SKILLEVO: AN EXPERIENCE LEARNING FRAMEWORK WITH REINFORCEMENT LEARNING FOR SKILL EVOLUTION

ABSTRACT

Large Language Models (LLMs) have evolved into agents capable of perception, reasoning, and acting in open environments. Yet, in long-horizon tasks with sparse rewards, existing methods are often inefficient. Group-based reinforcement learning (e.g., GRPO) provides critic-free and stable optimization, but its coarse credit signals cannot distinguish high-quality trajectories from those that merely succeed but contain redundant or invalid actions, leading to weak generalization. We propose **SkillEvo**(**Skill Evo**lution), a two-stage framework for efficient and sustainable agent learning. In the first stage, WebGRPO integrates a Reasoning and Execution Reward Model (RXERM) to deliver fine-grained feedback, and employs a dual-uncertainty filtering strategy to select informative tasks, improving sample efficiency and stability. In the second stage, SkillGenesis transforms trajectories into reusable skills, organized in a dynamically evolving Skill Path Graph (SPG). This enables skill composition, reuse, and the emergence of composite skills for long-term adaptability. On WebArena-Lite, SkillEvo raises the success rate of Llama-3.1-8B from 4.8% to 60.4% and GLM-4-9B from 6.1% to 57.6%, achieving new state-of-the-art results. These findings highlight that effective long-horizon learning requires not only refined credit signals but also systematic mechanisms for skill evolution.

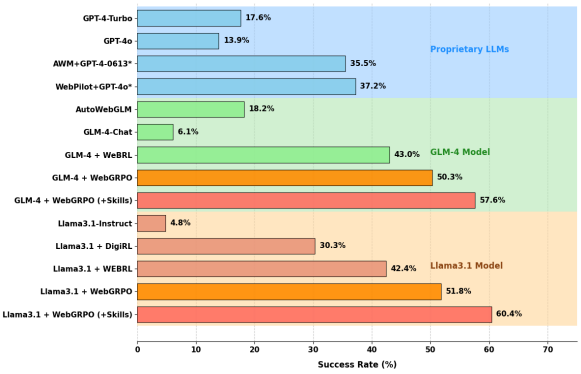

Figure 1: Comparative Success Rates of proprietary LLMs and open-sourced LLMs on WebArena-Lite.

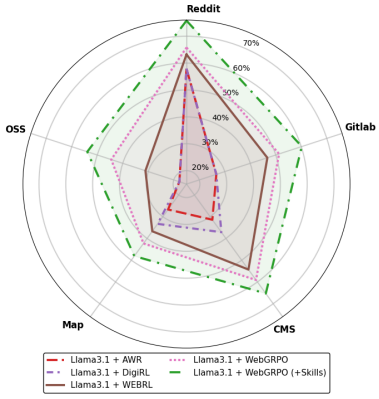

Figure 2: Comparison of Llama3.1 trained with WebGRPO and baseline methods.

1 INTRODUCTION

Large Language Models (LLMs) (Team et al., 2025; OpenAI et al., 2024; Qwen et al., 2025; DeepSeek-AI et al., 2025) have rapidly advanced in recent years, evolving from static question-answering systems into versatile agents capable of perception, reasoning, and acting in open environments. In complex scenarios such as embodied navigation (Shridhar et al., 2021; Li et al., 2025; Wang et al., 2023), web interaction (Furuta et al., 2024; Zheng et al., 2024; Gou et al., 2025), and

tool-augmented reasoning (Yao et al., 2023b; Schick et al., 2023), LLMs have demonstrated strong capabilities in task planning and logical reasoning. However, enabling LLM-based agents to learn efficiently in environments that involve long horizons, multi-turn interactions, sparse rewards, and dynamic structures remains a fundamental challenge.

Recently, group-based reinforcement learning (group-based RL) has gained significant attention in large-model training. Representative methods such as RLOO (Ahmadian et al., 2024) and GRPO (Shao et al., 2024b) compute relative advantages within a batch of rollouts, thereby avoiding value function estimation. These methods exhibit desirable properties such as being critic-free, memory-efficient, and stable in convergence, and have thus become important approaches for LLM post-training. They have achieved strong performance in tasks with immediate rewards, such as mathematical reasoning (Shao et al., 2024b) and code generation (Chen et al., 2021). However, when applied directly to long-horizon environments, group-based RL faces inherent limitations: the differences among trajectories are often flattened, making it difficult to distinguish high-quality trajectories from those that contain redundant or invalid actions—even if both are labeled as "successful." This coarse-grained credit signal significantly limits the effectiveness of group-based RL in long-horizon tasks.

In fact, a successful trajectory is not necessarily a high-quality trajectory. Under the same task and initial conditions, agents may complete the task through various trajectories, some of which involve repetitive or invalid actions (e.g., frequently switching between the same webpages or revisiting the same rooms in embodied settings). If training relies solely on terminal rewards, these trajectories are treated as equally successful, thereby obscuring the value of efficient reasoning and effective execution. Over time, this leads to slower convergence and weaker generalization.

To address these challenges, we propose the SkillEvo framework, whose central insight is: optimizing long-horizon tasks requires not only finer-grained reward signals but also answers to two higher-level questions—what experiences are worth learning, and how can these experiences be consolidated into long-term capabilities?

In the first stage, SkillEvo introduces the WebGRPO module, which combines a Reasoning and Execution Reward Model (RXERM) for trajectory optimization. Unlike traditional group-based RL approaches that rely solely on overall returns, RXERM explicitly models reasoning validity and execution efficiency, thereby distinguishing high-quality from low-quality trajectories and providing more informative signals for long-horizon optimization. In addition, we propose a dual-uncertainty active learning strategy, which jointly models execution uncertainty and reasoning uncertainty to dynamically select informative instances. This strategy not only avoids overtraining on "too easy" or "too hard" cases but also naturally forms a curriculum-like schedule: ensuring efficiency within each phase while gradually increasing task difficulty across phases, thereby mitigating inefficiency and instability in long-horizon optimization (Settles, 2009).

Nevertheless, trajectory-level optimization alone is insufficient for long-term agent evolution. Even if an agent learns high-quality experiences through refined reward signals and selective training, such experiences may not be consolidated or reused, leaving the agent to start from scratch when facing new tasks. To overcome this limitation, we introduce the SkillGenesis framework as the second stage of SkillEvo. SkillGenesis systematically transforms interaction experiences into reusable skills through a three-stage process of skill proposal, skill generation, and skill evolution. These skills are organized within a dynamically expanding Skill Path Graph (SPG), which supports dynamic composition and reuse of skills, and fosters the emergence of composite skills. This mechanism enables the agent to continually expand its skill repertoire and develop adaptive, transferable capabilities across tasks. The transition from merely "completing tasks" to genuinely "acquiring and evolving skills" represents the fundamental distinction of SkillEvo compared to prior approaches.

Building on this two-stage design, SkillEvo demonstrates both theoretical and empirical advantages. Experimental results on the WebArena-Lite benchmark (Liu et al., 2024) show that our method significantly boosts task success rates: Llama-3.1-8B improves from 4.8% to 60.4%, while GLM-4-9B improves from 6.1% to 57.6%. Across both open-source and proprietary LLMs, SkillEvo consistently outperforms existing methods, achieving new state-of-the-art (SOTA) performance. These results validate our core insight: by focusing on high-quality experiences through reward modeling and uncertainty-driven selection, and further consolidating these experiences into evolving skills, LLM agents can achieve more efficient, stable, and sustainable learning in complex environments.

2 RELATED WORK

Reinforcement Learning for LLM Agents. Reinforcement learning plays a central role in enhancing reasoning and decision-making for LLM agents. Classical methods such as PPO (Schulman et al., 2017) and AWR (Peng et al., 2019a) are widely used in RLHF, but their reliance on value networks incurs significant computational overhead and makes effective credit assignment difficult in long-horizon tasks. Recently, group-based reinforcement learning methods (e.g., GRPO (Shao et al., 2024a), Dr.GRPO (Liu et al., 2025), DAPO (Yu et al., 2025)) have gained attention for avoiding value modeling by computing relative advantages within groups of trajectories. These approaches achieve critic-free, efficient, and stable optimization, and have shown strong performance in short-horizon tasks. However, in long-horizon environments with sparse and delayed rewards, such methods often provide only coarse-grained credit signals, failing to distinguish high-quality trajectories from those containing redundant actions. To overcome this limitation, we propose the WebGRPO module, which integrates a Reasoning and Execution Reward Model (RXERM) to deliver fine-grained feedback, and further design a dual-uncertainty active filtering mechanism to dynamically select informative task instances. This approach improves sample efficiency and training stability, addressing the challenges of sparse rewards and insufficient credit assignment in long-horizon optimization.

LLMs as Agents. In recent years, large language models (LLMs) have increasingly been employed as agents in open environments. Prompt-based approaches (e.g., ReAct (Yao et al., 2023a) and Reflexion (Shinn et al., 2023)) leverage structured reasoning prompts and tool use (Schick et al., 2023; Xie et al., 2024) to extend model interaction capabilities, but they exhibit limited long-term learning and generalization. Training-based methods instead learn directly from interaction trajectories via behavior cloning or reinforcement learning, achieving stronger task grounding and execution robustness (Shen et al., 2024; Lai et al., 2024). Meanwhile, several works have explored skill abstraction and reuse (Zheng et al., 2025; Wu et al., 2025) to improve cross-task adaptability. However, these methods generally lack systematic mechanisms for skill evolution, making it difficult to continuously expand capabilities. To address this limitation, we introduce the SkillGenesis framework, which transforms interaction experiences into reusable skills through proposal, generation, and evolution stages, and organizes them via a Skill Path Graph (SPG). This design supports dynamic reuse and the emergence of composite skills, enabling long-term consolidation of capabilities.

3 METHODOLOGY

3.1 THE FINITE-HORIZON MARKOV DECISION PROCESS FOR WEB TASK

We model the process of completing a Web task as a finite-horizon Markov Decision Process (MDP), denoted as $\mathcal{M} = (T, S, A, P, K)$, where T represents the task goals; S represents the state space (e.g., observation sequences or interaction histories); A represents the action space, including all executable operations of the agent (e.g., `<Click>`, `<Type>`, `<Search>`, etc.); P represents the state transition dynamics and reward generation process; and K denotes the maximum interaction steps, after which the task terminates. Given a task goal T, the agent needs to complete the corresponding task. The agent policy π_θ generates an action a_t based on the task goal T, the current state s_t, and the interaction history $\tau_{<t} = \{s_0, a_0, r_0, \ldots, s_{t-1}, a_{t-1}, r_{t-1}\}$:

$$a_t \sim \pi_\theta(\cdot|T, s_t, \tau_{<t}), \quad (r_t, s_{t+1}) \sim P(\cdot|T, s_t, a_t) \tag{1}$$

The agent receives a positive reward only if the task is completed. In the finite-horizon setting, the trajectory ends upon task completion or reaching K steps. The final trajectory τ is as follows:

$$\tau = \{s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_K\} \tag{2}$$

3.2 TRAJECTORY-LEVEL OPTIMIZATION FOR WEB INTERACTION POLICIES

In web environments, tasks involve complex structures, long action sequences, and sparse, delayed rewards. To address this, WebGRPO optimizes the entire reasoning–interaction trajectory, enhancing long-term reasoning and policy execution. Unlike existing methods (Qi et al., 2025; Lai et al., 2024), WebGRPO directly maximizes cumulative rewards by optimizing full trajectories (including observations, reasoning processes, actions and feedback) , as illustrated in Figure 3.

$$J_{\text{WebGRPO}}(\theta) = \mathbb{E}_{M,\tau\sim\pi_\theta}\left[R(\tau)\right] \tag{3}$$

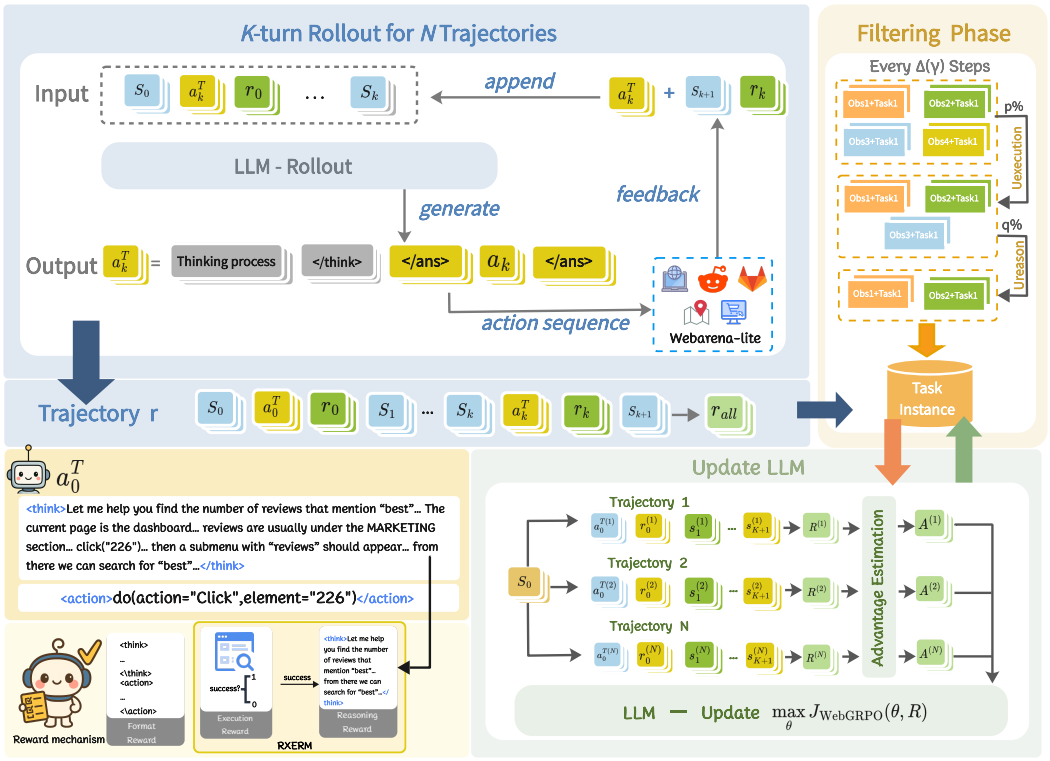

Figure 3: Overall WebGRPO pipeline, which leverages Two-Stage Reward mechanism RXERM and Dual-Uncertainty-Based Active Learning for Task Instance Filtering to achieve efficient training.

Where $\mathcal{M}$ is the finite-horizon Markov Decision Process (MDP), τ is a full sequence of reasoning-augmented interactions, and $R(\tau)$ is the total reward accumulated over trajectory τ.

3.2.1 Reasoning-Interaction Trajectories for Web Tasks

In each training iteration, the agent begins with an initial state–task goal pair (s_0, T), where a task instance corresponds to this specific pair, representing a concrete scenario for policy interaction and evaluation. Subsequently, the agent generates N complete interaction trajectories and, at each time step t, produces a reasoning-guided structured output:

$$a_t^\top = \texttt{<think>...</think> <action>}\, a_t \,\texttt{</action>} \tag{4}$$

The `<think>` section records reasoning (e.g., element localization, form filling), and `<action>` specifies the executable web action a_t(e.g., `<Click [3]>`, `<Scroll Down>`). The web environment executes a_t, updates the state s_{t+1}, and returns an immediate reward r_t, forming a web interaction trajectory: $\tau = \{s_0, a_0^\top, r_0, s_1, a_1^\top, r_1, \ldots, s_K\}$. WebGRPO interleaves rollout and update steps. It employs an on-policy optimization approach, where trajectories are collected under the current policy π_θ for updates. In each training iteration, the agent samples from P initial state–task goal pair (s_0, T), collecting N complete trajectories per state, resulting in $P \cdot N$ trajectories per iteration. Given a batch size of E, each iteration performs $\frac{P \cdot N}{E}$ gradient update steps. After L training iterations, the total number of gradient updates is $S = L \cdot \left(\frac{P \cdot N}{E}\right)$.

3.2.2 Trajectory-Level GRPO Optimization with RXERM

We propose a trajectory-level WebGRPO optimization framework based on Group Relative Policy Optimization (GRPO). Through format reward and a two-stage LLMs-based reward mechanism—RXERM (Reasoning & Execution Reward Model), rich and effective reward signals are provided for policy learning. A dual uncertainty-based active learning strategy further selects high-information task instances, improving the stability and performance of policy optimization.

Token-Level Trajectory Definition. In the WebGRPO framework, the policy probability $\pi_\theta(\tau)$ can be decomposed into token-level likelihoods due to the autoregressive nature of LLM-based agents. For every initial state–task goal pair (s_0, T), multiple rollouts are sampled to collect complete interaction trajectories. The trajectory can be defined as:$\tau_i = \{\tau_{i,(1)}, \ldots, \tau_{i,(L)}\}$, where $\tau_{i,(t)}$ denotes the t-th token in trajectory τ_i, and L is the total number of tokens in the trajectory.

Two-Stage Reward mechanism RXERM and Format Reward. Under the same task and initial environment conditions, even if the agent successfully completes the task, many trajectories may still contain redundant or invalid actions. If only execution rewards are considered, it becomes difficult to effectively distinguish the quality among different successful trajectories. To address this limitation, we propose the two-stage LLMs-based reward mechanism—RXERM (Reasoning & Execution Reward Model) , which integrates both trajectory-level credit assignment and action-level credit assignment. RXERM separately evaluates both intermediate reasoning trace $\mathcal{R}$ (each `<think>` reasoning content) and task completion $\mathcal{E}$, providing fine-grained and comprehensive reward signals. Specifically, RXERM consists of two stages: In the first stage, Execution Reward Model evaluates task completion $\mathcal{E}$ based on the task goal T, trajectory τ, and final state s_K, producing a binary execution reward $R_{\text{execution}}$ (1 for success, 0 otherwise). Only successful trajectories proceed to the second stage, where Reasoning Reward Model evaluates $\mathcal{R}$ using predefined prompts to compute intermediate reasoning reward R_{reason}, defined as the average reward over all reasoning steps along the trajectory. To separately assess action correctness and reasoning quality while ensuring output consistency, we introduce a widely used format reward. This encourages structured outputs using `<think>` for reasoning and `<action>` for actions. The final trajectory reward is computed as:

$$R(\tau) = \alpha R_{\text{reason}}(\tau) + (1-\alpha) R_{\text{execution}}(\tau) + \beta R_{\text{format}}(\tau) \tag{5}$$

where α controls the balance between reasoning and execution rewards, and β is a fixed weight for the format reward, used to provide basic structural constraints. This design ensures that the optimization process emphasizes high-quality reasoning–interaction trajectories rather than merely achieving task completion, thereby effectively preventing the agent from attaining non-robust success through repeated trials or accidental exploration.

Dual-Uncertainty-Based Active Learning for Task Instance Filtering. To improve policy optimization stability and efficiency, we propose a dual uncertainty-based active learning strategy for task instance filtering. Based on active learning theory (Settles, 2009), this strategy selects the most informative task instances by prioritizing those with high uncertainty, avoiding low-value trivial tasks and overly difficult tasks with weak reward signals. We jointly consider uncertainty in task completion $\mathcal{E}$ and intermediate reasoning trace $\mathcal{R}$ to guide task instance selection. Specifically, two uncertainty metrics are defined:

$$U_{\text{execution}} = \text{Std}_{\tau \sim \pi_\theta(\cdot|s_0,T)} \left[R_{\text{execution}}(\tau) \right], \quad U_{\text{reason}} = \text{Std}_{\tau \sim \pi_\theta(\cdot|s_0,T)} \left[R_{\text{reason}}(\tau) \right]. \tag{6}$$

Both uncertainties are computed as the standard deviation of rewards over multiple rollouts for the same task instance. In WebGRPO training, we first rank all task instances by $U_{\text{execution}}$ and retain the top $p\%$ with the highest uncertainty, where the corresponding threshold is denoted as δ_p, i.e., $\mathcal{I}_{\text{execution}} = \{i \mid U_{\text{execution}}(i) \geq \delta_p\}$. From $\mathcal{I}_{\text{execution}}$, we further retain the top $q\%$ based on reasoning uncertainty U_{reason}, with threshold δ_q, i.e., $\mathcal{I}_{\text{filtered}} = \{i \in \mathcal{I}_{\text{execution}} \mid U_{\text{reason}}(i) \geq \delta_q\}$. Only instances in $\mathcal{I}_{\text{filtered}}$ are used for policy optimization. Through this dual uncertainty filtering strategy, the model effectively eliminates low-information rollouts and significantly mitigates the risk of policy collapse in multi-turn reasoning scenarios. This filtering is not a one-time preprocessing step but is periodically repeated during training. We introduce a curriculum variable γ to adaptively adjust the update interval Δ. At each phase, after every $\Delta(\gamma)$ steps, uncertainty is recalculated and the task instances updated. With linear scheduling $\Delta(\gamma) = \Delta_0 + \gamma$, shorter intervals in early stages ensure frequent updates for fast learning, while longer intervals in later stages provide stable training. This balances efficiency within phases and curriculum-driven progression across phases.

Trajectory Advantage Normalization. Generate multiple trajectories for the retained high-quality instances, assign scalar rewards $R(\tau_i)$, and perform normalization within the batch. The normalized reward $\hat{A}_{i,t}$ is shared across all tokens within the same trajectory.

$$\hat{A}_{i,t} = \frac{R(\tau_i) - \text{mean}\left[R(\tau_1), \ldots, R(\tau_G)\right]}{\text{std}\left[R(\tau_1), \ldots, R(\tau_G)\right]}, \tag{7}$$

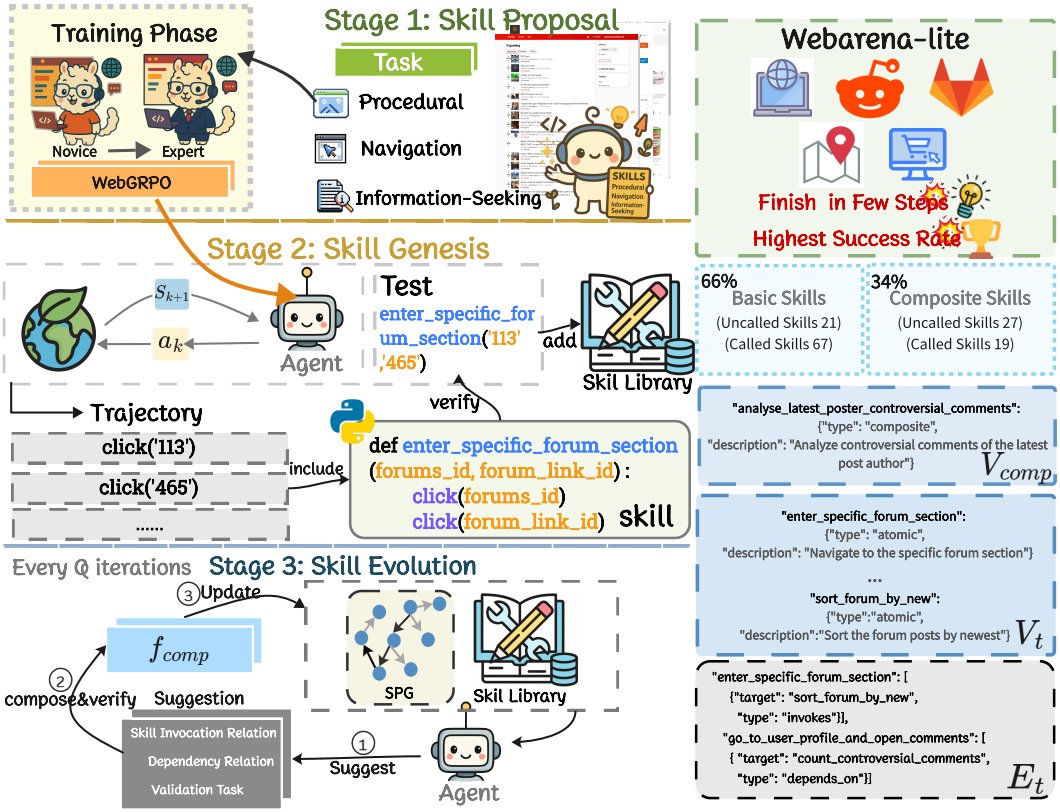

Figure 4: SkillGenesis, A Skill Evolution and Path Optimization Framework. WebGRPO can be seamlessly integrated into the SkillGenesis framework.

Policy Optimization Objective. The GRPO objective becomes:

$$J_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|\tau_i|}\sum_{t=1}^{|\tau_i|}\min\left[\rho_{i,t}\cdot\hat{A}_{i,t},\ \text{clip}(\rho_{i,t},1-\epsilon,1+\epsilon)\cdot\hat{A}_{i,t}\right], \tag{8}$$

where $\rho_{i,t} = \frac{\pi_\theta(\tau_{i,(t)}|\tau_{i,<t})}{\pi_{\text{old}}(\tau_{i,(t)}|\tau_{i,<t})}$, G is the number of trajectories in the batch, $\tau_{i,(t)}$ denotes the t-th token in trajectory τ_i, and $\tau_{i,<t}$ is its prefix.

3.3 SkillGenesis: Dynamic Skill Evolution with Skill Path Graphs

To enhance the agent's ability to complete tasks and improve skill reusability in complex web environments, we seamlessly integrate the WebGRPO into the SkillGenesis framework. This framework consists of three stages: Skill Proposal, Skill Genesis, and Skill Evolution, enabling dynamic expansion of the skill library through a periodic skill evolution mechanism and optimization strategies based on the Skill Path Graph (SPG), as illustrated in Figure 4(The figure is based on the case study in Appendix F) .

Skill Library and SPG. At time step t, the skill library $\mathcal{A}_t$ stores executable functions (Python programs) for low-level web actions (e.g., `enter_specific_forum_section()`), while the SPG $\mathcal{G}_t = (V_t, E_t)$ organizes skills as a directed graph, where V_t represents nodes as structural abstractions (e.g., {"enter_specific_forum_section": type=atomic, description="Navigate to the specific forum section"}) and E_t represents edges. Edges indicate either an invocation sequence or a dependency relation, such as "SearchItem $\mapsto$ Login" (dependency) and "SearchItem $\rightarrow$ AddToCart" (sequence). Dependency edges represent mandatory prerequisites, while sequence edges denote recommended but non-mandatory orders.

Skill Proposal Stage. In the Skill Proposal stage, LLM generates practical and reusable task goals $\{T_i\}_{i=1}^{P}$, based on P initial webpage states $\{s_0^{(i)}\}_{i=1}^{P}$. These task goals cover common web interaction types, including procedural tasks, navigational tasks, and information-seeking tasks.

Skill Genesis Stage. In the Skill Genesis stage, the agent uses the WebGRPO policy to generate reasoning-interaction trajectories $\tau = \{s_0, a_0^\top, r_0, s_1, a_1^\top, r_1, \ldots, s_K\}$. When a task T is successfully completed, we adopt a trajectory prefix rewriting method to induce skills. By truncating the original trajectory τ, we construct a skill-invocation-based prefix $\tau_D = \{s_0, a_0, \ldots, s_D\}$. The agent continues to generate subsequent actions based on τ_D, forming a complete and verifiable trajectory $\tau_f = \tau_D \cup \{a_{D+1}, \ldots, a_K\}$. The validity of the trajectory is evaluated by LLM based on three criteria: 1) *Correctness*: The task is successfully completed; 2) *Skill Usage*: New skills are utilized in the trajectory; 3) *Skill Effectiveness*: Skill invocations effectively change the environment state. If the criteria are met, the new cleaned skill set $\mathcal{D}_{\text{called}}$ is added to the skill library, updating:$\mathcal{A}_{t+1} = \mathcal{A}_t \cup \mathcal{D}_{\text{called}}$.

Skill Evolution Stage. The framework periodically triggers Skill Evolution stage after every fixed Q SkillGenesis iterations. LLM proposes skill composition suggestions based on the current skill library $\mathcal{A}_t$ and the SPG $\mathcal{G}_t = (V_t, E_t)$. These suggestions include skill invocation relations, dependency relations, and corresponding validation tasks T_{comp}. Based on these suggestions, LLM utilizes the current SPG to composes candidate composite skills f_{comp} and verifies them by executing T_{comp}. If the composite skill successfully completes the validation task, it is added to the skill library, updating

$$\mathcal{A}_{t+1} = \mathcal{A}_t \cup \{f_{\text{comp}}\}. \tag{9}$$

Simultaneously, the SPG is updated to reflect the new composite skill and its dependencies:

$$V_{t+1} = V_t \cup \{v_{\text{comp}}\}, \quad E_{t+1} = E_t \cup \{(v_i, v_j) \mid (v_i \to v_j) \text{ or } (v_i \mapsto v_j) \text{ discovered in } v_{\text{comp}}\} \tag{10}$$

where v_{comp} denotes the new composite skill node. This periodic evolution mechanism ensures the skill library continuously accumulates high-quality composite skills, enhancing task efficiency and generalization capability. By continuously expanding and optimizing the skill library during exploration, the generated skills can be directly reused in subsequent tasks, effectively improving task performance in complex web environments.

4 EXPERIMENTS

4.1 ENVIRONMENTS AND BASELINES

4.1.1 ENVIRONMENT

To evaluate the performance of our proposed approach and baselines, we conducted experiments in the WebArena environment (Zhou et al., 2023). WebArena is an interactive benchmark designed for complex web navigation tasks, providing a self-hosted sandboxed web environment covering five major application domains: OpenStreetMap (Map), Reddit, GitLab, online store content management system (CMS), and OneStopShop (OSS). The full WebArena benchmark contains 812 instructions, but annotation errors and vague evaluation standards in the original dataset may undermine fairness and credibility. To ensure statistical reliability and lower the computational overhead of large-scale evaluations, we adopt WebArena-Lite (Lai et al., 2024), a human-verified subset of WebArena. This version selects 165 representative tasks as the evaluation set, providing a more consistent and trustworthy basis for assessment.

4.1.2 BASELINES

To evaluate the effectiveness of our method, we compare it against several baseline approaches, which can be divided into two categories: proprietary large language models utilizing prompting techniques, and open-source large language models trained with alternative paradigms. For fine-tuning methods, we adopt Llama3.1 (Grattafiori et al., 2024) and GLM-4-9B (GLM et al., 2024b) as backbone models. Following (Qi et al., 2025), the imitation learning (also known as supervised fine-tuning, SFT) baselines are trained on 9,460 trajectories from the human-labeled demonstration dataset of WebArena-Lite. For reinforcement learning baselines, we use the SFT-trained model as the initial actor, while the critic is also based on the SFT-trained model with an additional randomly

initialized value head. For the WebGRPO method, we employ a skill proposer during training to generate a large number of task instances, construct the initial task pool, and apply task instance filtering. After convergence, the policy π_θ is fixed, and the agent explores within the WebArena environment to continuously expand and validate the skill library, thereby adapting to more complex and dynamic scenarios. Specifically, during the SkillGenesis stage, the agent executes 160 exploration iterations, where newly discovered skills are incrementally added to the skill library. Every 20 iterations ($Q = 20$) of SkillGenesis, the Skill Evolution stage is triggered to evolve the skills using SPG and the existing skill library. The RXERM model used in reinforcement learning training and the LLM applied in the SkillGenesis stage both adopt GPT-4o. The rationale can be found in Appendix E.1and Appendix E.6. More details of the training process can be found in Appendix B.

Table 1: Performance comparison of WebGRPO against baseline methods in terms of Task Success Rate (SR), evaluated on WebArena-Lite (Zhou et al., 2023; Liu et al., 2024). WebArena-Lite is a human-verified subset of the WebArena benchmark. (Results marked with * are reported from the literature on the full WebArena dataset.) The **best** and second-best models are indicated.

Models	#Params	Reddit	Gitlab	CMS	Map	OSS	Avg. SR
Proprietary LLMs 🔓							
GPT-4-Turbo (Achiam et al., 2023)	N/A	10.5	16.7	14.3	36.7	13.3	17.6
GPT-4o	N/A	10.5	10.0	20.0	20.0	11.1	13.9
AWM + GPT-4-0613* (Wang et al., 2024)	N/A	50.9	31.8	29.1	43.3	30.8	35.5
WebPilot + GPT-4o* (Zhang et al., 2025)	N/A	65.1	39.4	24.7	33.9	36.9	37.2
SkillWeaver+GPT-4o* (Zheng et al., 2025)	N/A	50.0	22.2	25.8	33.9	27.2	29.8
ASI+Claude-3.5-sonnet* (Wang et al., 2025)	N/A	54.7	32.2	44.0	43.1	40.1	40.4
Open-sourced LLMs 🔓							
AutoWebGLM (Lai et al., 2024)	6B	9.4	15.0	28.6	24.8	17.1	18.2
GLM-4-Chat (GLM et al., 2024a)	9B	5.3	10.0	6.7	3.3	6.7	6.1
GLM-4 + SFT (BC)	9B	47.4	13.3	31.4	23.3	13.3	22.4
GLM-4 + Filtered BC	9B	52.6	10.0	31.4	26.7	20.0	24.8
GLM-4 + AWR (Peng et al., 2019b)	9B	52.6	16.7	34.3	30.0	22.2	27.9
GLM-4 + DigiRL (Bai et al., 2024a)	9B	63.2	30.0	34.3	26.7	26.7	31.5
GLM-4 + WEBRL (Qi et al., 2025)	9B	57.9	50.0	48.6	36.7	37.8	43.0
GLM-4 + WebGRPO	9B	62.3	52.1	52.9	45.3	46.9	50.3
GLM-4 + WebGRPO (+Skills)	9B	<u>71.2</u>	**61.2**	<u>59.2</u>	**50.6**	<u>51.5</u>	<u>57.6</u>
Δ		↑ 14%	↑ 18%	↑ 12%	↑ 12%	↑ 10%	↑ 15%
Llama3.1-Instruct (Grattafiori et al., 2024)	8B	0.0	3.3	2.9	3.3	11.1	4.8
Llama3.1 + SFT (BC)	8B	36.8	6.7	20.0	33.3	17.8	20.6
Llama3.1 + Filtered BC	8B	52.6	20.0	31.4	23.3	8.9	23.0
Llama3.1 + AWR (Peng et al., 2019a)	8B	57.9	26.7	31.4	26.7	17.8	28.5
Llama3.1 + DigiRL (Bai et al., 2024b)	8B	57.9	26.7	37.1	33.3	17.8	30.3
Llama3.1 + WEBRL (Qi et al., 2025)	8B	63.2	46.7	54.3	36.7	31.1	42.4
Llama3.1 + WebGRPO	8B	65.8	51.3	59.1	42.3	44.5	51.8
Llama3.1 + WebGRPO (+Skills)	8B	**75.8**	<u>60.2</u>	**65.3**	<u>48.1</u>	**53.9**	**60.4**
Δ		↑15.2%	↑17.3%	↑10.5%	↑13.7%	↑21.1%	↑16.6%

5 Main Results

Table 1 presents a comparison of WebGRPO with existing baseline methods on the WebArena-Lite benchmark. Our proposed method consistently outperforms all baselines across various environments, demonstrating the effectiveness of trajectory-level optimization and skill reuse. Specifically, for GLM-4, GLM-4 + WebGRPO (+Skills) achieves the highest average task success rate (SR) of 57.6%, surpassing the strongest baseline, GLM-4 + WEBRL (43.0%), by 14.6%, and outperforming all proprietary models. Notably, in the *Gitlab* and *Reddit* environments, compared to WebGRPO without skill reuse, the success rates further improve by +18.0% and +14.0%, respectively, demonstrating strong generalization and task adaptability. Similarly, for the Llama3.1 series, WebGRPO (+Skills) achieves an impressive average SR of 60.4%, outperforming the top baseline, Llama3.1 + WEBRL (42.4%), by 18.0%. In particular, the success rates in the *Gitlab* and *OSS* environments improve by +17.3% and +21.1%, respectively, compared to WebGRPO without skill reuse, further highlighting the significant advantages of skill reuse. Overall, these results demonstrate that WebGRPO, through the integration of reasoning-augmented interaction trajectories and efficient skill

Table 2: Quantitative ablation of the Skill Evolution Stage on GLM-4-9B and Llama-3.1-8B.

Model	Config	Reddit	GitLab	CMS	Map	OSS	Avg
GLM-4-9B	No skills	62.3	52.1	52.9	45.3	46.9	50.3
	Base-only	66.8	55.8	54.1	47.5	48.7	54.6
	Base+Compose	71.2	61.2	59.2	50.6	51.5	57.6
Llama-3.1-8B	No skills	65.8	51.3	59.1	42.3	44.5	51.8
	Base-only	69.7	55.2	61.3	44.1	49.4	55.9
	Base+Compose	75.8	60.2	65.3	48.1	53.9	60.4

library construction, significantly enhances the agent's reasoning capability and task completion performance in complex web environments.

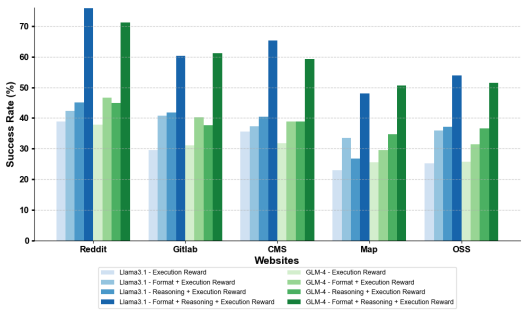

Figure 5: Quantitative Ablation Study on RXERM and Format Reward.

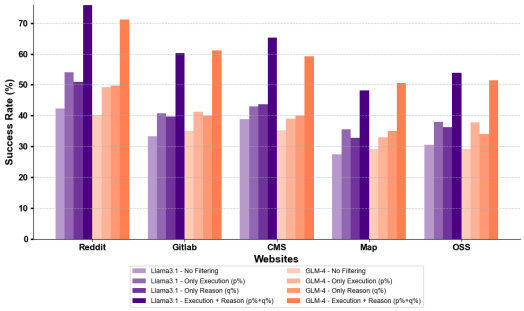

Figure 6: Quantitative Ablation Study on Dual-Uncertainty-Based Active Learning for Task Instance Filtering.

6 Ablation Studies

Reward Mechanism. As shown in Figure 5, we performed ablation studies on Llama3.1 and GLM-4 to assess the two-stage RXERM reward mechanism and format reward. Removing the reasoning reward reduced average success rates by 7.62% and 7.42%, with the largest drop in *Reddit*, showing its importance for complex reasoning. Removing the format reward caused drops of 7.49% and 6.78%, confirming its role in stability. With only execution reward, rates fell to 30.42% and 30.36%, indicating that a single reward is insufficient.

Dual-Uncertainty-Based Active Learning for Task Instance Filtering. With Execution + Reasoning Reward Filtering, the success rates of Llama3.1 and GLM-4 increase to 60.4% and 57.6%, respectively, representing a substantial improvement over No Filtering. This demonstrates that the dual uncertainty strategy effectively boosts task success rate and stability (see Figure 6).

Effect of Skill Evolution Stage. To evaluate the effect of the skill evolution stage, we compared models under three configurations: *No skills*, *Base-only skills*, and *Base+Compose skills*. Results for GLM-4-9B and Llama-3.1-8B are shown in Table 2. Experimental results show that GLM-4-9B improved from 50.3% (no skills) to 54.6% (basic) and 57.6% (composite). Llama-3.1-8B similarly rose from 51.8% to 55.9% and 60.4%. Thus, the **Skill Evolution stage**, especially composite skills, is essential for improving performance on complex tasks. The detailed statistics of skill coverage, call frequency, and task-level skill call counts are provided in the Appendix D.

More detailed qualitative experiments can be found in Appendix E, including the evaluation of the Reward Mechanism RXERM in Section E.1, the analysis of model performance variance in Section E.2, the sensitivity analysis of the Reward Balancing Factor α in Section E.3, the sensitivity analysis of the Formatting Reward Weight β in Section E.4, the sensitivity analysis of Dual-Uncertainty Task Filtering in Section E.5, and the performance comparison of SkillEvo with different LLMs in Section E.6. These sections provide detailed analysis and experimental results across various aspects of our method.

7 Conclusion

We propose **SkillEvo**, a framework combining trajectory-level optimization (WebGRPO) and dynamic skill evolution (SkillGenesis), which achieves state-of-the-art performance in long-horizon web tasks. Despite its effectiveness, our failure cases show limitations in dynamic environments

due to strong reward dependence, fragile skill composition, and high computational cost. Future work will focus on reducing training overhead, improving robustness of skill representations, and enabling continual learning with adaptive curricula for better scalability in real-world scenarios.

ETHICS STATEMENT

This research strictly adheres to the ICLR Code of Ethics. No human subjects or animal experiments were involved, and therefore no ethical risks are present. All experiments were conducted on publicly available datasets (e.g., WebArena-Lite) in accordance with their usage agreements, and no personally identifiable information was used. In the design of our method and experiments, we carefully considered the avoidance of potential biases and discriminatory outcomes to ensure fairness and generalizability of the results across different models and environments. Furthermore, this study does not involve any activities that may cause privacy breaches, security risks, or inappropriate social impacts. We are committed to maintaining academic integrity and transparency throughout the entire research and publication process.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of this work, we provide detailed descriptions of the experimental environment, dataset configurations, training procedures, and hyperparameter settings in the appendices (see Subsection4.1.1, Subsection4.1.2, Appendix B). All theoretical derivations, model details, and pseudocode are explicitly presented in the main text and appendices. In addition, upon acceptance, we will release the full source code and execution scripts on GitHub, enabling the research community to verify and reproduce our results. The released resources will include preprocessing scripts, implementations of WebGRPO and SkillGenesis, key hyperparameter configurations, and evaluation metric computation methods. We believe these resources provide sufficient detail for researchers to fully reproduce and extend our experimental results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Hao Bai, Yifei Zhou, Mert Cemri, Jiayi Pan, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning, 2024a. URL https://arxiv.org/abs/2406.11896.

Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. *Advances in Neural Information Processing Systems*, 37:12461–12495, 2024b.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Representation Learning*, volume 2024, pp. 29528–29558, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/7ef7d8359036afd8c2378d82c21058a4-Paper-Conference.pdf.

Team GLM, :, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Jingyu Sun, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. Chatglm: A family of large language models from glm-130b to glm-4 all tools, 2024a. URL https://arxiv.org/abs/2406.12793.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024b.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents, 2025. URL https://arxiv.org/abs/2410.05243.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, et al. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5295–5306, 2024.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li Erran Li, Ruohan Zhang, Weiyu Liu, Percy Liang, Li Fei-Fei, Jiayuan Mao, and Jiajun Wu. Embodied agent interface: Benchmarking llms for embodied decision making, 2025. URL https://arxiv.org/abs/2410.07166.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, et al. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan

Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019a.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019b. URL https://arxiv.org/abs/1910.00177.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025. URL https://arxiv.org/abs/2411.02337.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. URL https://arxiv.org/abs/2302.04761.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison, Madison, WI, 2009. URL http://burrsettles.com/pub/settles.activelearning.pdf.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024a. URL https://arxiv.org/abs/2402.03300.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024b. URL https://arxiv.org/abs/2402.03300.

Junhong Shen, Atishay Jain, Zedian Xiao, Ishan Amlekar, Mouad Hadji, Aaron Podolny, and Ameet Talwalkar. Scribeagent: Towards specialized web agents using production-scale workflow data, 2024. URL https://arxiv.org/abs/2411.15004.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning, 2021. URL https://arxiv.org/abs/2010.03768.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson,

Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Güra, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rrustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayana Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, Alex Kaskasoli, Sébastien M. R. Arnold, Vijay Vasudevan, Shubham Agrawal, Jason Riesa, Dmitry Lepikhin, Richard Tanburn, Srivatsan Srinivasan, Hyeontaek Lim, Sarah Hodkinson, Pranav Shyam, Johan Ferret, Steven Hand, Ankush Garg, Tom Le Paine, Jian Li, Yujia Li, Minh Giang, Alexander Neitz, Zaheer Abbas, Sarah York, Machel Reid, Elizabeth Cole, Aakanksha Chowdhery, Dipanjan Das, Dominika Rogozińska, Vitaliy Nikolaev, Pablo Sprechmann, Zachary Nado, Lukas Zilka, Flavien Prost, Luheng He, Marianne Monteiro, Gaurav Mishra, Chris Welty, Josh Newlan, Dawei Jia, Miltiadis Allamanis, Clara Huiyi Hu, Raoul de Liedekerke, Justin Gilmer, Carl Saroufim, Shruti Rijhwani, Shaobo Hou, Disha Shrivastava, Anirudh Baddepudi, Alex Goldin, Adnan Ozturel, Albin Cassirer, Yunhan Xu, Daniel Sohn, Devendra Sachan, Reinald Kim Amplayo, Craig Swanson, Dessie Petrova, Shashi Narayan, Arthur Guez, Siddhartha Brahma, Jessica Landon, Miteyan Patel, Ruizhe Zhao, Kevin Villela, Luyu Wang, Wenhao Jia, Matthew Rahtz, Mai Giménez, Legg Yeung, James Keeling, Petko Georgiev, Diana Mincu, Boxi Wu, Salem Haykal, Rachel Saputro, Kiran Vodrahalli, James Qin, Zeynep Cankara, Abhanshu Sharma, Nick Fernando, Will Hawkins, Behnam Neyshabur, Solomon Kim, Adrian Hutter, Priyanka Agrawal, Alex Castro-Ros, George van den Driessche, Tao Wang, Fan Yang, Shuo yiin Chang, Paul Komarek, Ross McIlroy, Mario Lučić, Guodong Zhang, Wael Farhan, Michael Sharman, Paul Natsev, Paul Michel, Yamini Bansal, Siyuan Qiao, Kris Cao, Siamak Shakeri, Christina Butterfield, Justin Chung, Paul Kishan Rubenstein, Shivani Agrawal, Arthur Mensch, Kedar Soparkar, Karel Lenc, Timothy Chung, Aedan Pope, Loren Maggiore, Jackie Kay, Priya Jhakra, Shibo Wang, Joshua Maynez, Mary Phuong, Taylor Tobin, Andrea Tacchetti, Maja Trebacz, Kevin Robinson, Yash Katariya, Sebastian Riedel, Paige Bailey, Kefan Xiao, Nimesh Ghelani, Lora Aroyo, Ambrose Slone, Neil Houlsby, Xuehan Xiong, Zhen Yang, Elena Gribovskaya, Jonas Adler, Mateo Wirth, Lisa Lee, Music Li, Thais Kagohara, Jay Pavagadhi, Sophie Bridgers, Anna Bortsova, Sanjay Ghemawat, Zafarali Ahmed, Tianqi Liu, Richard Powell, Vijay Bolina, Mariko Iinuma, Polina Zablotskaia, James Besley, Da-Woon Chung, Timothy Dozat, Ramona Comanescu, Xiance Si, Jeremy Greer, Guolong Su, Martin Polacek, Raphaël Lopez Kaufman, Simon Tokumine, Hexiang Hu, Elena Buchatskaya, Yingjie Miao, Mohamed Elhawaty, Aditya Siddhant, Nenad Tomasev, Jinwei Xing, Christina Greer, Helen Miller, Shereen Ashraf, Aurko Roy, Zizhao Zhang, Ada

Ma, Angelos Filos, Milos Besta, Rory Blevins, Ted Klimenko, Chih-Kuan Yeh, Soravit Changpinyo, Jiaqi Mu, Oscar Chang, Mantas Pajarskas, Carrie Muir, Vered Cohen, Charline Le Lan, Krishna Haridasan, Amit Marathe, Steven Hansen, Sholto Douglas, Rajkumar Samuel, Mingqiu Wang, Sophia Austin, Chang Lan, Jiepu Jiang, Justin Chiu, Jaime Alonso Lorenzo, Lars Lowe Sjösund, Sébastien Cevey, Zach Gleicher, Thi Avrahami, Anudhyan Boral, Hansa Srinivasan, Vittorio Selo, Rhys May, Konstantinos Aisopos, Léonard Hussenot, Livio Baldini Soares, Kate Baumli, Michael B. Chang, Adrià Recasens, Ben Caine, Alexander Pritzel, Filip Pavetic, Fabio Pardo, Anita Gergely, Justin Frye, Vinay Ramasesh, Dan Horgan, Kartikeya Badola, Nora Kassner, Subhrajit Roy, Ethan Dyer, Víctor Campos Campos, Alex Tomala, Yunhao Tang, Dalia El Badawy, Elspeth White, Basil Mustafa, Oran Lang, Abhishek Jindal, Sharad Vikram, Zhitao Gong, Sergi Caelles, Ross Hemsley, Gregory Thornton, Fangxiaoyu Feng, Wojciech Stokowiec, Ce Zheng, Phoebe Thacker, Çağlar Ünlü, Zhishuai Zhang, Mohammad Saleh, James Svensson, Max Bileschi, Piyush Patil, Ankesh Anand, Roman Ring, Katerina Tsihlas, Arpi Vezer, Marco Selvi, Toby Shevlane, Mikel Rodriguez, Tom Kwiatkowski, Samira Daruki, Keran Rong, Allan Dafoe, Nicholas FitzGerald, Keren Gu-Lemberg, Mina Khan, Lisa Anne Hendricks, Marie Pellat, Vladimir Feinberg, James Cobon-Kerr, Tara Sainath, Maribeth Rauh, Sayed Hadi Hashemi, Richard Ives, Yana Hasson, Eric Noland, Yuan Cao, Nathan Byrd, Le Hou, Qingze Wang, Thibault Sottiaux, Michela Paganini, Jean-Baptiste Lespiau, Alexandre Moufarek, Samer Hassan, Kaushik Shivakumar, Joost van Amersfoort, Amol Mandhane, Pratik Joshi, Anirudh Goyal, Matthew Tung, Andrew Brock, Hannah Sheahan, Vedant Misra, Cheng Li, Nemanja Rakićević, Mostafa Dehghani, Fangyu Liu, Sid Mittal, Junhyuk Oh, Seb Noury, Eren Sezener, Fantine Huot, Matthew Lamm, Nicola De Cao, Charlie Chen, Sidharth Mudgal, Romina Stella, Kevin Brooks, Gautam Vasudevan, Chenxi Liu, Mainak Chain, Nivedita Melinkeri, Aaron Cohen, Venus Wang, Kristie Seymore, Sergey Zubkov, Rahul Goel, Summer Yue, Sai Krishnakumaran, Brian Albert, Nate Hurley, Motoki Sano, Anhad Mohananey, Jonah Joughin, Egor Filonov, Tomasz Kępa, Yomna Eldawy, Jiawern Lim, Rahul Rishi, Shirin Badiezadegan, Taylor Bos, Jerry Chang, Sanil Jain, Sri Gayatri Sundara Padmanabhan, Subha Puttagunta, Kalpesh Krishna, Leslie Baker, Norbert Kalb, Vamsi Bedapudi, Adam Kurzrok, Shuntong Lei, Anthony Yu, Oren Litvin, Xiang Zhou, Zhichun Wu, Sam Sobell, Andrea Siciliano, Alan Papir, Robby Neale, Jonas Bragagnolo, Tej Toor, Tina Chen, Valentin Anklin, Feiran Wang, Richie Feng, Milad Gholami, Kevin Ling, Lijuan Liu, Jules Walter, Hamid Moghaddam, Arun Kishore, Jakub Adamek, Tyler Mercado, Jonathan Mallinson, Siddhinita Wandekar, Stephen Cagle, Eran Ofek, Guillermo Garrido, Clemens Lombriser, Maksim Mukha, Botu Sun, Hafeezul Rahman Mohammad, Josip Matak, Yadi Qian, Vikas Peswani, Pawel Janus, Quan Yuan, Leif Schelin, Oana David, Ankur Garg, Yifan He, Oleksii Duzhyi, Anton Älgmyr, Timothée Lottaz, Qi Li, Vikas Yadav, Luyao Xu, Alex Chinien, Rakesh Shivanna, Aleksandr Chuklin, Josie Li, Carrie Spadine, Travis Wolfe, Kareem Mohamed, Subhabrata Das, Zihang Dai, Kyle He, Daniel von Dincklage, Shyam Upadhyay, Akanksha Maurya, Luyan Chi, Sebastian Krause, Khalid Salama, Pam G Rabinovitch, Pavan Kumar Reddy M, Aarush Selvan, Mikhail Dektiarev, Golnaz Ghiasi, Erdem Guven, Himanshu Gupta, Boyi Liu, Deepak Sharma, Idan Heimlich Shtacher, Shachi Paul, Oscar Akerlund, François-Xavier Aubet, Terry Huang, Chen Zhu, Eric Zhu, Elico Teixeira, Matthew Fritze, Francesco Bertolini, Liana-Eleonora Marinescu, Martin Bölle, Dominik Paulus, Khyatti Gupta, Tejasi Latkar, Max Chang, Jason Sanders, Roopa Wilson, Xuewei Wu, Yi-Xuan Tan, Lam Nguyen Thiet, Tulsee Doshi, Sid Lall, Swaroop Mishra, Wanming Chen, Thang Luong, Seth Benjamin, Jasmine Lee, Ewa Andrejczuk, Dominik Rabiej, Vipul Ranjan, Krzysztof Styrc, Pengcheng Yin, Jon Simon, Malcolm Rose Harriott, Mudit Bansal, Alexei Robsky, Geoff Bacon, David Greene, Daniil Mirylenka, Chen Zhou, Obaid Sarvana, Abhimanyu Goyal, Samuel Andermatt, Patrick Siegler, Ben Horn, Assaf Israel, Francesco Pongetti, Chih-Wei "Louis" Chen, Marco Selvatici, Pedro Silva, Kathie Wang, Jackson Tolins, Kelvin Guu, Roey Yogev, Xiaochen Cai, Alessandro Agostini, Maulik Shah, Hung Nguyen, Noah Ó Donnaile, Sébastien Pereira, Linda Friso, Adam Stambler, Adam Kurzrok, Chenkai Kuang, Yan Romanikhin, Mark Geller, ZJ Yan, Kane Jang, Cheng-Chun Lee, Wojciech Fica, Eric Malmi, Qijun Tan, Dan Banica, Daniel Balle, Ryan Pham, Yanping Huang, Diana Avram, Hongzhi Shi, Jasjot Singh, Chris Hidey, Niharika Ahuja, Pranab Saxena, Dan Dooley, Srividya Pranavi Potharaju, Eileen O'Neill, Anand Gokulchandran, Ryan Foley, Kai Zhao, Mike Dusenberry, Yuan Liu, Pulkit Mehta, Ragha Kotikalapudi, Chalence Safranek-Shrader, Andrew Goodman, Joshua Kessinger, Eran Globen, Prateek Kolhar, Chris Gorgolewski, Ali Ibrahim, Yang Song, Ali Eichenbaum, Thomas Brovelli, Sahitya Potluri, Preethi Lahoti, Cip Baetu, Ali Ghorbani, Charles Chen, Andy Crawford, Shalini Pal, Mukund Sridhar, Petru Gurita, Asier Mujika, Igor Petrovski, Pierre-Louis Cedoz, Chenmei Li, Shiyuan Chen, Nic-

colò Dal Santo, Siddharth Goyal, Jitesh Punjabi, Karthik Kappaganthu, Chester Kwak, Pallavi LV, Sarmishta Velury, Himadri Choudhury, Jamie Hall, Premal Shah, Ricardo Figueira, Matt Thomas, Minjie Lu, Ting Zhou, Chintu Kumar, Thomas Jurdi, Sharat Chikkerur, Yenai Ma, Adams Yu, Soo Kwak, Victor Ähdel, Sujeevan Rajayogam, Travis Choma, Fei Liu, Aditya Barua, Colin Ji, Ji Ho Park, Vincent Hellendoorn, Alex Bailey, Taylan Bilal, Huanjie Zhou, Mehrdad Khatir, Charles Sutton, Wojciech Rzadkowski, Fiona Macintosh, Roopali Vij, Konstantin Shagin, Paul Medina, Chen Liang, Jinjing Zhou, Pararth Shah, Yingying Bi, Attila Dankovics, Shipra Banga, Sabine Lehmann, Marissa Bredesen, Zifan Lin, John Eric Hoffmann, Jonathan Lai, Raynald Chung, Kai Yang, Nihal Balani, Arthur Bražinskas, Andrei Sozanschi, Matthew Hayes, Héctor Fernández Alcalde, Peter Makarov, Will Chen, Antonio Stella, Liselotte Snijders, Michael Mandl, Ante Kärrman, Paweł Nowak, Xinyi Wu, Alex Dyck, Krishnan Vaidyanathan, Raghavender R, Jessica Mallet, Mitch Rudominer, Eric Johnston, Sushil Mittal, Akhil Udathu, Janara Christensen, Vishal Verma, Zach Irving, Andreas Santucci, Gamaleldin Elsayed, Elnaz Davoodi, Marin Georgiev, Ian Tenney, Nan Hua, Geoffrey Cideron, Edouard Leurent, Mahmoud Alnahlawi, Ionut Georgescu, Nan Wei, Ivy Zheng, Dylan Scandinaro, Heinrich Jiang, Jasper Snoek, Mukund Sundararajan, Xuezhi Wang, Zack Ontiveros, Itay Karo, Jeremy Cole, Vinu Rajashekhar, Lara Tumeh, Eyal Ben-David, Rishub Jain, Jonathan Uesato, Romina Datta, Oskar Bunyan, Shimu Wu, John Zhang, Piotr Stanczyk, Ye Zhang, David Steiner, Subhajit Naskar, Michael Azzam, Matthew Johnson, Adam Paszke, Chung-Cheng Chiu, Jaume Sanchez Elias, Afroz Mohiuddin, Faizan Muhammad, Jin Miao, Andrew Lee, Nino Vieillard, Jane Park, Jiageng Zhang, Jeff Stanway, Drew Garmon, Abhijit Karmarkar, Zhe Dong, Jong Lee, Aviral Kumar, Luowei Zhou, Jonathan Evens, William Isaac, Geoffrey Irving, Edward Loper, Michael Fink, Isha Arkatkar, Nanxin Chen, Izhak Shafran, Ivan Petrychenko, Zhe Chen, Johnson Jia, Anselm Levskaya, Zhenkai Zhu, Peter Grabowski, Yu Mao, Alberto Magni, Kaisheng Yao, Javier Snaider, Norman Casagrande, Evan Palmer, Paul Suganthan, Alfonso Castaño, Irene Giannoumis, Wooyeol Kim, Mikołaj Rybiński, Ashwin Sreevatsa, Jennifer Prendki, David Soergel, Adrian Goedeckemeyer, Willi Gierke, Mohsen Jafari, Meenu Gaba, Jeremy Wiesner, Diana Gage Wright, Yawen Wei, Harsha Vashisht, Yana Kulizhskaya, Jay Hoover, Maigo Le, Lu Li, Chimezie Iwuanyanwu, Lu Liu, Kevin Ramirez, Andrey Khorlin, Albert Cui, Tian LIN, Marcus Wu, Ricardo Aguilar, Keith Pallo, Abhishek Chakladar, Ginger Perng, Elena Allica Abellan, Mingyang Zhang, Ishita Dasgupta, Nate Kushman, Ivo Penchev, Alena Repina, Xihui Wu, Tom van der Weide, Priya Ponnapalli, Caroline Kaplan, Jiri Simsa, Shuangfeng Li, Olivier Dousse, Fan Yang, Jeff Piper, Nathan Ie, Rama Pasumarthi, Nathan Lintz, Anitha Vijayakumar, Daniel Andor, Pedro Valenzuela, Minnie Lui, Cosmin Paduraru, Daiyi Peng, Katherine Lee, Shuyuan Zhang, Somer Greene, Duc Dung Nguyen, Paula Kurylowicz, Cassidy Hardin, Lucas Dixon, Lili Janzer, Kiam Choo, Ziqiang Feng, Biao Zhang, Achintya Singhal, Dayou Du, Dan McKinnon, Natasha Antropova, Tolga Bolukbasi, Orgad Keller, David Reid, Daniel Finchelstein, Maria Abi Raad, Remi Crocker, Peter Hawkins, Robert Dadashi, Colin Gaffney, Ken Franko, Anna Bulanova, Rémi Leblond, Shirley Chung, Harry Askham, Luis C. Cobo, Kelvin Xu, Felix Fischer, Jun Xu, Christina Sorokin, Chris Alberti, Chu-Cheng Lin, Colin Evans, Alek Dimitriev, Hannah Forbes, Dylan Banarse, Zora Tung, Mark Omernick, Colton Bishop, Rachel Sterneck, Rohan Jain, Jiawei Xia, Ehsan Amid, Francesco Piccinno, Xingyu Wang, Praseem Banzal, Daniel J. Mankowitz, Alex Polozov, Victoria Krakovna, Sasha Brown, MohammadHossein Bateni, Dennis Duan, Vlad Firoiu, Meghana Thotakuri, Tom Natan, Matthieu Geist, Ser tan Girgin, Hui Li, Jiayu Ye, Ofir Roval, Reiko Tojo, Michael Kwong, James Lee-Thorp, Christopher Yew, Danila Sinopalnikov, Sabela Ramos, John Mellor, Abhishek Sharma, Kathy Wu, David Miller, Nicolas Sonnerat, Denis Vnukov, Rory Greig, Jennifer Beattie, Emily Caveness, Libin Bai, Julian Eisenschlos, Alex Korchemniy, Tomy Tsai, Mimi Jasarevic, Weize Kong, Phuong Dao, Zeyu Zheng, Frederick Liu, Fan Yang, Rui Zhu, Tian Huey Teh, Jason Sanmiya, Evgeny Gladchenko, Nejc Trdin, Daniel Toyama, Evan Rosen, Sasan Tavakkol, Linting Xue, Chen Elkind, Oliver Woodman, John Carpenter, George Papamakarios, Rupert Kemp, Sushant Kafle, Tanya Grunina, Rishika Sinha, Alice Talbert, Diane Wu, Denese Owusu-Afriyie, Cosmo Du, Chloe Thornton, Jordi Pont-Tuset, Pradyumna Narayana, Jing Li, Saaber Fatehi, John Wieting, Omar Ajmeri, Benigno Uria, Yeongil Ko, Laura Knight, Amélie Héliou, Ning Niu, Shane Gu, Chenxi Pang, Yeqing Li, Nir Levine, Ariel Stolovich, Rebeca Santamaria-Fernandez, Sonam Goenka, Wenny Yustalim, Robin Strudel, Ali Elqursh, Charlie Deck, Hyo Lee, Zonglin Li, Kyle Levin, Raphael Hoffmann, Dan Holtmann-Rice, Olivier Bachem, Sho Arora, Christy Koh, Soheil Hassas Yeganeh, Siim Põder, Mukarram Tariq, Yanhua Sun, Lucian Ionita, Mojtaba Seyedhosseini, Pouya Tafti, Zhiyu Liu, Anmol Gulati, Jasmine Liu, Xinyu Ye, Bart Chrzaszcz, Lily Wang, Nikhil Sethi, Tianrun Li, Ben Brown, Shreya Singh, Wei Fan, Aaron Parisi, Joe

Stanton, Vinod Koverkathu, Christopher A. Choquette-Choo, Yunjie Li, TJ Lu, Abe Ittycheriah, Prakash Shroff, Mani Varadarajan, Sanaz Bahargam, Rob Willoughby, David Gaddy, Guillaume Desjardins, Marco Cornero, Brona Robenek, Bhavishya Mittal, Ben Albrecht, Ashish Shenoy, Fedor Moiseev, Henrik Jacobsson, Alireza Ghaffarkhah, Morgane Rivière, Alanna Walton, Clément Crepy, Alicia Parrish, Zongwei Zhou, Clement Farabet, Carey Radebaugh, Praveen Srinivasan, Claudia van der Salm, Andreas Fidjeland, Salvatore Scellato, Eri Latorre-Chimoto, Hanna Klimczak-Plucińska, David Bridson, Dario de Cesare, Tom Hudson, Piermaria Mendolicchio, Lexi Walker, Alex Morris, Matthew Mauger, Alexey Guseynov, Alison Reid, Seth Odoom, Lucia Loher, Victor Cotruta, Madhavi Yenugula, Dominik Grewe, Anastasia Petrushkina, Tom Duerig, Antonio Sanchez, Steve Yadlowsky, Amy Shen, Amir Globerson, Lynette Webb, Sahil Dua, Dong Li, Surya Bhupatiraju, Dan Hurt, Haroon Qureshi, Ananth Agarwal, Tomer Shani, Matan Eyal, Anuj Khare, Shreyas Rammohan Belle, Lei Wang, Chetan Tekur, Mihir Sanjay Kale, Jinliang Wei, Ruoxin Sang, Brennan Saeta, Tyler Liechty, Yi Sun, Yao Zhao, Stephan Lee, Pandu Nayak, Doug Fritz, Manish Reddy Vuyyuru, John Aslanides, Nidhi Vyas, Martin Wicke, Xiao Ma, Evgenii Eltyshev, Nina Martin, Hardie Cate, James Manyika, Keyvan Amiri, Yelin Kim, Xi Xiong, Kai Kang, Florian Luisier, Nilesh Tripuraneni, David Madras, Mandy Guo, Austin Waters, Oliver Wang, Joshua Ainslie, Jason Baldridge, Han Zhang, Garima Pruthi, Jakob Bauer, Feng Yang, Riham Mansour, Jason Gelman, Yang Xu, George Polovets, Ji Liu, Honglong Cai, Warren Chen, XiangHai Sheng, Emily Xue, Sherjil Ozair, Christof Angermueller, Xiaowei Li, Anoop Sinha, Weiren Wang, Julia Wiesinger, Emmanouil Koukoumidis, Yuan Tian, Anand Iyer, Madhu Gurumurthy, Mark Goldenson, Parashar Shah, MK Blake, Hongkun Yu, Anthony Urbanowicz, Jennimaria Palomaki, Chrisantha Fernando, Ken Durden, Harsh Mehta, Nikola Momchev, Elahe Rahimtoroghi, Maria Georgaki, Amit Raul, Sebastian Ruder, Morgan Redshaw, Jinhyuk Lee, Denny Zhou, Komal Jalan, Dinghua Li, Blake Hechtman, Parker Schuh, Milad Nasr, Kieran Milan, Vladimir Mikulik, Juliana Franco, Tim Green, Nam Nguyen, Joe Kelley, Aroma Mahendru, Andrea Hu, Joshua Howland, Ben Vargas, Jeffrey Hui, Kshitij Bansal, Vikram Rao, Rakesh Ghiya, Emma Wang, Ke Ye, Jean Michel Sarr, Melanie Moranski Preston, Madeleine Elish, Steve Li, Aakash Kaku, Jigar Gupta, Ice Pasupat, Da-Cheng Juan, Milan Someswar, Tejvi M., Xinyun Chen, Aida Amini, Alex Fabrikant, Eric Chu, Xuanyi Dong, Amruta Muthal, Senaka Buthpitiya, Sarthak Jauhari, Nan Hua, Urvashi Khandelwal, Ayal Hitron, Jie Ren, Larissa Rinaldi, Shahar Drath, Avigail Dabush, Nan-Jiang Jiang, Harshal Godhia, Uli Sachs, Anthony Chen, Yicheng Fan, Hagai Taitelbaum, Hila Noga, Zhuyun Dai, James Wang, Chen Liang, Jenny Hamer, Chun-Sung Ferng, Chenel Elkind, Aviel Atias, Paulina Lee, Vít Listík, Mathias Carlen, Jan van de Kerkhof, Marcin Pikus, Krunoslav Zaher, Paul Müller, Sasha Zykova, Richard Stefanec, Vitaly Gatsko, Christoph Hirnschall, Ashwin Sethi, Xingyu Federico Xu, Chetan Ahuja, Beth Tsai, Anca Stefanoiu, Bo Feng, Keshav Dhandhania, Manish Katyal, Akshay Gupta, Atharva Parulekar, Divya Pitta, Jing Zhao, Vivaan Bhatia, Yashodha Bhavnani, Omar Alhadlaq, Xiaolin Li, Peter Danenberg, Dennis Tu, Alex Pine, Vera Filippova, Abhipso Ghosh, Ben Limonchik, Bhargava Urala, Chaitanya Krishna Lanka, Derik Clive, Yi Sun, Edward Li, Hao Wu, Kevin Hongtongsak, Ianna Li, Kalind Thakkar, Kuanysh Omarov, Kushal Majmundar, Michael Alverson, Michael Kucharski, Mohak Patel, Mudit Jain, Maksim Zabelin, Paolo Pelagatti, Rohan Kohli, Saurabh Kumar, Joseph Kim, Swetha Sankar, Vineet Shah, Lakshmi Ramachandruni, Xiangkai Zeng, Ben Bariach, Laura Weidinger, Tu Vu, Alek Andreev, Antoine He, Kevin Hui, Sheleem Kashem, Amar Subramanya, Sissie Hsiao, Demis Hassabis, Koray Kavukcuoglu, Adam Sadovsky, Quoc Le, Trevor Strohman, Yonghui Wu, Slav Petrov, Jeffrey Dean, and Oriol Vinyals. Gemini: A family of highly capable multimodal models, 2025. URL https://arxiv.org/abs/2312.11805.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks, 2025. URL https://arxiv.org/abs/2504.06821.

Mengsong Wu, Tong Zhu, Han Han, Xiang Zhang, Wenbiao Shao, and Wenliang Chen. Chain-of-tools: Utilizing massive unseen tools in the cot reasoning of frozen language models, 2025. URL https://arxiv.org/abs/2503.16779.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024. URL https://arxiv.org/abs/2404.07972.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023a. URL https://arxiv.org/abs/2210.03629.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023b. URL https://arxiv.org/abs/2210.03629.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23378–23386, 2025.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web agent, if grounded, 2024. URL https://arxiv.org/abs/2401.01614.

Boyuan Zheng, Michael Y. Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, and Yu Su. Skillweaver: Web agents can self-improve by discovering and honing skills, 2025. URL https://arxiv.org/abs/2504.07079.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

A THE USE OF LARGE LANGUAGE MODELS

During the preparation of this manuscript, we utilized a Large Language Model (LLM) as a writing assistant. The role of the LLM was strictly limited to language polishing, which included improving grammar, clarity, and overall readability. The LLM did not contribute to the research ideation, experimental design, methodology, or analysis of the results. All authors have reviewed the final text and take full responsibility for the content of this paper.

B TRAINING DETAILS

B.1 DETAILS OF WEBGRPO

Agent Actions. The agent actions are mostly similar to those defined in *WebArena-Lite*(Liu et al., 2024):

- **Click**: Clicks an element with a specific ID.
- **Hover**: Hovers over an element with a specific ID.
- **Fill**: Types a message into an input box with a specific ID.
- **Search**: Types a message into an input box with a specific ID and presses Enter to initiate a search.
- **Keyboard_Press**: Emulates a specific keyboard key combination.
- **Scroll**: Scrolls the page up or down.
- **Select dropdown option**: Selects an option from a dropdown menu with a specific ID.
- **New tab**: Opens a new tab in the current browser.
- **Tab focus**: Switches focus to a browser tab at a specified index.
- **Close tab**: Closes the current tab.
- **Goto**: Navigates to a specific URL.
- **Go back**: Returns to the previous page.
- **Go forward**: Moves to the next page if available.
- **Exit**: Terminates the operation, returns the response, and exits.

Detailed Training Process of WebGRPO. During WebGRPO training with GLM-4 and Llama3.1 models, we configured 8×H100 GPUs and utilized the FSDP strategy. Each training starts with 8 state-task goal pairs ($P = 8$) and collects 16 trajectories ($N = 16$) per state. The dual uncertainty filtering is set to p=35% and q=45%, with $\beta = 0.4$ to balance reasoning and execution rewards. The maximum KL steps are limited to 10 ($K = 10$). In the periodic filtering schedule, we adopt a linear curriculum strategy with the initial update interval set to $\Delta_0 = 10$ and the curriculum index $\gamma \in 0, 10, 20, 30$, yielding update intervals $\Delta(\gamma) \in 10, 20, 30, 40$. The model trains for a total of 200 steps, optimizing the strategy for efficient and stable learning. The detailed hyperparameter settings are shown in Table 3. The detailed SkillGenesis framework is shown in Algorithm 1.

Cost Analysis. The training cost of the SkillEvo framework mainly comes from two stages: WebGRPO's trajectory-level optimization training and SkillGenesis's skill evolution exploration. In the training phase, WebGRPO used 8×H100 GPUs for approximately 38 hours of training, and SkillGenesis's exploration phase conducted 160 iterations, taking approximately 2.5 hours, with total training GPU hours of approximately 324 GPU hours. In terms of query costs, WebGRPO uses RXERM for trajectory reward modeling. For each successful trajectory, $1 + T$ GPT-4o calls are required (where T is the number of reasoning steps, including T reasoning evaluations and 1 execution evaluation); if the trajectory is unsuccessful, no reasoning evaluation will be performed. The SkillGenesis stage triggers one GPT-4o skill evolving call every 20 iterations. Based on the overall process estimation, the total call volume is approximately 101,000 GPT-4o calls. Among these, RXERM two-stage reward evaluation accumulates approximately 248M tokens (execution reward approximately 220M tokens, reasoning reward approximately 28M tokens). Additional overhead mainly comes from SkillGenesis bringing approximately 42M tokens (skill proposal one-time approximately 64k tokens, skill genesis approximately 42M tokens, skill evolving approximately 50k tokens).

Algorithm 1 SkillGenesis Framework

procedure SKILLGENESIS($P, \{s_0^{(i)}\}_1^P$)
  $\{T_i\} \leftarrow$ `GPT4o.GenerateGoals`($\{s_0^{(i)}\}_1^P$) ▷ Skill Proposal
  $\mathcal{A}_0 \leftarrow \emptyset$; $\mathcal{G}_0 \leftarrow (\emptyset, \emptyset)$ ▷ Initialize skill library and SPG
  for $t = 1$ to T **do**
    $\tau \leftarrow$ `WebGRPO.GenerateTrajectory`(T) ▷ Skill Genesis
    if `Validate`(τ, T) **then**
      $\tau_D \leftarrow$ `TruncatePrefix`(τ)
      $\tau_f \leftarrow$ `ExtendWithPolicy`(τ_D)
      $\mathcal{D}_{\text{called}} \leftarrow$ `ExtractSkills`(τ_f)
      $\mathcal{A}_t \leftarrow \mathcal{A}_{t-1} \cup \mathcal{D}_{\text{called}}$ ▷ Update skill library
    end if
    if $t \bmod Q = 0$ **then** ▷ Periodic Skill Evolution
      Suggestions $\leftarrow$ `GPT4o.ComposeSuggestions`($\mathcal{A}_t, \mathcal{G}_t$)
      for f_{comp} in Suggestions **do**
        if `Validate`(f_{comp}.Task) **then**
          $\mathcal{A}_t \leftarrow \mathcal{A}_t \cup \{f_{\text{comp}}\}$
          $\mathcal{G}_t \leftarrow$ `UpdateSPG`($\mathcal{G}_t, f_{\text{comp}}$)
        end if
      end for
    end if
  end for
  return $\mathcal{A}_t, \mathcal{G}_t$
end procedure

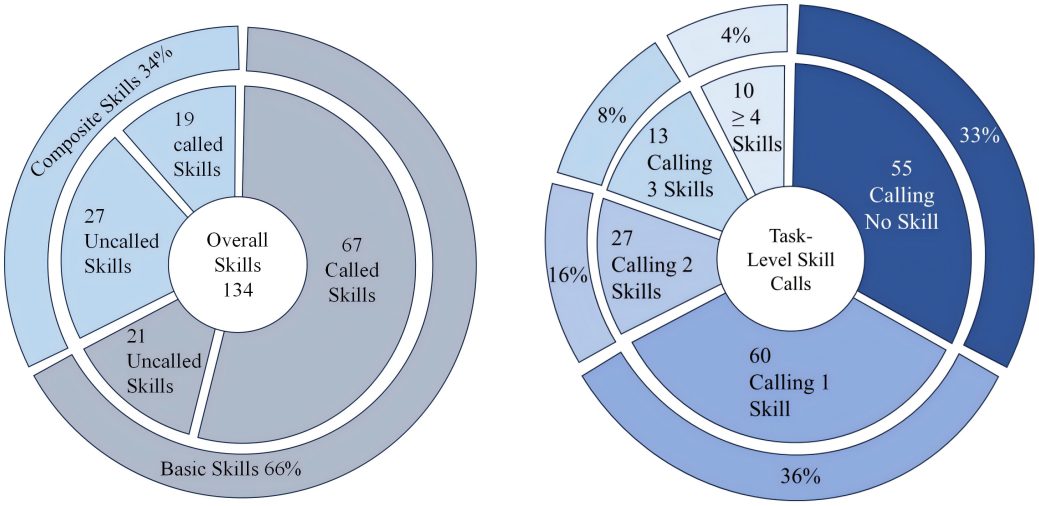

Figure 7: Overall statistics of skills, including distribution and usage frequency.

B.2 Hyperparameters of WebGRPO and Baselines.

C Skill Access and Filtering

In this study, access to and filtering of the skill set is accomplished by directly listing the skill functions in the Prompt. Specifically, the 134 skill functions are divided into 5 batches, each containing about 27 functions, which are embedded in the `<functions>...</functions>` block of the Prompt. Based on the task description, the model determines whether each skill is relevant within every batch Prompt and outputs a JSON result containing step-by-step reasoning and function names. The results from the 5 batches are then merged and deduplicated to obtain a candidate skill

Table 3: The hyperparameters we employ in WEBGRPO and baselines.

Method	Hyperparameter	Value
SFT	learning rate	1e-5
	lr scheduler type	cosine
	warmup ratio	0.1
	batch size	128
	training epoch	1
	cutoff length	16384
Filtered BC	learning rate	1e-6
	lr scheduler type	constant
	batch size	128
	training epoch	1
	cutoff length	16384
	filtering threshold	70th percentile
AWR	actor learning rate	1e-6
	actor lr scheduler type	constant
	critic learning rate	1e-6
	critic lr scheduler type	constant
	batch size	128
	discount factor	0.9
	actor training epoch	1
	critic training epoch	1
DigiRL	actor learning rate	1e-6
	actor lr scheduler type	constant
	critic learning rate	1e-6
	critic lr scheduler type	constant
	instruction value function lr	1e-6
	instruction value function lr scheduler type	constant
	batch size	128
	discount factor	0.9
	actor training epoch	1
	critic training epoch	1
	instruction value function epoch	1
	rollout temperature	1
	replay buffer size	100000
WEBRL	actor learning rate	1e-6
	actor lr scheduler type	constant
	critic learning rate	1e-6
	critic lr scheduler type	constant
	batch size	128
	discount factor	0.9
	actor training epoch	1
	critic training epoch	1
	rollout temperature	1
WEBGRPO	actor learning rate	1e-6
	actor lr scheduler type	constant
	mini batch size	64
	discount factor	0.95
	rollout temperature	1
	validation temperature	0.5
	max prompt length	4096
	maximum response length	512
	KL-divergence loss coefficient	0.001

set. A subsequent ranking Prompt is used to select the Top-15 most relevant skill functions, which are finally injected into the strategy model Prompt for task execution, as shown in Figure 20 and 21.

D SKILLS DISTRIBUTION AND USAGE

As illustrated in Figure 7, in the complete 165 test tasks, we analyzed the coverage rate, call frequency, and complexity distribution of skill calls. After 160 exploration iterations, with composite skill generation triggered every 20 Genesis iterations, the final skill library size reached 134 skills, including 88 basic skills and 46 composite skills. In the test set, 86 skills were actually invoked, with 67 basic skills and 19 composite skills. The total number of skill calls was 193 (approximately 1.17 per task), with basic skill calls accounting for 141 (73%) and composite skills for 52 (27%). Although fewer in number, composite skills exhibited a higher reuse rate, with an average of 2.74 calls per skill, compared to 2.10 for basic skills. Task-level statistics show that 33% of tasks (55) can be completed entirely with atomic operations, without invoking any skills; 36% (60) invoked only 1 skill; tasks requiring 2 and 3 skills account for 16% (27) and 8% (13), respectively; high-complexity tasks requiring 4 skills constitute 6% (10). Thus, while simple tasks can often be solved with atomic operations or few basic skills, high-complexity tasks ($\geq$3 skill calls) significantly depend on skill calls, contributing 41% of all invocations.

E OTHER QUANTITATIVE EXPERIMENTS

E.1 EVALUATION OF REWARD MECHANISMD RXERM

RXERM integrates **Execution Reward** (judging task completion) and **Reasoning Reward** (evaluating intermediate reasoning quality). We validated RXERM on 500 task trajectories using GPT-4o, comparing with human annotations and other evaluators (Claude-3.5-Sonnet, DeepSeek-R1, Gemini-1.5-Pro, GPT-4o-mini). As shown in Figure 8, GPT-4o achieved **92.7% execution accuracy** (second to DeepSeek-R1, 93.5%) and 92.1% reasoning accuracy (best, $\sim$3% higher than Claude-3.5), leading to the highest overall accuracy (92.4%). Given its high consistency with human annotations, we ultimately chose GPT-4o as the RXERM model.

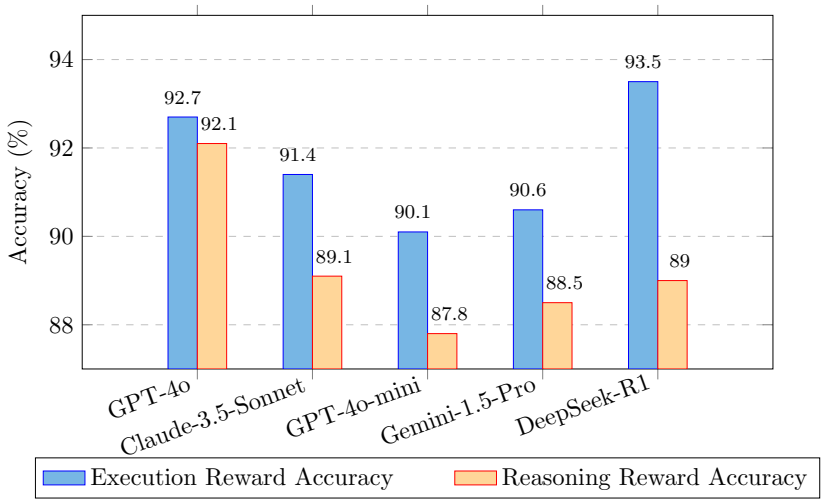

Figure 8: Execution Reward and Reasoning Reward Accuracy of different LLMs.

E.2 ANALYSIS OF MODEL PERFORMANCE VARIANCE

Figure 9 presents the task success rates and error bars of various training methods on WebArena-Lite using the **GLM-4 (Left)** and **LLaMA3.1 (Right)** model series. Each method was evaluated with 5 runs using different random seeds to assess training stability and robustness. Due to the

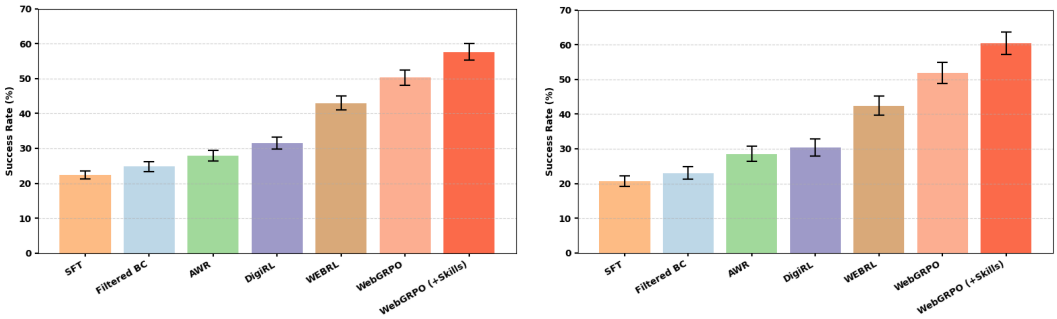

Figure 9: Task success rates and standard deviation error bars of different training methods on WebArena-Lite using the **GLM-4 (left)** and **LLaMA3.1 (right)** model series.

stochastic nature of Skill Proposal, WebGRPO exhibits relatively high variance. Nevertheless, both WebGRPO and its enhanced version, WebGRPO (+Skills), consistently outperform all baselines across both model series, achieving the highest average success rates. Even at the lower bound of the error bars, their performance remains superior, demonstrating the effectiveness and robustness of the proposed approach.

E.3 Sensitive Analysis Reward Balancing Factor α

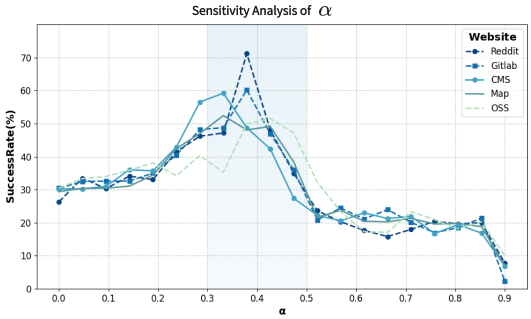

Figure 10: Sensitivity Analysis of α

According to the sensitivity analysis, the choice of α has a significant impact on the success rate across different websites. As shown in Figure 10, the highest success rates for Reddit (0.38), Gitlab (0.38), CMS (0.34), Map (0.33), and OSS (0.42) are achieved when α is between 0.3 and 0.5, indicating the importance of increasing the weight of execution rewards. Both excessively low and high reasoning rewards reduce success rates—too low makes it hard to complete complex tasks, while too high may hinder the successful completion of trajectories. A proper balance between reasoning and execution rewards yields optimal performance.

E.4 Sensitivity Analysis of Formatting Reward Weight β

In multi-turn tasks, reward signals are often sparse, delayed, and heavily outcome-based. As a result, models may produce the correct final answer through trial-and-error or shortcut strategies rather than coherent reasoning. We observe that in the absence of structural constraints, models frequently generate illogical or hallucinated reasoning traces—a phenomenon known as *reasoning collapse*. To address this, we introduce a formatting reward coefficient β, which penalizes outputs that do not conform to the expected `<think>-<answer>` structure, even when the final answer is correct. This encourages the model to maintain interpretable and logically consistent intermediate reasoning steps. As shown in Figure 11a, increasing the formatting reward weight to a moderate level (e.g., $\beta = 0.4$) leads to improved model stability and task success rate. Conversely, when

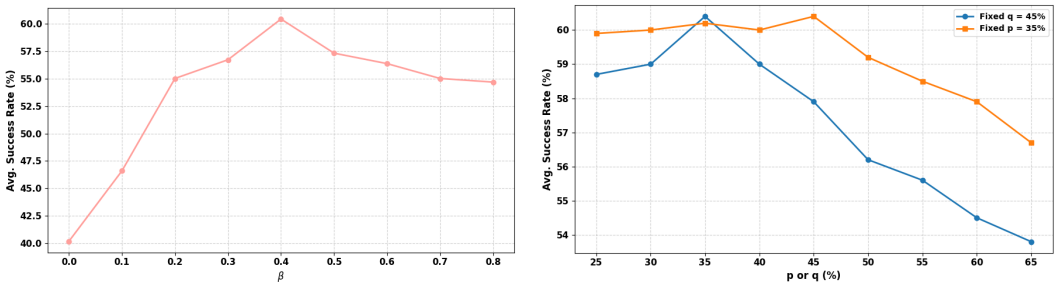

Figure 11: Sensitivity Analysis. **Left:** Sensitivity analysis of the formatting reward weight β. The plot shows the average task success rate across different values of β, which determines the influence of structural correctness in the overall reward. Performance improves significantly as β increases, peaking at $\beta = 0.4$ with a success rate of 60.4%. A too-small or absent formatting reward leads to reasoning collapse and poor learning stability, while overly large values reduce performance due to excessive emphasis on structure. **Right:** Sensitivity analysis when varying percentile thresholds for execution and reasoning uncertainty. The figure shows the average task success rate under two settings: fixing the reasoning uncertainty percentile $q = 45\%$ (blue line) while varying p, and fixing the execution uncertainty percentile $p = 35\%$ (orange line) while varying q. The best performance (60.4%) is observed at $(p = 35\%, q = 45\%)$.

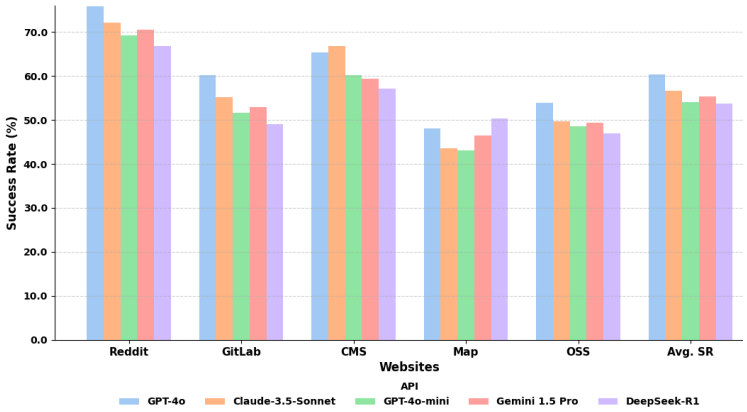

Figure 12: Success rate (%) of different LLMs on five web environments and the overall average in the SkillEvo workflow. GPT-4o achieves the highest average performance and leads on most tasks, while Claude-3.5-Sonnet slightly outperforms others on CMS and DeepSeek-R1 performs best on Map.

the formatting reward is too small or absent, the model's reasoning process often collapses, making policy learning unstable and more difficult to converge.

E.5 Sensitivity Analysis of Dual-Uncertainty Task Filtering

We conduct a systematic sensitivity analysis over the execution uncertainty percentile p and reasoning uncertainty percentile q. As shown in Figure 11b, the highest average task success rate (60.4%) is achieved when selecting the top 35% of task instances based on execution uncertainty and further filtering the top 45% based on reasoning uncertainty. This confirms that selecting moderately uncertain tasks contributes to both informativeness and stable policy optimization. In contrast, increasing either p or q leads to overly permissive filtering, allowing low-information or noisy instances to be included, which can destabilize training and even cause policy collapse. Conversely, excessively small p/q values may discard too many useful samples, slightly lowering performance due to reduced diversity.

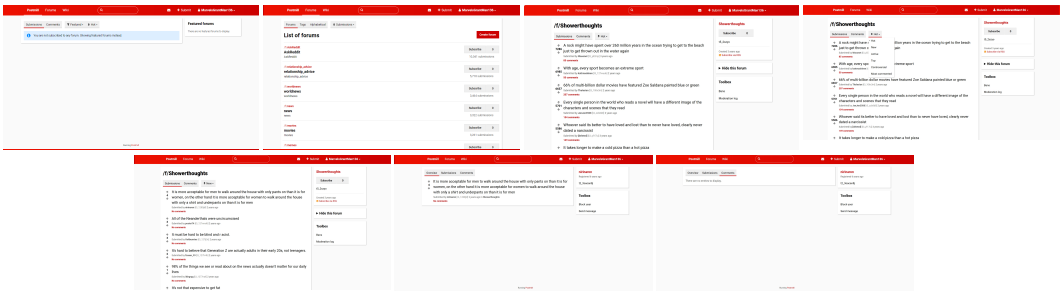

Figure 13: Reddit Example:Tell me the count of comments that have received more downvotes than upvotes for the user who made the latest post on the Showerthoughts forum.

E.6 Performance Comparison of SkillEvo with Different LLMs

In LLaMA3.1 + WebGRPO (+Skills), we compare the performance of SkillEvo during the SkillGenesis stage when implemented with different large language models (LLMs). The figure presents the results of five LLMs—GPT-4o, Claude-3.5-Sonnet, GPT-4o-mini, Gemini-2.5-Pro, and DeepSeek-R1—across five web-based tasks (Reddit, GitLab, CMS, Map, OSS) and their average success rate (Avg. SR). The results show that GPT-4o consistently outperforms others on most tasks and achieves the highest average performance; Claude-3.5-Sonnet performs best on CMS, while DeepSeek-R1 slightly outperforms others on Map. Based on the overall performance, we ultimately select GPT-4o to implement the Skills component in SkillEvo.

F Cases Studies

To qualitatively analyze the impact of the SPG (Skill Path Graph) mechanism on action planning efficiency, we illustrate a representative Reddit example. Figure 13 presents the action trajectory for accomplishing the task "Tell me the count of comments that have received more downvotes than upvotes for the user who made the latest post on the Showerthoughts forum." . This multi-hop reasoning task requires understanding temporal posting order, user identity, and comment-level vote analysis. As shown in Figure 14, compared to the baseline agent (w/o SPG), which completes the task in four discrete steps, the agent with SPG (w/ SPG) achieves the same goal in a single step by optimizing the existing simple skills and composing them with more complex and advanced skills, which effectively improves the efficiency and cost of implementing web tasks.

In the initial stages of task execution, the agent identifies basic atomic skills such as `enter_specific_forum_section` (navigate to a specific forum section), `sort_forum_by_new` (sort posts by newest), and `go_to_user_profile_and_open_comments` (access user profile and open the comments section) to complete relatively simple tasks. As the Skill Path Graph (SPG) is gradually constructed, SkillEvo begins to recognize that these atomic skills are typically called in a specific order to achieve more complex goals. During the skill evolution process, these atomic skills are automatically integrated into more efficient composite skills. For example, the composite skill `analyse_latest_poster_controversial_comments` combines atomic skills such as `enter_specific_forum_section`, `sort_forum_by_new`, `go_to_user_profile_and_open_comments`, and `count_controversial_comments`, forming a complete execution workflow.

As shown in Figure15, the SPG clearly records the dependencies between skills (such as `depends_on`) and invocation edges (such as `invokes`), allowing the agent to replace the individual calls of 3 to 4 atomic skills with a single composite skill. This optimization significantly enhances the efficiency and stability of task execution. In this way, the SPG mechanism not only reduces the number of execution steps but also improves the flexibility and reliability of execution, thereby greatly improving the efficiency of web task execution and reducing costs.

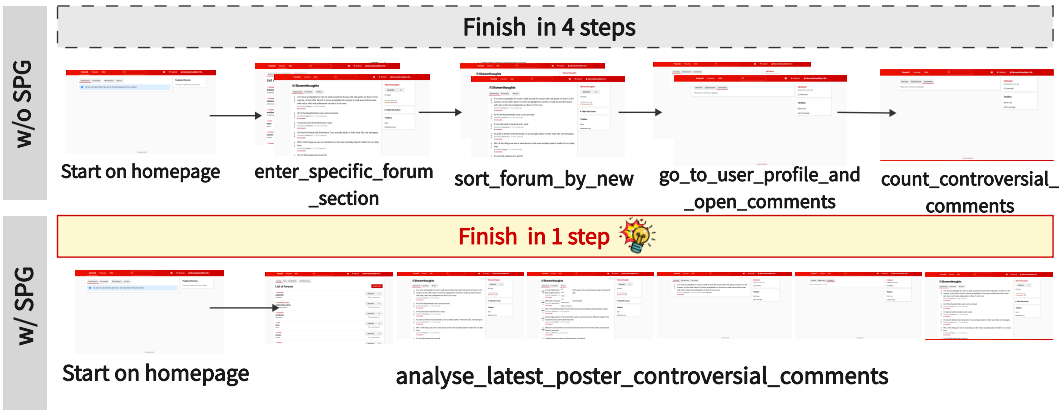

Figure 14: Workflow comparison between the w/o SPG and w/ SPG for task "Tell me the count of comments that have received more downvotes than upvotes for the user who made the latest post on the Showerthoughts forum.".

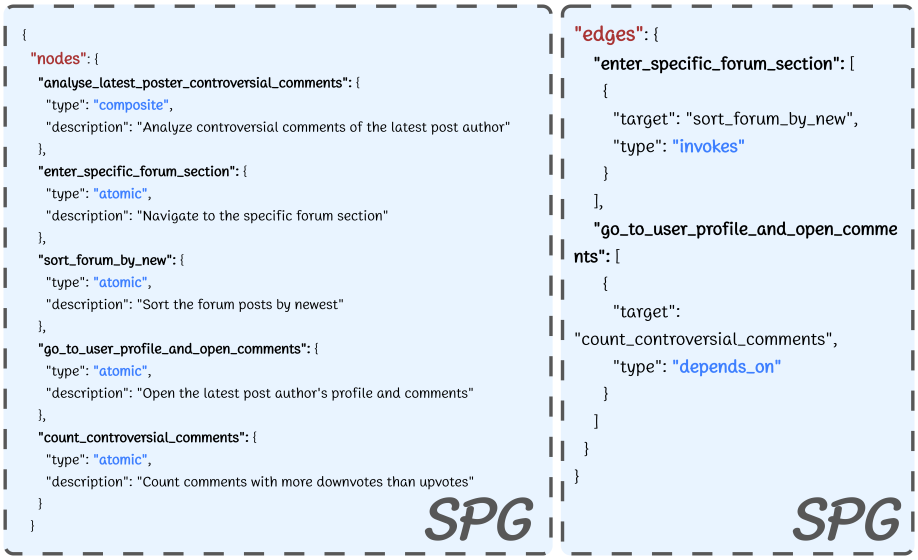

Figure 15: Skill Path Graph (SPG) illustrating the nodes and edges for the task of "counting comments with more downvotes than upvotes for the user who made the latest post on the Showerthoughts forum". The nodes represent both atomic and composite skills, while the edges indicate the relationships between them, such as invocations and dependencies.

G Failure Case Analysis

Our preliminary analysis reveals two major categories of failure. First, **skill invocation degradation** is observed after WebGRPO training. Although the model can recognize appropriate skills during the exploration phase, in the execution phase it often reverts to atomic webpage operations instead of leveraging the skill library. For instance, in the task "view details of a book", the model should invoke the API `search_books_by_title('...')`, but instead fails to call it. This suggests that the policy lacks a global mechanism for evaluating the utility of skill calls, resulting in underutilization of evolved skills. Second, **parameter filling errors** are frequently encountered. In the task "find recipes containing broccoli but no dairy products", the model incorrectly generates the API call `search_recipes_by_ingredients('broccoli, milk-free')`, whereas the correct call should be `search_recipes_by_ingredients('broccoli', exclude='dairy')`. These errors typically stem from insufficient understanding of negation

and exclusion constraints, which leads to semantic misinterpretations when constructing API parameters. Together, these cases highlight current limitations of our approach, particularly in skill integration into policy execution and robust handling of fine-grained constraints.

H Details of Prompts Used in SkillEvo

To enable the evaluation of reasoning and execution behaviors, as well as the generation and verification of skills for web-interacting agents, we have designed a series of carefully crafted prompt templates. These templates cover key aspects such as reasoning rewards, execution rewards, and the core components of the SkillGenesis framework, including skill proposal, validation, and composition.

As illustrated in Figure 22, we designed a reasoning quality evaluation prompt to assess the logical coherence, relevance, and interpretability of the agent's intermediate reasoning during task execution. This evaluation provides fine-grained reward signals. The execution success evaluation prompt, shown in Figure 23, determines whether the agent's sequence of actions successfully achieves the user's goal, serving as a key basis for execution rewards.

For skill generation, we introduced the SkillGenesis prompting system. The skill proposal prompt, shown in Figure 17, guides the agent to propose reusable and high-value interaction skills. The skill validation prompt, shown in Figure 16, evaluates the effectiveness and utility of proposed skills in specific tasks. Additionally, Figure 18 presents a prompt for abstracting reusable functions and rewriting action trajectories, promoting modularity and reusability in agent behaviors and Figure 19 shows a prompt for composing new skills from existing ones, supporting structured skill evolution and enhancing task efficiency. Together, these prompts form the core mechanism that supports the agent's learning and self-evolution process.

I Definition of "Skill"

A **skill** (σ) is defined as an executable Python function that takes two primary inputs: the current webpage state (s) and optional parameters (θ), and returns an execution trajectory (τ) consisting of multiple low-level actions such as clicking, typing, scrolling, etc. This can be expressed as the following formula:

$$\sigma : (s, \theta) \mapsto \tau \tag{11}$$

Where s represents the current webpage state (or its abstract representation), θ represents optional parameters (such as product name, filter conditions, etc.), and τ is the execution trajectory composed of several low-level actions, such as clicks, typing, scrolling, etc.

Semantically, a **skill** represents a reusable high-level operation template that accomplishes specific tasks through a sequence of low-level actions. For example, a typical **skill** could be "search books by title," which is implemented in Python as follows:

```
def search_books_by_title(title: str):
    # Includes several low-level actions.
    ...
```

During execution, the **skill** unfolds into a series of low-level actions, such as typing the book title into a search box and clicking the search button, thereby completing the task. In this way, the SkillGenesis framework abstracts complex web operations into high-level **skills**, enhancing task automation and reusability.

Limitations and Future Work

The current work demonstrates the effectiveness of Skill Path Graphs (SPG) in web environments, but there are several limitations that need to be addressed. Firstly, the experiments were primarily conducted in the WebArena-Lite environment, which includes five real-world websites, limiting the generalizability of the results to larger and more complex environments. While consistent improvements were observed across different websites and model families, future work will focus on

expanding the experiments to cover more websites and real-world scenarios to better assess the scalability and adaptability of the framework.

Secondly, although the core components of SkillEvo (such as WebGRPO + RXERM and SkillGenesis + SPG) are designed to be environment-agnostic, further validation is needed in embodied environments (e.g., long-range tasks based on Habitat/ALFRED) and mobile/GUI automation tasks (e.g., Android control or desktop automation) to evaluate the cross-domain generalization ability. This will help test the stability and effectiveness of the framework across different tasks and environments.

Additionally, as the number of skills increases, the pruning and subgraph retrieval strategies for SPG will face challenges. To handle large-scale skill sets, we plan to explore more efficient pruning algorithms and retrieval mechanisms to ensure that the system can effectively manage and optimize the skill path graph without compromising performance.

In summary, while the current system performs well in specific environments, future research will focus on improving its cross-environment generalization, expanding it to embodied and GUI environments, and optimizing the system to handle larger and more dynamic skill sets.

Skill Verification Prompt:

You are an expert in **verifying the validity of web interaction skills** generated by an autonomous agent. The agent attempts to complete a task using a combination of primitive actions and invoked skills. Please analyze the following based on the trajectory and invoked skills.
The user's goal is: {Task}
You will assess the trajectory against the following three criteria:
1. **Correctness** — Does the agent successfully complete the task goal?
2. **Skill Usage** — Does the trajectory include explicit invocations of reusable skills?
3. **Skill Effectiveness** — Do the invoked skills result in meaningful changes to the webpage state (e.g., navigation, DOM updates, successful form submission)?

[Trajectory]: ## {State-action sequence; includes primitive actions and SKILL calls}

[Invoked Skills]: ## {List of skills invoked in the trajectory}

For each criterion, respond with **YES** or **NO**, followed by a brief explanation.
Finally, answer the following:

Should the invoked skills be added to the skill library?
Respond only with `YES` or `NO`.

Figure 16: Prompt for validating generated skills in the SkillGenesis framework.

Skill Proposal Prompt:

You are a **web agent** learning how to use a website. Your goal is to propose **"skills"** — Python functions that automate common tasks on the website. Each skill should act as a shortcut that combines multiple user interactions into a single reusable function. You are **not allowed** to interact with login/logout/account-related features on any site. If the site uses Magento, avoid the "Advanced Reporting" tab. If the site is OpenStreetMap, do not interact with community features.

You have already proposed the following skills:
<proposed>
##{procedural knowledge}##
</proposed>

The structure of the current website is provided below as HTML:
<html>
##{HTML structure}##
</html>

Please propose a new skill that reflects a task a real user might frequently perform. The skill should be meaningful and combine multiple interactions into a single callable function. **Avoid skills that perform only a single click.**

Follow these design guidelines for each skill:
- The skill name must be expressed in natural language.
- Do **not** use ‘*id‘ as a parameter.
- Each skill should simulate a multi-step sequence of interactions.
- **Prioritize tasks of the following types**:
    -Creating data (e.g., submitting a form)
    -Editing data (e.g., modifying entries)
    -Querying or filtering data (e.g., using search or filters)
- The total number of interactions (clicks, inputs, etc.) must **not exceed 10 steps**.

For each skill, evaluate it across the following three dimensions:
1. **Usefulness (1–3 points)**
    -3: A complex and frequently performed task with high user value.
    -2: A moderately frequent or moderately valuable task.
    -1: A rare or low-impact task.
2. **Generalizability (1–3 points)**
    -3: Can be reused across multiple pages or components.
    -2: Applies only to the current page but has a stable structure.
    -1: Depends on very specific or fragile HTML structure.
3. **Interaction Steps Score (1–10 points)**
    -Count the number of user interactions (click, input, select, etc.).
    -The more steps, the better — skills with more steps are more worthwhile to automate.
    -Maximum allowed steps: **10**

Final Score = Usefulness + Generalizability + Number of Steps

Your response must include a ‘step by step reasoning‘ section explaining the three individual scores and listing each interaction step, followed by a ‘proposed skill‘ section that names the highest-scoring skill using natural language.

Figure 17: Prompt used for skill proposer to propose reusable skills.

Reusable Function Abstraction and Trajectory Rewriting Prompt:

You are a proficient software engineer. Your task is to:
(1) **Summarize reusable functions as APIs** from the provided action trajectories;
(2) **Rewrite the trajectories using the reusable functions** you generated in (1).

Step 1: Generate reusable functions
From the provided trajectory, extract Python functions that encapsulate reusable and meaningful tasks.
- Each function should:
  - Contain **at least 3 actions**, but no more than 10 lines of code;
  - Be **general enough** for reuse in related scenarios;
  - Use **only common variables** as arguments (e.g., strings, lists), **not functions or complex objects**;
  - **Avoid try-except blocks**.
- The only valid actions are:
  - **Click**: Clicks an element with a specific ID.
  - **Hover**: Hovers over an element with a specific ID.
  - **Type**: Types a message into an input box with a specific ID.
  - **Search**: Types a message into an input box with a specific ID and presses Enter to initiate a search.
  - **Press**: Emulates a specific keyboard key combination.
  - **Scroll**: Scrolls the page up or down.
  - **Select dropdown option**: Selects an option from a dropdown menu with a specific ID.
  - **New tab**: Opens a new tab in the current browser.
  - **Tab focus**: Switches focus to a browser tab at a specified index.
  - **Close tab**: Closes the current tab.
  - **Goto**: Navigates to a specific URL.
  - **Go back**: Returns to the previous page.
  - **Go forward**: Moves to the next page if available.
  - **Exit**: Terminates the operation, returns the response, and exits.
- Each function must include the following in its docstring:
  - **Args** – Describe parameters;
  - **Returns** – Describe return values;
  - **Examples** – Show how the function is used.

Step 2: Rewrite trajectories
For each example:
- Provide an **instruction** describing the refactoring;
- Rewrite the original trajectory using the reusable functions;
- Do **not** include any response content or example-specific logic in the function calls.

Important:
- Make sure all used element IDs or URLs are visible in the original trajectory;
- If you use ‘Exit‘, ensure the message is defined **within** the function;
- Each function should contain **2–10 steps only** to ensure simplicity;
- You may generate **zero, one, or multiple functions** depending on the input examples.

Figure 18: Prompt for reusable function abstraction and trajectory rewriting using WebArena-Lite action set.

Skill Composition Prompt:

You are an expert in skill modeling and composition. Your task is to help the system design a new composite skill by combining several existing skills in the skill library. The goal is to improve the efficiency and reusability of web agents in completing complex tasks.

Current Skill Library:
##{skill1, skill2, skill3, ...}##

##Skill Path Graph (SPG):##
- List of skill nodes:
<vt> ##{vt}## </vt>
- List of skill dependency edges:
<et> ##{et}## </et>

In the SPG:
$\rightarrow$ means sequential invocation, e.g., `f1` $\rightarrow$ `f2` $\rightarrow$ `f3`;
$\rightsquigarrow$ means strong dependency, e.g., `f2` $\rightsquigarrow$ `f3`.

Your Task:
1. Select **2–4 existing skills** and design a new composite skill f_{comp};
2. Assign a **natural language name** to the skill;
3. Explain:
- What this skill does;
- When it is useful;
- Why this combination makes sense;
4. Specify:
- The invocation order: `f1` $\rightarrow$ `f2` $\rightarrow$ `f3`;
- Any strong dependencies: `f2` $\rightsquigarrow$ `f3`;
- A validation task T_{comp} that can test this skill.

Output Format:
1.**Skill Name**: <natural language name>
2.**Composed From**: <list of component skills>
3.**Invocation Order**: <e.g., f1 $\rightarrow$ f2$\rightarrow$ f3>
4.**Dependency Edges**: <e.g., f2$\rightsquigarrow$ f3> (optional)
5.**Validation Task**: <task to test the skill>
6.**Why Useful**: <brief explanation>

Notes:
1.The composite skill must have **no more than 10 steps**;
2.Only propose meaningful, reusable, and testable combinations;
3.Return nothing if no reasonable skill composition exists;
4.Focus on complete user flows, frequent patterns, or performance benefits;
5.The skill must be verifiable by outcome.

Figure 19: Prompt for proposing composite skills in the Skill Evolution stage.

Skill Filtering Prompt:

You are provided with a list of Python functions that represent action shortcuts available on a website (in addition to basic actions like click, type, hover, select option, etc.).

##Function Space:##
<functions>...</functions>

##Task Description:##
{repr task}

Your Task:
1. Analyze each function and determine whether it is relevant or potentially useful for the task (not only the most obvious ones).
2. Include all functions that could be useful, even if they are not strictly necessary but might help in completing the task.
3. Output the result in the following JSON format:
{ "step by step reasoning": "Explain your reasoning process", "function names": ["function_name_1", "function_name_2", "function_name_3", ...] }

Figure 20: Prompt for filtering skills in the SkillGenesis framework.

Skill Ranking Prompt:

Now you have a list of candidate functions:
{candidate functions list}

##Task Description:##
{repr task}

Your Task:
1. Rank these functions based on their relevance to the task;
2. Return the Top-15 most relevant functions;
3. Output the result in the following JSON format:
{ "step by step reasoning": "Explain how you determined the ranking", "function names": ["Top1_function_name", "Top2_function_name", ..., "Top15_function_name"] }

Notes:
1. Ensure reasoning is clear and directly tied to task requirements;
2. Return only valid JSON structures;
3. If fewer than 15 functions are relevant, return all that apply.

Figure 21: Prompt for ranking skills in the SkillGenesis framework.

Reasoning Quality Evaluation Prompt:

You are an **expert reasoning evaluator for web-based interactive agents**. Your task is to analyze the agent's intermediate reasoning and assess its quality for reward feedback. Follow the evaluation criteria carefully.

##[Task Goal]:## {Task}
##[Current Webpage State]:## {Html of Current State}
##[Interaction History]:## {Action History}
##[Agent's Reasoning Trace]:## {Reasoning Trace}

Please evaluate the reasoning based on the following four criteria, assigning a score between 0 and 1 for each:
1. **Relevance to Task Goal (0–1)**
- Does the reasoning clearly connect to the specified task goal?
- Are the identified webpage elements or proposed reasoning directly helpful for achieving the task?
2. **Understanding of Webpage State (0–1)**
- Does the reasoning accurately describe relevant elements on the current webpage (e.g., buttons, forms, navigation bars)?
- Is there clear evidence that the agent understands the current structure and state of the webpage?
3. **Logical Consistency of Reasoning (0–1)**
- Does the reasoning exhibit a clear and coherent causal relationship between observations and planned actions?
- Are the proposed actions logically justified, explaining why these actions are necessary to achieve the task goal?
4. **Interpretability and Intermediate Steps (0–1)**
- Does the reasoning explicitly describe intermediate goals or sub-tasks rather than jumping directly to the final goal?
- Are the intermediate steps clearly explained, with reasoning about why each step is necessary?

##[Final Reasoning Quality Score] (Weighted average of the four criteria, range [0, 1]):## {Final Score}

##[Brief Explanation of Evaluation]## (Why did you assign this score?):

IMPORTANT: Only provide the final numerical score and a concise explanation. Do **not** solve the task direct.

Figure 22: Prompt for evaluating agent reasoning quality.

Task Completion Evaluation Prompt:

You are an expert in **evaluating the performance of a website navigation agent**.
The agent is designed to help a human user navigate the website to complete a task.
Please observe the following action history of an agent assisting a user on a website.
The user's goal is: {Task}
Based on the agent's action history and the final screen state, your goal is to determine whether the agent successfully completed the task.
Respond only with `YES` or `NO`.
##[Interaction History]:##
{Action History}
##[Final Webpage State]:##
{Html of Final Webpage State}

Figure 23: Prompt for evaluating agent's task completion status.