmark for real-world, long-horizon agent evaluation, 2026.

Guanting Dong, Junting Lu, Junjie Huang, Wanjun Zhong, Longxiang Liu, Shijue Huang, Zhenyu Li, Yang Zhao, Xiaoshuai Song, Xiaoxi Li, Jiajie Jin, Yutao Zhu, Hanbin Wang, Fangyu Lei, Qinyu Luo, Mingyang Chen, Zehui Chen, Jiazhan Feng, Ji-Rong Wen, and Zhicheng Dou. Agent-World: Scaling real-world environment synthesis for evolving general agent intelligence, 2026. URL https://arxiv.org/abs/2604.18292.

Tulsee Doshi. Introducing Gemini 3.7 Flash, August 2026. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/introducing-gemini-3-7-flash.

Shihan Dou, Jiazheng Zhang, Jianxiang Zang, Yunbo Tao, Weikang Zhou, Haoxiang Jia, Shichun Liu, Yuming Yang, Shenxi Wu, Zhiheng Xi, Muling Wu, Rui Zheng, Changze Lv, Limao Xiong, Shaoqing Zhang, Lin Zhang, Wenyu Zhan, Rongxiang Weng, Jingang Wang, Xunliang Cai, Yueming Wu, Ming Wen, Yixin Cao, Tao Gui, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. Multi-programming language sandbox for llms. In Pushkar Mishra, Smaranda Muresan, and Tao Yu (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 40–50. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-DEMO.5. URL https://doi.org/10.18653/v1/2025.acl-demo.5.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. WorkArena: How capable are web agents at solving common knowledge work tasks? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11642–11662. PMLR, 2024. URL https://proceedings.mlr.press/v235/drouin24a.html.

Zhiyuan Fan, Tinghao Yu, Yuanjun Cai, Jiangtao Guan, Yun Yang, Dingxin Hu, Jiang Zhou, Xing Wu, Zhuo Han, Feng Zhang, and Lilin Wang. Toward scalable terminal task synthesis via skill graphs, 2026. URL https://arxiv.org/abs/2604.25727.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20271–20309. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24y.html.

Kimi Team. Kimi K2.5: Visual agentic intelligence, 2026. URL https://arxiv.org/abs/2602.02276.

MiniMax. MiniMax M2.7: Early echoes of self-evolution, March 2026a. URL https://www.minimax.io/news/minimax-m27-en.

MiniMax. MiniMax M3: Frontier coding, 1M context, native multimodality—all in one model, June 2026b. URL https://www.minimax.io/blog/minimax-m3.

OpenAI. Introducing codex, May 2025. URL https://openai.com/index/introducing-codex/.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition, July 2026. URL https://openai.com/index/gpt-5-6/.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simon Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie Kim, Samuel Miserendino, Gildas Chabot, David Li, Patrick Chao, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. GDPval: Evaluating AI model performance on real-world economically valuable tasks. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/290c2430f91912204f30bbcc990fff1d-Paper-Conference.pdf.

Renjie Pi, Grace Lam, Mohammad Shoeybi, Pooya Jannaty, Bryan Catanzaro, and Wei Ping. On data engineering for scaling LLM terminal capabilities, 2026. URL https://arxiv.org/abs/2602.21193.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/5e3661f7fe4c8ac5652d62eb3d3c96ea-Paper-Datasets_and_Benchmarks_Track.pdf.

Qwen Team. Qwen3-Coder: Agentic coding in the world, July 2025. URL https://qwen.ai/blog?id=qwen3-coder.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Ram Ramrakhya, Andrew Szot, Omar Attia, Yuhao Yang, Anh Nguyen, Bogdan Mazoure, Zhe Gan, Harsh Agrawal, and Alexander Toshev. Scaling synthetic task generation for agents via exploration, 2025. URL https://arxiv.org/abs/2509.25047.

Negin Raoof, Richard Zhuang, Marianna Nezhurina, Etash Guha, Atula Tejaswi, Ryan Marten, Charlie F. Ruan, Tyler Griggs, Alexander Glenn Shaw, Hritik Bansal, E. Kelly Buchanan, Artem Gazizov, Reinhard Heckel, Chinmay Hegde, Sankalp Jajee, Daanish Khazi, Emmanouil Koukoumidis, Xiangyi Li, Hange Liu, Shlok Natarajan, Harsh Raj, Nicholas Roberts, Ethan Shen, Nishad Singhi, Michael Siu, Ashima Suvarna, Hanwen Xing, Patrick Yubeaton, Robert Zhang, Leon Liangyu Chen, Xiaokun Chen, Steven Dillmann, Saadia Gabriel, Xunyi Jiang, Anurag Kashyap, Boxuan Li, Yein Park, Minh Pham, Sujay Sanghavi, Lin Shi, Ke Sun, Yixin Wang, Zhiwei Xu, Erica Zhang, Siyan Zhao, Wanjia Zhao, Jenia Jitsev, Alex Dimakis, Benjamin Feuer,

and Ludwig Schmidt. OpenThoughts-Agent: Data recipes for agentic models, 2026. URL https://arxiv.org/abs/2606.24855.

Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, Jian Yang, Ge Zhang, Jiaheng Liu, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. TaskCraft: Automated generation of agentic tasks, 2025. URL https://arxiv.org/abs/2506.10055.

Kou Shi, Zun Wang, Qisheng Su, Shiting Huang, Ziao Zhang, Zhen Fang, Qingnan Ren, Jin Liu, Yu Zeng, Yiming Zhao, Lin Chen, Zehui Chen, and Feng Zhao. FACET: Preserving source intent and executable state in terminal task synthesis, 2026. URL https://arxiv.org/abs/2608.18580.

Xiaoshuai Song, Haofei Chang, Guanting Dong, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. EnvScaler: Scaling tool-interactive environments for LLM agent via programmatic synthesis, 2026. URL https://arxiv.org/abs/2601.05808.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. *arXiv preprint arXiv:2504.01848*, 2025. doi: 10.48550/arXiv.2504.01848.

Peter Steinberger. Introducing OpenClaw, January 2026. URL https://openclaw.ai/blog/introducing-openclaw.

Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, Weishu Zhang, Tyler Zeng, Ying Yan, Bo Liu, Hanson Wen, Mingyang Xu, Xiaoyuan Liu, Zimeng Chen, Weiyan Shi, Amanda Dsouza, Vincent Sunn Chen, Dawn Song, Patrick Bryant, Carl Boettiger, Yamini Rangan, Bradley Rothenberg, Kyle Steinfeld, Arvind Rao, Tapio Schneider, Georgios Yannakakis, Laure Zanna, Kaan Ozbay, Ida Sim, Tarek Zohdi, George Em Karniadakis, Jack Gallant, Teresa Head-gordon, et al. Agents' last exam. *arXiv preprint arXiv:2606.05405*, 2026. doi: 10.48550/arXiv.2606.05405.

Zirui Tang, Xuanhe Zhou, Yumou Liu, Linchun Li, Yukai Wu, Weizheng Wang, Hongzhang Huang, Wei Zhou, Jun Zhou, Jiachen Song, Shaoli Yu, Jinqi Wang, Zihang Zhou, Hongyi Zhou, Yuting Lv, Jinyang Li, Jiashuo Liu, Ruoyu Chen, Chunwei Liu, GuoLiang Li, Jihua Kang, and Fan Wu. Workspace-Bench 1.0: Benchmarking AI agents on workspace tasks with large-scale file dependencies, 2026. URL https://arxiv.org/abs/2605.03596.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 16022–16076, 2024. doi: 10.18653/v1/2024.acl-long.850.

Dunwei Tu, Hongyan Hao, Hansi Yang, Yihao Chen, Yi-Kai Zhang, Zhikang Xia, Yu Yang, Yueqing Sun, Xingchen Liu, Furao Shen, Qi Gu, Hui Su, and Xunliang Cai. ScaleEnv: Scaling environment synthesis from scratch for generalist interactive tool-use agent training, 2026. URL https://arxiv.org/abs/2602.06820.

Bertie Vidgen, Austin Mann, Abby Fennelly, John Wright Stanly, Lucas Rothman, Marco Burstein, Julien Benchek, David Ostrofsky, Anirudh Ravichandran, Debnil Sur, Neel Venugopal, Alannah Hsia, Isaac Robinson, Calix Huang, Olivia Varones, Daniyal Khan, Michael Haines, Austin Bridges, Jesse Boyle, Koby Twist, Zach Richards, Chirag Mahapatra, Brendan Foody, and Osvald Nitski. APEX-Agents, 2026.

Zhaoyang Wang, Canwen Xu, Boyi Liu, Yite Wang, Siwei Han, Zhewei Yao, Huaxiu Yao, and Yuxiong He. Agent world model: Infinity synthetic environments for agentic reinforcement learning, 2026a. URL https://arxiv.org/abs/2602.10090.

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Pei Chen, Ziwei Dong, Jing Huang, Jiri Gesi, Xianfeng Tang, Chen Luo, Qun Liu, Yisi Sang, Hanqing Lu, Manling Li, Jin Lai, and Dakuo Wang. Trajectory2Task: Training robust tool-calling agents with synthesized yet verifiable data for complex

user intents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 44021–44044, 2026b. doi: 10.18653/v1/2026.acl-long.2037.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Jiaqi Liu, Honglin Guo, Yajie Yang, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. AgentGym-RL: An open-source framework to train LLM agents for long-horizon decision making via multi-turn RL. In *International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=ZgCCDwcGwn.

Zhiheng Xi, Dingwen Yang, Jiaqi Liu, Jixuan Huang, Honglin Guo, Baodai Huang, Tinggang Chen, Qi Zhang, Zhonghang Lu, Chenyu Liu, Jiajun Sun, Jiazheng Zhang, Dingwei Zhu, Xin Guo, Junzhe Wang, Zhihao Zhang, Yuming Yang, Junjie Ye, Minghe Gao, Dongrui Liu, Jiaming Ji, Guohao Li, Tao Gui, Qi Zhang, and Xuanjing Huang. AgentGym2: Benchmarking large language model agents in de-idealized real-world environments. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44451–44479. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.acl-long.2058. URL https://aclanthology.org/2026.acl-long.2058/.

Jingxu Xie, Dylan Xu, Xuandong Zhao, and Dawn Song. AgentSynth: Scalable task generation for generalist computer-use agents, 2025. URL https://arxiv.org/abs/2506.14205.

Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. TOUCAN: Synthesizing 1.5m tool-agentic data from real-world MCP environments, 2025. URL https://arxiv.org/abs/2510.01179.

Bowen Ye, Rang Li, Qibin Yang, Yuanxin Liu, Linli Yao, Hanglong Lv, Zhihui Xie, Chenxin An, Lei Li, Lingpeng Kong, Qi Liu, Zhifang Sui, and Tong Yang. Claw-Eval: Towards trustworthy evaluation of autonomous agents, 2026.

Z.ai. GLM-5.2: Built for long-horizon tasks, June 2026. URL https://z.ai/blog/glm-5.2.

Jiazheng Zhang, Wenqing Jing, Zizhuo Zhang, Zhiheng Xi, Shihan Dou, Rongxiang Weng, Jiahuan Li, Jingang Wang, Mingxu Chai, Shibo Hong, Tao Gui, and Qi Zhang. Two minds better than one: Collaborative reward modeling for LLM alignment. *CoRR*, abs/2505.10597, 2025. doi: 10.48550/ARXIV.2505.10597. URL https://doi.org/10.48550/arXiv.2505.10597.

Jiazheng Zhang, Ziche Fu, Junrui Shen, Yunbin Zhao, Yunke Zhang, Zhiheng Xi, Long Ma, Chenxin An, Zhihao Zhang, Shichun Liu, Dingwei Zhu, Shihan Dou, Shaofan Liu, Han Li, Wiggin Zhou, Aiden Adams, Tao Gui, Fei Huang, Qi Zhang, and Xuanjing Huang. Entropy polarity in reinforcement fine-tuning: Direction, asymmetry, and control, 2026a. URL https://arxiv.org/abs/2605.11775.

Jiazheng Zhang, Ziche Fu, Zhiheng Xi, Wenqing Jing, Mingxu Chai, Wei He, Guoqiang Zhang, Chenghao Fan, Chenxin An, Wenxiang Chen, Zhicheng Liu, Haojie Pan, Dingwei Zhu, Tao Gui, Qi Zhang, and Xuanjing Huang. AgentV-RL: Scaling reward modeling with agentic verifier. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 23078–23100. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.findings-acl.1156. URL https://aclanthology.org/2026.findings-acl.1156/.

Jiarong Zhao, Zhikai Lei, Zhiheng Xi, Rui Zheng, Hang Yan, Jie Zhou, Qin Chen, and Liang He. NexForge: Scaling agent capabilities through requirement-driven task synthesis for LLMs, 2026. URL https://arxiv.org/abs/2607.14186.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/4410c0711e9154a7a2d26f9b3816d1ef-Paper-Conference.pdf.

Mingchen Zhuge, Changsheng Zhao, Dylan R. Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-judge: Evaluate agents with agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80569–80611. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhuge25a.html.

APPENDIX

A DOMAIN TAXONOMY AND DISTRIBUTION

A.1 DOMAIN TAXONOMY

WorkForge organizes 16,665 SFT tasks into 40 domains, as shown in Figure 7. The taxonomy spans common workplace activities such as office productivity, data management, business operations, finance, and communication, while also covering software, scientific, media, healthcare, security, and infrastructure-oriented tasks. The included workflows range from document and data processing to technical investigation and domain-specific decision making.

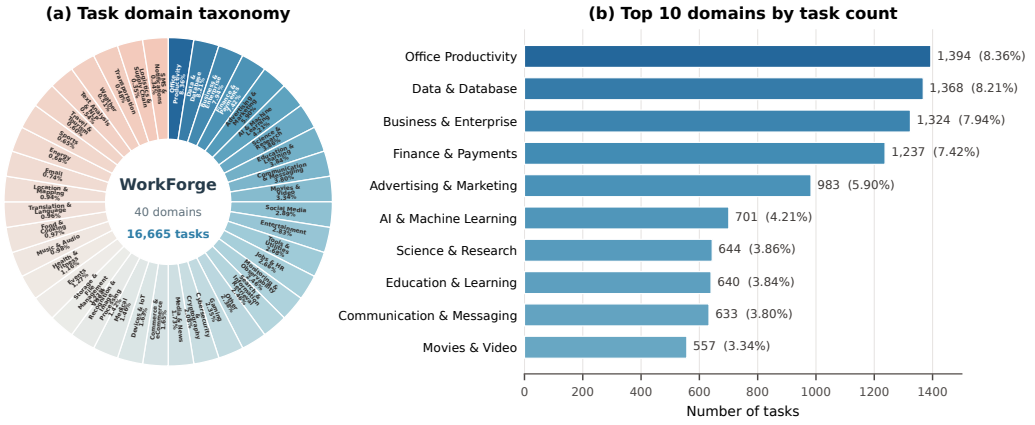

Figure 7: **Domain taxonomy and task distribution of the WorkForge SFT corpus.** The left panel shows all 40 domains, and the right panel reports the ten most represented domains.

A.2 DOMAIN DISTRIBUTION

Table 4 lists the complete distribution. The four most represented domains account for 31.9% of the corpus, and the top ten account for 56.9%. The remaining 30 domains contribute 43.1% of all tasks. The distribution combines repeated coverage of broadly applicable workplace domains with coverage across specialized applications.

B PIPELINE PSEUDOCODE AND PROMPT TEMPLATES

B.1 PIPELINE PSEUDOCODE

Algorithm 1 presents the end-to-end construction procedure and follows the notation introduced in Section 2.

B.2 TASK SYNTHESIS PROMPT TEMPLATES

Braced fields denote values injected at runtime. *Template scope.* The following revised, method-level templates specify the pipeline described in Section 2; they are not a verbatim record of prompts

Table 4: **Complete domain distribution.** Counts are computed over 16,665 SFT tasks.

#	Domain	Count	%	#	Domain	Count	%
1	Office Productivity	1,394	8.36	21	Commerce & eCommerce	275	1.65
2	Data & Database	1,368	8.21	22	Devices & IoT	271	1.63
3	Business & Enterprise	1,324	7.94	23	Medical	243	1.46
4	Finance & Payments	1,237	7.42	24	Visual Recognition & Image Processing	237	1.42
5	Advertising & Marketing	983	5.90	25	Storage & File Management	212	1.27
6	AI & Machine Learning	701	4.21	26	Events	211	1.27
7	Science & Research	644	3.86	27	Health & Fitness	194	1.16
8	Education & Learning	640	3.84	28	Music & Audio	163	0.98
9	Communication & Messaging	633	3.80	29	Food & Cooking	162	0.97
10	Movies & Video	557	3.34	30	Translation & Language	160	0.96
11	Social Media	482	2.89	31	Location & Mapping	156	0.94
12	Entertainment	471	2.83	32	Email	123	0.74
13	Tools & Utilities	447	2.68	33	Energy	113	0.68
14	Jobs & HR	443	2.66	34	Sports	108	0.65
15	Monitoring & Observability	410	2.46	35	Travel & Tourism	100	0.60
16	Search & Information Retrieval	410	2.46	36	Text Analysis & NLP	90	0.54
17	Other	396	2.38	37	Weather	85	0.51
18	Gaming	391	2.35	38	Transportation	80	0.48
19	Cybersecurity & Cryptography	346	2.08	39	Logistics & Supply Chain	59	0.35
20	Media & News	289	1.73	40	SMS & Notifications	57	0.34

Algorithm 1: Authority-Separated Task Construction and Trajectory Generation in WORKFORGE

Input: Expert-authored professional workflow specifications; resource-acquisition interface $\mathcal{A}_{\text{ground}}$; solver policy π_θ

Output: Grounded task packages $\mathcal{T}$; scored rollout corpus $\mathcal{D}_{\text{rollout}}$

$\mathcal{T} \leftarrow \varnothing$; $\mathcal{D}_{\text{rollout}} \leftarrow \varnothing$; **// Phase 1: Workflow Specification**

Obtain Scene Blueprints $\mathcal{B} = \{b_j\}_{j=1}^{M}$;

foreach *Scene Blueprint* $b_j \in \mathcal{B}$ **do**

  // Phase 2: Environment Synthesis

  $\{R_{jk}\}_{k=1}^{K_j} \leftarrow \text{GroundResourcePools}(b_j; \mathcal{A}_{\text{ground}})$;

  foreach *resource pool* R_{jk} **do**

    $W_{jk} \leftarrow \text{Instantiate}(R_{jk})$;

    $P_{jk} \leftarrow \text{Explore}(W_{jk}; b_j)$;

    // Phase 3: Task Generation and Verification Design

    Synthesize task instruction q_{jk} and solution plan h_{jk} from b_j and P_{jk};

    $\widehat{F}_{jk} \leftarrow \text{VerifyFacts}(h_{jk}; W_{jk}, P_{jk})$;

    $(\mathcal{V}_{jk}^{\text{prog}}, \mathcal{V}_{jk}^{\text{sem}}) \leftarrow \text{BuildRubrics}(q_{jk}, h_{jk}, P_{jk}, \widehat{F}_{jk})$;

    $\mathcal{V}_{jk} \leftarrow \mathcal{V}_{jk}^{\text{prog}} \cup \mathcal{V}_{jk}^{\text{sem}}$;

    Construct G_{jk}^{prog} from $\mathcal{V}_{jk}^{\text{prog}}$ and G_{jk}^{sem} from q_{jk} and $\mathcal{V}_{jk}^{\text{sem}}$;

    $\mathcal{T}_{jk} \leftarrow ((W_{jk}, q_{jk}), (P_{jk}, h_{jk}, \widehat{F}_{jk}, \mathcal{V}_{jk}, G_{jk}^{\text{prog}}, G_{jk}^{\text{sem}}))$;

    $\mathcal{T} \leftarrow \mathcal{T} \cup \{\mathcal{T}_{jk}\}$;

    // Phase 4: Trajectory Generation

    $(\tau_{jk}, Y_{jk}) \leftarrow \text{Rollout}(\pi_\theta; q_{jk}, W_{jk})$;

    $S_{jk}^{\text{prog}} = \dfrac{1}{|\mathcal{V}_{jk}^{\text{prog}}|} \displaystyle\sum_{v \in \mathcal{V}_{jk}^{\text{prog}}} \mathbb{I}[v(W_{jk}, Y_{jk}) = 1], \qquad S_{jk}^{\text{sem}} = \dfrac{1}{|\mathcal{V}_{jk}^{\text{sem}}|} \displaystyle\sum_{v \in \mathcal{V}_{jk}^{\text{sem}}} \dfrac{s_v}{10}$;

    $\mathcal{D}_{\text{rollout}} \leftarrow \mathcal{D}_{\text{rollout}} \cup \{(\tau_{jk}, Y_{jk}, S_{jk}^{\text{prog}}, S_{jk}^{\text{sem}}, \mu_{jk})\}$;

return $\mathcal{T}, \mathcal{D}_{\text{rollout}}$;

used in historical runs. Platform-specific API arguments and search scheduling details are omitted. Scene Blueprints are supplied as workflow specifications, rather than generated by the acquisition agent.

B.3 RESOURCE ACQUISITION

Resource Acquisition System Prompt

```
You are a Resource Acquisition Agent. Given a Scene Blueprint, acquire distinct, domain-
    relevant resource pools for instantiating workspaces.

## Resource Pools and Workspace Boundaries
- A resource pool is a coherent, non-empty repository or multi-file resource package whose
    files naturally belong together.
- Keep candidate pools separate. Each retained pool instantiates one independent workspace;
     do not merge unrelated downloads into one workspace.
- Preserve the retained pool's file organization and dependencies.
- Search for alternative resource realizations of the blueprint, rather than inventing a
    task and manufacturing matching inputs.

## Acquisition Procedure
1. Use the blueprint's workflow, source requirements, dependencies, and expected
    deliverables to guide searches.
2. Read source pages, download candidate packages, and preview their actual contents before
     retaining them.
3. Check that files are accessible, relevant, and usable; reject error pages and empty or
    broken downloads.
4. Associate inspected artifacts with blueprint resource requirements and record their
    roles and source locations.
5. Report missing resources or unsupported workflow steps. Do not synthesize supporting
    data, configurations, or reference facts to fill those gaps.

## Completion
Use the provided acquisition budget to search and inspect candidates. Finish when suitable
    pools have been retained or the budget is exhausted.
Return the retained pools and unresolved resource requirements. Do not claim that every
    blueprint step is supported when the inspected resources do not support it.
Only non-empty, usable pools proceed to workspace instantiation and profiling.
```

Resource Acquisition User Prompt Template

```
## Scene Blueprint
Role: {role}
Business context: {business_context}
Workflow narrative: {task_narrative}
Resource requirements and dependencies: {steps_display}

## Current State
Inspected files ({n_files}):
{files_summary}
Requirements supported: {grounded_count}/{total_steps}
Search attempts: {search_attempts}
Acquisition round: {round_num}/{max_rounds}

## Available Operations
- search: locate candidate real-world resource packages.
- research: read reference pages to assess domain relevance and interpret source
    documentation.
- download_kaggle, download_huggingface, download_url, clone_github: retrieve resources
    using identifiers or URLs observed in source pages or search results.
- preview_file: inspect downloaded content before retention or assignment.
- assign_step: associate an inspected artifact with a blueprint requirement as data_source,
     working_environment, reference_template, or config.
- done: finish acquisition and report retained pools and remaining gaps.

Choose an operation supported by the acquisition interface. Never invent source identifiers
     or use a download's filename alone as evidence of relevance.
For each retained pool, report its source, local files, artifact roles, supported
    requirements, and any unmet requirements.
Keep different pools identifiable so that each can be instantiated as a separate workspace.
```

B.4 WORKSPACE EXPLORATION

Workspace Exploration System Prompt

```
You are a Workspace Exploration Agent. Inspect the instantiated workspace and produce an
     evidence-grounded profile for task and verifier design.

## Exploration
1. Inventory file organization, types, dependencies, and documentation.
2. Inspect the key artifacts and identify their professional context and roles.
3. For data, examine schemas, value ranges, missing values, and cross-file relationships;
     for code, inspect interfaces and dependencies; for documents, identify relevant
     sections and claims.
4. Use tools to verify numerical and structural facts. Record source paths and reproducible
      queries, computations, or document locations.
5. Identify task directions, constraints, and challenges supported by the inspected
     evidence. Mark unresolved assumptions rather than presenting them as facts.

## Output
Return a structured profile with three fields:
- understanding: domain context, file roles, dependencies, and supported task directions (U
     ).
- facts: observable workspace facts with identifiers, source locations, and verification
     traces (F).
- profile: per-artifact schemas, structure, statistics, and inspection limitations (A).

Separate source observations from interpretation. Record numerical units and precision; do
     not guess values from truncated previews.
The profile describes the existing workspace. Do not alter source files to make a proposed
     task or expected answer feasible.
```

Workspace Exploration User Prompt Template

```
## Workspace Exploration Task
Domain direction: {domain}
Task archetype: {archetype}

Inspect the provided workspace to determine:
1. Which files, interfaces, and dependencies are available.
2. Which professional workflows these resources support.
3. Which numerical, structural, and documentary facts can be verified.
4. Which cross-file relationships and constraints may matter for task completion.

Start with the file inventory, then inspect the relevant artifacts in depth.
Return understanding, facts, and profile as defined in the system prompt. Include evidence
     locations and explicitly state any inspection gaps.
```

B.5 WORKSPACE-CONDITIONED TASK DESIGN

Task Design System Prompt

```
You are a professional task designer. Given a Scene Blueprint and an inspected workspace
     profile, jointly produce an agent-visible task instruction and a hidden solution plan.

## Task Requirements
- Ground the task in files, data features, interfaces, and dependencies confirmed in the
     workspace profile.
- Specify the professional objective, intended audience, required deliverables, and
     acceptance conditions in the task instruction.
- Require meaningful exploration, cross-file reasoning, or professional decisions where the
      actual workspace supports them. Do not force arbitrary difficulty quotas.
- Distinguish constraints stated in the instruction from constraints discoverable in the
     provided workspace. Do not introduce evaluator-only requirements that cannot be
     inferred from either.
- Preserve feasible alternative methods: a hidden solution plan is a reference route, not
     the only acceptable execution path.

## Output Separation
- task_md: the instruction q that the solver receives, including deliverable paths under
     results/.
```

```
- solution_plan: the hidden plan h, including relevant source paths, evidence dependencies,
      proposed reasoning steps, and factual claims to verify.

Check that every referenced input path exists in the workspace profile and that the task is
      supported by its contents. Revise unsupported components before returning the
     instruction and plan.
Do not include the hidden plan, reference answers, or verifier internals in task_md.
```

Task Design User Prompt Template

```
## Scene Blueprint
{scene_blueprint}

## Workspace Profile
{workspace_context}

## Design Context
Archetype: {archetype} ({archetype_description})
Domain: {domain}
{format_table}
{diversity_rules}

Design a realistic task for a {archetype_role} using the inspected workspace.
Return task_md and solution_plan separately. Include only requirements supported by the
     blueprint and workspace evidence; do not fabricate missing resources to satisfy format
      or diversity preferences.
For any challenge not explicitly stated in task_md, identify the workspace evidence through
      which the solver can discover it.
```

B.6 Fact Verification

Fact Verification System Prompt

```
You are a Fact Verification Agent. Check the key factual claims in the hidden solution plan
      against the actual workspace and its profile before rubric construction.

## Verification Scope
- Use tools to verify numerical claims, schemas, file structures, dependencies, and other
     observable facts on which acceptance criteria depend.
- Record each claim's source path and reproducible query, computation, or document location
     .
- Mark each claim as confirmed, corrected, or unverifiable.
- For corrected claims, supply the observed value and identify the affected plan or task
     requirements.
- For unverifiable claims, do not invent an answer or permit them to become grading ground
     truth. Revise unsupported task or plan components before constructing dependent
     criteria.

Do not treat a subjective quality judgment or a preferred implementation method as a
     verified fact. Open-ended requirements are assessed by semantic criteria, not by an
     invented exact answer.

## Output
Return verified_facts and plan_issues.
Each fact record contains id, claim, status, verified_value, source, computation, and
     usable_for_rubric.
Set usable_for_rubric to true only for confirmed or corrected claims supported by recorded
     workspace evidence. Include units and relevant precision for numerical values.
These records constitute the verified evidence F_hat used in rubric construction; retain
     them outside the solver-visible inputs.
```

B.7 Rubric Generation and LLM Judge

Rubric Generation System Prompt

```
You generate atomic evaluation criteria from the task instruction, workspace profile,
     hidden solution plan, and verified evidence.

## Dimensions
```

```
- DC (Deliverable Completeness): required artifacts exist and have valid formats and
    structures.
- FA (Factual Accuracy): task-relevant claims agree with verified source evidence.
- AD (Analysis Depth): reasoning satisfies the task's domain-specific analytical
    requirements.
- HC (Hidden Constraint): handling of task-relevant constraints discoverable in the
    provided workspace, even when not repeated in the instruction.
- DQ (Deliverable Quality): outputs are usable by the audience specified in the task.
- NP (Negative Penalty): absence of fabrication, contradiction, or critical task-relevant
    errors.

## Rules
1. ATOMIC: Each criterion evaluates one independently assessable requirement.
2. GROUNDED: Link each criterion to a task requirement or a necessary validity condition
    supported by accessible workspace evidence. The hidden plan alone cannot authorize a
    requirement.
3. DISCOVERABLE: HC criteria must identify the workspace evidence that exposes the
    constraint. Exclude challenges known only to the evaluator.
4. OBSERVABLE: State what evidence in the submitted artifacts or final workspace supports
    satisfaction or violation.
5. METHOD-AGNOSTIC: Accept valid alternative methods unless a method is explicitly required
     by the task or necessary under documented workspace constraints.

Partition criteria into programmatic and semantic sets. Programmatic criteria return binary
     outcomes; semantic criteria use criterion-specific scores from 0 to 10.
Retain dimension, polarity, and must/nice tags as descriptive metadata; they do not
    introduce extra weights or deductions into the arithmetic means defined in the paper.
For NP criteria, higher scores mean fewer or less severe errors. Do not subtract these
    scores again.
```

Rubric Generation User Prompt Template

```
## Task Instruction
{task_md}

## Workspace Profile
{workspace_context}

## Hidden Solution Plan
{solution_plan}

## Domain Context
{domain_context}

## Verified Evidence
{verified_facts}

## Candidate Workspace-Discoverable Constraints
{hidden_constraints}
Treat these as candidates, not authoritative facts. Include a candidate only if the solver
    can infer it from the instruction or identifiable workspace evidence.

## Required Output Files
{expected_files}
Use only deliverables required by task_md; the hidden plan must not add mandatory files.

## Construct Two Criterion Sets
Programmatic criteria:
- Check deterministic properties such as required file validity, structure, or verified
    values.
- For factual checks, reference a usable verified_fact id and specify how to extract the
    result from the submitted artifacts or workspace.
- State units and any justified tolerance, including whether it is absolute or relative. Do
     not apply a blanket tolerance to unrelated metrics.
- Use semantic assessment where validity cannot be determined by a reliable binary check.

Semantic criteria:
- Provide a criterion, evidence anchors, scoring_note, and failure_mode for each item.
- Calibrate 0-10 scores to task-specific satisfaction; use intermediate scores for partial
    satisfaction.
- Cover relevant analytical, discoverable-constraint, quality, and error-avoidance
    requirements without forcing a fixed count per dimension.
- Exclude unverifiable expectations, duplicate checks, and requirements supported only by
    the hidden plan.
```

```
Return the programmatic and semantic sets separately. Ensure both are non-empty for a task
    package evaluated by the two-component scheme; revise the task or candidate criteria
    if meaningful criteria for either component cannot be constructed.
```

LLM Judge System Prompt

```
You evaluate an agent's submitted deliverables after execution using the provided task
    instruction, semantic criteria, and supporting evidence.

## Scoring
- Score each semantic criterion independently from 0 to 10 using its scoring_note and
    failure_mode.
- For positive criteria, 0 means unsatisfied or incorrect, intermediate values indicate
    partial satisfaction, and 10 means full satisfaction of the specified criterion.
- For NP criteria, 10 means the specified error is absent; lower scores indicate increasing
     severity. Higher scores always mean better performance.
- Base each score on identifiable content in the artifacts or final workspace and the
    criterion's evidence anchors. Do not reward unsupported assertions or infer missing
    work.
- Do not add requirements beyond the provided criteria or demand the hidden plan's
    particular implementation route.
- Do not perform an additional weighting, penalty subtraction, or combination with
    programmatic scores.

Return the per-criterion scores and concise evidence-based reasons. The evaluator
    normalizes each score by 10 and averages over the semantic criteria, as defined in the
     paper; programmatic scores are retained separately.
```

LLM Judge User Prompt Template

```
## Agent-Visible Task
{task_context}

## Semantic Criteria and Evidence
{criteria_blocks}
Each block contains the criterion, scoring guidance, relevant submitted content, and any
    source evidence needed to assess it.

Evaluate only the listed criteria. If a criterion cannot be assessed from the supplied
    content, report the evidence gap and request the missing material before completing
    the scores.
If the evidence gap cannot be resolved, stop this evaluation without producing an aggregate
     task score; do not assign an arbitrary score or omit the criterion from the
    denominator.
Once the evidence is sufficient, return exactly {num_criteria} entries, one per criterion,
    with id, score, and a concise reasoning field citing the relevant artifact or evidence
    .
Use the supplied criterion IDs. Keep programmatic check outcomes outside this response.
```

C Experimental, Training, and Evaluation Details

C.1 Supervised Fine-Tuning Setup

We perform full-parameter supervised fine-tuning on 16,665 trajectories from WorkForge-16k. The training trajectories are generated by GLM 5.3 with the Claude Code and OpenClaw harnesses. We train Qwen3.5-35B-A3B-Base and the post-trained Qwen3.5-27B model for five epochs using the same optimization recipe. Table 5 reports the training configuration.

C.2 Training Infrastructure and Data Formatting

Training uses the `verl` supervised fine-tuning trainer with Ray 2.46, PyTorch 2.7.1, CUDA 12.8, Transformers 5.10.4, and FlashAttention 2.8.1. We use FSDP with Ulysses sequence parallelism of size four. The Qwen3.5-35B-A3B-Base runs use 64 NVIDIA H20 96GB GPUs, and the Qwen3.5-27B runs use 128 GPUs. Optimizer states are offloaded to CPU, while model parameters remain on GPU.

Table 5: Supervised fine-tuning configuration used for the main models and training-data ablations.

Configuration	Value
Optimizer	AdamW
Adam coefficients	$\beta_1 = 0.9$, $\beta_2 = 0.95$
Peak learning rate	1×10^{-5}
Weight decay	0.1
Gradient clipping	1.0
Learning-rate schedule	Cosine decay to 1×10^{-7}
Warmup ratio	5% of total steps
Training epochs	5
Global batch size	64 sequences
Micro-batch size	1 sequence per GPU
Maximum sequence length	262,144 tokens
Numerical precision	BF16 mixed precision
Sequence packing	Enabled
Dynamic batching	Enabled
Gradient checkpointing	Enabled

Training examples are rendered incrementally from cumulative conversation prefixes so that the reasoning content associated with each assistant turn is preserved. The loss is applied according to the per-message assistant loss mask stored with each trajectory. Variable-length examples are packed without padding, with sequence boundaries retained through cumulative sequence-length indices. Dynamic batching limits each GPU batch by its token count, with a maximum budget of 262,144 tokens.

C.3 Inference Configuration

We serve local models with vLLM and use the same decoding configuration for the WorkForge models and the re-evaluated open-source baselines in Table 2. Table 6 lists the shared settings. The maximum context length is 262,144 tokens, whereas the 32,768-token limit applies to each generated model response. For models whose native context window is shorter than 262,144 tokens, we use YaRN position extrapolation to extend the context window to the shared limit. Specifically, EnvScaler-Qwen3-8B, Arctic-AWM-14B, Nemotron-Terminal-32B, and OpenThinkerAgent-32B use the Qwen3 architecture with the YaRN settings in Table 7.

Table 6: Decoding configuration used for model evaluation.

Parameter	Value
Temperature	0.7
Top-p	0.9
Top-k	20
Maximum generated tokens per response	32,768
Maximum context length	262,144

Table 7: Shared YaRN configuration for the Qwen3-based baselines.

Parameter	Value
Maximum position embeddings	262,144
RoPE base (θ)	1,000,000
RoPE scaling type	YaRN
Scaling factor	6.4
Original maximum position embeddings	40,960

C.4 BENCHMARK PROTOCOLS

All results in Table 2 are obtained through our evaluation runs on WildClawBench, Claw-Eval, GDPval, APEX-Agents, and Agents' Last Exam. Table 8 summarizes their metrics and sampling protocols. Pass@1 denotes the fraction of tasks passed in a single rollout under each benchmark's success criterion. Claw-Eval uses three independent rollouts per task and reports Passˆ3, the fraction of tasks passed in all three rollouts. Unlike Pass@3, which requires at least one successful rollout, Passˆ3 measures consistency across repeated runs. All other benchmarks use one rollout per task. WildClawBench and Claw-Eval use OpenClaw; GDPval and ALE use Claude Code; APEX-Agents uses its official ReAct Toolbelt agent. Table 9 lists the harnesses, agent-level wall-clock limits, and interaction budgets.

Table 8: Benchmark-specific metrics and sampling protocols.

Benchmark	**Evaluation set**	**Reported metrics**	**Rollouts/task**
WildClaw	Full set	Score, Pass@1	1
Claw-Eval	Full set	Score, Passˆ3	3
GDPval	201 text-only tasks	Rubric score	1
APEX	Full set	Score, Pass@1	1
ALE	Full set	Score, Pass@1	1

Table 9: Evaluation harnesses and per-task agent execution budgets.

Benchmark	**Harness**	**Wall-clock limit**	**Interaction limit**
WildClaw	OpenClaw	24,000 s	Not set
Claw-Eval	OpenClaw	3,600 s	50 turns
GDPval	Claude Code	14,400 s	300 turns
APEX	ReAct Toolbelt	14,400 s	250 steps
ALE	Claude Code	28,800 s	Unlimited

C.5 GDPVAL EVALUATION SUBSET

We evaluate all models on the same 201-task subset of GDPval's official 220-task evaluation set, excluding 19 multimodal tasks. Tasks are aligned using the evaluation payload's `case_id`, which corresponds to the official `task_id`. We report the official rubric score rather than the win-or-tie rate. Table 10 lists the excluded tasks and their occupations.

C.6 GDPVAL RUBRIC-LEVEL ANALYSIS

We group GDPval criteria into five deliverable-quality categories and compute the pass rate within each category. For category c and model m, let $\mathcal{I}_c$ denote its criteria and $z_j^{(m)} \in \{0, 1\}$ indicate whether criterion j is satisfied. The category pass rate is

$$R_c^{(m)} = \frac{\sum_{j \in \mathcal{I}_c} z_j^{(m)}}{|\mathcal{I}_c|}. \tag{11}$$

We report $100R_c^{(m)}$ as a percentage and compute each gain as $100(R_c^{(\mathrm{SFT})} - R_c^{(\mathrm{Base})})$ percentage points. Table 11 reports the number and share of criteria assigned to each category together with the resulting pass rates.

Table 10: Multimodal tasks excluded from the GDPval evaluation set. Row numbers refer to the official 220-task set.

Row	Task ID	Occupation
12	38889c3b-e3d4-49c8-816a-3cc8e5313aba	Audio and Video Technicians
13	ff85ee58-bc9f-4aa2-806d-87edeabb1b81	Audio and Video Technicians
14	4b894ae3-1f23-4560-b13d-07ed1132074e	Audio and Video Technicians
56	e222075d-5d62-4757-ae3c-e34b0846583b	Film and Video Editors
57	c94452e4-39cd-4846-b73a-ab75933d1ad7	Film and Video Editors
58	75401f7c-396d-406d-b08e-938874ad1045	Film and Video Editors
59	a941b6d8-4289-4500-b45a-f8e4fc94a724	Film and Video Editors
79	4d1a8410-e9c5-4be5-ab43-cc55563c594c	First-Line Supervisors of Office & Admin Support
102	b9665ca1-4da4-4ff9-86f2-40b9a8683048	Industrial Engineers
111	46fc494e-a24f-45ce-b099-851d5c181fd4	Mechanical Engineers
127	3baa0009-5a60-4ae8-ae99-4955cb328ff3	News Analysts, Reporters, and Journalists
134	e6429658-4de1-42dd-a9e0-2d2b9b02fb10	Nurse Practitioners
147	57b2cdf2-ad62-4591-aa91-aad489740320	Private Detectives and Investigators
149	a46d5cd2-55fe-48fa-a4c6-6aaf6b9991b5	Private Detectives and Investigators
153	e4f664ea-0e5c-4e4e-a0d3-a87a33da947a	Producers and Directors
171	11593a50-734d-4449-b5b4-f8986a133fd8	Real Estate Sales Agents
172	94925f49-36bc-42da-b45b-61078d329300	Real Estate Sales Agents
188	f2986c1f-2bbf-4b83-bc93-624a9d617f45	Pharmacists
203	fe0d3941-e32c-4bf1-a643-b566d2b4cb3c	Sales Representatives, Wholesale & Manufacturing

Table 11: GDPval rubric pass rates across five deliverable-quality categories.

Category	Criteria	Share	Base	SFT	Δ
Visual / layout quality	371	4.2%	74.1%	87.1%	+13 pp
Numerical claims	523	5.9%	64.6%	72.8%	+8.2 pp
Cross-source reconciliation	103	1.2%	86.4%	92.2%	+5.8 pp
Reasoning / argumentation	200	2.2%	82.0%	86.5%	+4.5 pp
Content coverage	7,718	86.6%	77.4%	83.4%	+6.0 pp

D Additional Ablation Analyses

The training-data ablations use Qwen3.5-35B-A3B-Base with the supervised fine-tuning recipe in Table 5.

D.1 Data Volume Scaling

We train models at four data scales: 1K, 3.6K, 8K, and 16K trajectories. For each target scale, we independently sample the requested number of trajectories from the full corpus; subsets at different scales are not nested. The 16K setting uses all 16,665 trajectories. Figure 4(a) reports the results on ALE and WildClaw.

D.2 Trajectory-Length Scaling

We construct the turn-scaling subsets from trajectories generated with the Claude Code harness. Trajectories are grouped into three disjoint intervals by their number of turns, and we randomly sample 1,000 trajectories from each interval (Table 12). Figure 4(b) reports performance on ALE and WildClaw, while Table 13 summarizes tool-use behavior on WildClaw.

Table 12: Training subsets for trajectory-length scaling.

Group	Turns per trajectory	Trajectories
G1	0–20	1,000
G2	21–40	1,000
G3	41–60	1,000

Table 13: Tool-use behavior under trajectory-length scaling on WildClaw. Longer-turn supervision increases exploration intensity, breadth, and productive execution.

Group	Productive calls	Skill discovery	Tool calls	Distinct actions	Tool types used
G1	11.7	21.0%	17.7	17.0	3.3
G2	23.1	21.9%	38.3	33.0	3.6
G3	**24.0**	**33.3%**	**42.0**	**38.3**	**4.0**

D.3 Verification-Filter Ablation

We construct three 1,000-trajectory training sets from the same pool of completed rollouts using the filtering rules in Table 14. The LLM-only variant selects trajectories by the semantic score S^{sem}, the programmatic-only variant uses S^{prog}, and the hybrid variant uses their equally weighted mean. Each variant applies a threshold of 0.7 to its selection score. The downstream results on ALE and APEX are reported in Table 3.

Table 14: Trajectory selection rules for the verification-filter ablation.

Variant	Selection rule	Trajectories
LLM only	$S^{\text{sem}} \geq 0.7$	1,000
Programmatic only	$S^{\text{prog}} \geq 0.7$	1,000
Hybrid	$0.5S^{\text{sem}} + 0.5S^{\text{prog}} \geq 0.7$	1,000

E Data Examples

E.1 Cross-Region Factor-Premium Analysis

This Finance & Payments task asks a quantitative strategist to assess factor premia and prepare an investment-committee briefing. WorkForge constructs the workspace from the public Kaggle dataset `nikitamanaenkov/famafrench-factors-and-portfolios`, a local mirror of the Kenneth French data library with an August 2025 vintage. The example connects workspace exploration, evidence-grounded verification, cross-region analysis, and the production of three linked deliverables.

E.1.1 Task Instruction

Task Instruction

```
You are a quantitative strategist at a multi-asset manager. The investment committee
is weighing a new factor-tilt allocation and wants evidence on whether the classic
equity factors still earn premia -- and where (US vs developed vs emerging markets).
Using the firm's local mirror of the Kenneth French data library, build the analysis
and a decision-ready briefing.

Assess each major factor's long-run and recent performance, compare premia across
regions over a fair common window, and recommend which factors and regions deserve
capital (and which to avoid). Be explicit about data caveats that affect your
conclusions.

Deliver:
```

```
- results/factor_briefing.pptx -- committee briefing with findings and a clear
  recommendation
- results/factor_analysis.xlsx -- supporting performance tables
- results/factor_returns_tidy.csv -- the cleaned return series behind your analysis

Explore the workspace to determine structure, coverage, and pitfalls yourself.
```

E.1.2 Workspace and Source Materials

The workspace contains 297 CSV files in 16 directories, with a maximum nesting depth of six. It includes factor series, sorted portfolios, research breakpoints, and industry returns, organized into US, Developed, Emerging, Industry, and Other branches. Momentum data also appear outside the US factor directory.

Workspace Layout

```
workspace/
|-- data/
|   |-- US/
|   |   |-- Factors/                          [5 files]
|   |   |-- Portfolios/
|   |   |   |-- Bivariate/                    [49 files]
|   |   |   `-- Univariate/                   [18 files]
|   |   `-- Research_Breakpoints_Data/        [8 files]
|   |-- Developed/
|   |   |-- Factors/                          [36 files]
|   |   `-- Portfolios/                       [114 files]
|   |-- Emerging/
|   |   |-- Factors/                          [2 files]
|   |   `-- Portfolios/                       [7 files]
|   |-- Industry/                            [24 files]
|   `-- Other/                               [34 files]
`-- results/                                 [initially empty]
```

Individual files contain descriptive preambles and multiple frequency blocks. For example, the US three-factor file stacks monthly observations and annual returns in the same CSV:

Source Excerpt: F-F_Research_Data_Factors.csv

```
This file was created using the 202508 CRSP database.
The 1-month TBill rate data until 202405 are from Ibbotson Associates.
Starting from 202406, the 1-month TBill rate is from ICE BofA US 1-Month Treasury Bill
     Index.

,Mkt-RF,SMB,HML,RF
192607,   2.89,  -2.55,  -2.39,   0.22
192608,   2.64,  -1.14,   3.81,   0.25
...
202508,   1.85,   3.87,   4.41,   0.38

 Annual Factors: January-December
1927,  29.47,  -2.46,  -3.75,   3.12
1931, -44.84,   3.07, -13.69,   1.07
1933,  56.49,  44.61,  31.28,   0.30
```

Other files declare missing-value sentinels of -99.99 or -999. The US files identify CRSP as their source, whereas Developed and Emerging files identify Bloomberg. The solver must separate frequency blocks, handle missing values, reconcile factor names such as `Mom` and `WML`, and determine overlapping valid observations across regions.

E.1.3 Workspace Evidence and Verification Criteria

Workspace inspection produces 12 usable fact records with computation references. One record corrects the assumption that international series share a July 1990 start: the Developed series begins in July 1990, but the Emerging five-factor file begins in July 1989. The final factual criterion references this corrected record.

Selected Workspace Fact Records

```
vf01: US three-factor monthly series starts at 192607.
      First row: 192607, 2.89, -2.55, -2.39, 0.22.
vf03: The US three-factor CSV contains a monthly block followed by an annual block.
vf07: File headers declare missing values as -99.99 or -999.
vf09: US files use CRSP; Developed and Emerging files use Bloomberg.
vf12 (corrected): Developed starts at 199007; Emerging five-factor data start at 198907.
      Computation: inspect the first data rows of the two regional factor files.
```

The task has 23 criteria: eight programmatic checks and 15 semantic criteria. Programmatic checks assess deliverable validity and specific reported facts. Semantic criteria assess analysis, data handling, evidence-to-recommendation consistency, and deliverable quality. Selected criteria are shown below.

Selected Verification Criteria

```
DC-2 | Programmatic | file_valid
The analysis workbook is valid and contains at least one populated sheet.

FA-4 | Programmatic | value_in_context | fact_ref: vf12
The Emerging market 5-factor series start (198907) is correctly distinguished from
the Developed start of 199007.
Expected value: 198907; context keywords: Emerging, 1989, start.

AD-2 | Semantic
Compare regional premia over a common window, rather than each region's full history.

HC-1 | Semantic
Separate monthly and annual blocks so annual returns do not enter monthly statistics.

HC-2 | Semantic
Treat -99.99 and -999 as missing values rather than numerical returns.

HC-3 | Semantic
Discuss the CRSP/Bloomberg source difference when interpreting regional comparisons.

DQ-3 | Semantic
Produce a tidy long-format CSV with date, region, factor, and return fields.
```

E.1.4 Trajectory Excerpts

GLM 5.3 executes the task with OpenClaw in 1,317 seconds, producing 121 messages and 60 tool calls. Early exploration checks both ends of the source files and then compares regional schemas, provenance, and date coverage.

Rollout Excerpts: Inspection and Data Handling

```
[9] Assistant
Now let me inspect the actual file formats -- French CSVs are notoriously
messy (preamble text, multiple stacked tables, annual sections, footers,
-99.99 missing codes).

[15] Assistant
Key pitfalls already visible:
- US is CRSP (1926+), Developed/Emerging are Bloomberg (1989/1990+)
- Emerging has `-99.99` missing RMW/CMA early
- whitespace-padded values
- momentum is a separate file (`Mom` for US, `WML` for intl)
- stacked monthly+annual tables
Let me check the toolchain and the Emerging RMW/CMA validity start.
```

The solver checks factor-level availability next. Emerging RMW becomes available in July 1991 and CMA in July 1992, making July 1992 the binding start for the cross-region five-factor panel.

Rollout Excerpt: Common-Window Construction

```
[32] Assistant
Clean panel built. The binding constraint for a fair all-factor,
all-region window is Emerging CMA starting 1992-07, so the common
window is 1992-07 to 2025-08 (398 months). Now the analytics engine.

Tool call: write /project/scripts/analytics.py
Compute factor performance analytics from results/factor_returns_tidy.csv.
Outputs (results/tables/*.csv) consumed by the xlsx/pptx builders, plus
results/_headline.json with key numbers for the briefing.
```