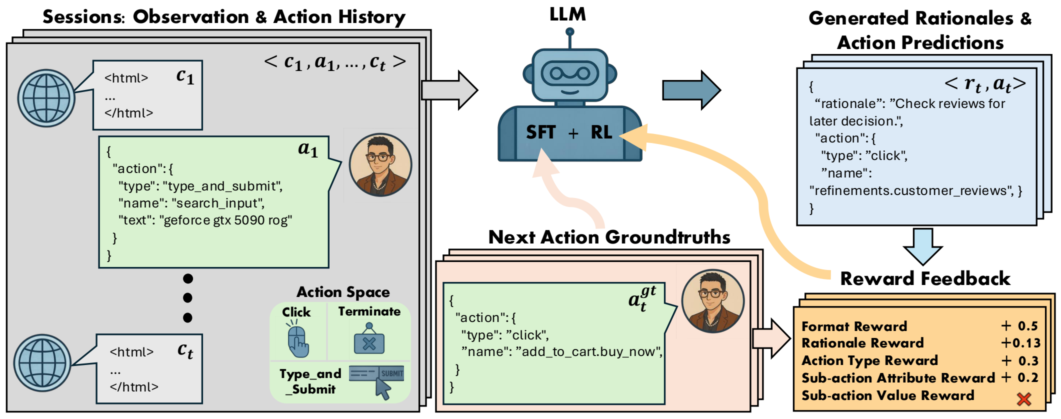
Shop-R1: Rewarding LLMs to Simulate Human Behavior in Online Shopping via Reinforcement Learning

Yimeng Zhang[1] **Ziyi Wang**[2] **Yuxuan Lu**[2] **Sinong Zhan**[3] **Dakuo Wang**[2]

[1]Michigan State University [2]Northeastern University [3]Northwestern University



```
<html>
...
</html>
```

```
{
 "action":{
  "type": "type_and_submit",
  "name": "search_input",
  "text": "geforce gtx 5090 rog"
 }
}
```

```
{
 "rationale": "Check reviews for later decision.",
 "action":{
  "type": "click",
  "name": "refinements.customer_reviews", }
}
```

```
{
 "action":{
  "type": "click",
  "name": "add_to_cart.buy_now",
 }
}
```

Figure 1: Overview of the proposed reinforcement learning framework, *Shop-R1*, designed to simulate real human behaviors in web-based shopping environments. Given an action history $a_{1\dots t-1}$ with corresponding website observations $c_{1\dots t-1}$, the model predicts the next action a_t and its rationale r_t based on the history and the latest website observation c_t. The generated responses are evaluated from four perspectives: format correctness, self-certainty of the rationale, action type accuracy, and sub-action (attribute and value) accuracy.

Abstract

Large Language Models (LLMs) have recently demonstrated strong potential in generating 'believable human-like' behavior in web environments. Prior work has explored augmenting training data with LLM-synthesized rationales and applying supervised fine-tuning (SFT) to enhance reasoning ability, which in turn can improve downstream action prediction. However, the performance of such approaches remains inherently bounded by the reasoning capabilities of the model used to generate the rationales. In this paper, we introduce ***Shop-R1***, a novel reinforcement learning (RL) framework aimed at enhancing the reasoning ability of LLMs for simulation of real human behavior in online shopping environments. Specifically, Shop-R1 decomposes the human behavior simulation task into two stages: rationale generation and action prediction, each guided by distinct reward signals. For rationale generation, we leverage internal model signals (e.g., logit distributions) to guide the reasoning process in a self-supervised manner. For action prediction, we propose a hierarchical reward structure with difficulty-aware scaling to prevent reward hacking and enable fine-grained reward assignment. This design evaluates both high-level action types and the correctness of fine-grained sub-action

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Scaling Environments for Agents (SEA).

details (attributes and values), rewarding outputs proportionally to their difficulty. Experimental results show that our method achieves a relative improvement of over 65% compared to the baseline.

1 Introduction

Large Language Models (LLMs) have shown remarkable performance in planning, reasoning, and decision-making tasks [1–11]. Recently, researchers have begun leveraging LLMs to simulate human behaviors in web-based environments, aiming to generate realistic, user-like action sequences on digital services [12–14]. This capability has promising applications across domains such as e-commerce [15, 16], education [17], and social computing [18]. Despite these advances, current LLM agents often fall short in producing behaviors that align with real humans. The most straightforward baseline is zero-shot prompting [19], where models are given textual instructions to imitate certain user types and output action sequences in a predefined format. While simple to implement, this method lacks the personalization and adaptability needed for high-fidelity behavior modeling [20]. To improve behavioral accuracy and reasoning coherence, recent work such as Lu et al. [20] has introduced synthetic training data augmentation. Specifically, they use Claude 3.5 Sonnet [21] to generate rationales to create ⟨context, action, rationale⟩ triplets. These triplets are then used to perform supervised fine-tuning (SFT), enabling the model to learn both the actions and their underlying rationales. However, this approach faces the key limitations: the quality and diversity of rationales are ultimately constrained by the LLM used during data generation.

Since RL offers a flexible and effective training paradigm, particularly suited for settings with sparse and delayed feedback, and allows for fine-grained control over behavioral outputs [22–26], we utilize RL for the simulation of human shopping behavior, compared to previous work that focuses primarily on task completion [27, 28]. In this work, we propose *Shop-R1*, a novel RL framework designed to enhance LLMs for simulation of human online shopping behaviors. As shown in **Fig. 1**, Shop-R1 decomposes the human behavior simulation task into two stages: (1) *rationale generation* and (2) *action prediction*, with tailored reward signals for each component. For the reward design, we begin by introducing a binary format reward that encourages the model to produce responses in a parse-friendly structure, thereby facilitating reliable downstream evaluation and reward computation. Specifically, the model receives a non-zero reward only when its output conforms to the expected format; otherwise, it is penalized with zero reward. For rationale generation, acquiring ground-truth rationales is inherently difficult. Although efforts like OPeRA [29] attempt to collect self-reported rationales from real users, such annotations may omit implicit or unconscious decision factors. To address this, we incorporate a *self-certainty* reward [30, 31], quantified via the average Kullback–Leibler (KL) divergence [32] between the model's output distribution and a uniform distribution. This signal captures the model's confidence in its generated rationales, providing a supervision-free alternative to ground-truth rationales. For action prediction, we go beyond binary reward signals by introducing a *hierarchical reward scheme* that accounts for both action type and sub-action correctness. This design allows the agent to receive partial credit for plausible but imperfect behaviors, promoting smoother and more robust learning. Furthermore, to mitigate reward hacking and reflect the varying difficulty of different actions, we apply a difficulty-aware reward scaling strategy that adjusts the reward magnitude based on action complexity. Our **main contributions** are summarized as follows:

- To the best of our knowledge, we are the first to introduce RL into a simulation-oriented human behavior modeling task. We reformulate human online shopping behavior simulation as a two-stage prediction problem, comprising rationale generation and action prediction, and design distinct RL objectives for each.
- We introduce *Shop-R1*, a reinforcement-learning framework with a hybrid reward design. It integrates a *self-certainty* signal for rationale generation with a *hierarchical reward scheme* for action prediction. To ensure stable learning and prevent reward hacking, we further introduce a *format reward* and a *difficulty-aware reward scaling* mechanism.
- Experiments show that our proposed training pipeline achieves an exact match accuracy of 27.72%, outperforming supervised fine-tuning (16.76%) by over 65%, demonstrating the strong effectiveness of our approach in simulation-oriented human shopping behavior

modeling. We further conduct a comprehensive ablation study to evaluate the contribution of each component in our design.

2 Related Work

LLM for human behavior simulation. Large Language Models (LLMs) have emerged as powerful tools to simulate human behaviors in diverse real-world settings. Recent advances have led to the development of agent systems capable of generating plausible user actions based on static personas and interaction histories, enabling the modeling of behavior in contexts such as social science [33, 34], recommender systems [35], and user experience research [36]. These systems typically condition on user profiles (e.g., preferences, demographics) and session histories (e.g., clickstreams, task sequences) to predict the next likely user action, allowing for personalized and context-aware simulations. Beyond behavior prediction, recent efforts have enriched these simulations by incorporating explicit reasoning processes. Methods like ReAct [1] and reflection-based models [37, 38] prompt LLMs to produce intermediate thought traces before action generation, enhancing interpretability and decision quality. Systems such as WebAgent [39] and UX-Agent [36] further decompose tasks into sub-goals using dedicated reasoning models, yielding improved control in complex environments like web interfaces. A parallel line of research explores agent-based LLM frameworks that simulate multi-agent interactions in dynamic environments [40–42]. These systems often adopt modular roles (e.g., planners, executors) and collaborative reasoning [43, 44], offering insights into emergent social behaviors and teamwork dynamics. Despite recent advances, there remains a significant gap in exploring how RL can be leveraged to further enhance the simulation of human behavior using LLMs, particularly in the context of web-based shopping environments.

Reward design for RL. Reward design plays a central role in the effectiveness and generalization of RL algorithms, particularly in the context of aligning LLMs with desired behaviors. The prominent paradigm is Reinforcement Learning from Human Feedback (RLHF), which has been widely adopted to fine-tune LLMs using reward models trained on human preference data [45]. While RLHF has demonstrated strong alignment capabilities, it is often bottlenecked by the high cost and limited scalability of collecting reliable human annotations [46]. Moreover, reward models themselves can introduce alignment biases and inaccuracies, especially when trained on limited or noisy preference comparisons [47]. To alleviate these limitations, Direct Preference Optimization (DPO) [23] proposes a more efficient alternative that directly optimizes model parameters against human preference signals without an explicit reward model. Though computationally lighter, DPO and its variants still depend on the availability and quality of human-generated or approximated preference data, which can be inconsistent across tasks and domains. A complementary direction has emerged through Reinforcement Learning with Verifiable Rewards (RLVR), particularly suited for domains with deterministic correctness criteria such as code generation and mathematical reasoning [48, 49]. RLVR frameworks employ rule-based verifiers to automatically compute reward signals based on strict correctness (e.g., exact string matching or functional equivalence) bypassing the need for human feedback. This shift toward automated objective reward functions has enabled the training of highly capable models such as DeepSeek-R1 [48] and inspired new policy optimization methods such as GRPO [50] and its recent extensions [51, 52]. Despite these advances, reward design remains a fundamental challenge in RL for human behaviors. RLHF offers flexibility for modeling subjective tasks, but often suffers from scalability and reliability issues [53–56]. In contrast, RLVR provides high precision by relying on clearly defined evaluation criteria, but is limited to tasks where such criteria exist [49, 57, 58]. To address the unique challenges of simulating human online shopping behavior, we propose a hybrid reward framework specifically tailored to this domain.

3 Methodology

In this section, we first formulate the problem of human behavior simulation in the context of web-based shopping. We then present the design of our proposed RL framework, *Shop-R1*, tailored specifically for simulating human behavior in this setting.

Problem statement. In the context of web shopping, a user session is composed of a sequence of multi-step actions $a_{1 \dots t \dots N}$, typically initiated with a search query and concluded by either a product purchase or a termination action (e.g., closing the browser). Following the setup of [20], the action space comprises three primary action types: '*type_and_submit*', '*click*', and '*terminate*'. More details about the action space can be found in App. A. Each action a_t is paired with a corresponding rationale r_t, which captures the user's underlying motivation or rationale at that time step. The model also receives contextual information, i.e., the observation space, representing the current state of the web environment. This context is encoded as a simplified HTML structure, as introduced in Lu et al. [13], which preserves essential layout and content elements while discarding non-informative components such as scripts and styles. The task of human online shopping behavior simulation is formally defined as learning a function f that predicts the next rationale and action, given the cumulative context and action history:

$$f(c_{1\dots t}, a_{1\dots t-1}, r_{1\dots t-1}) = r_t, a_t, \tag{1}$$

where f denotes the model trained to simulate user behavior by generating the next-step rationale r_t and action a_t conditioned on prior context $c_{1\dots t}$, past actions $a_{1\dots t-1}$, and prior rationales $r_{1\dots t-1}$. These rationales are generated using LLMs and serve as supervision signals during the supervised fine-tuning stage for a cold start. Need to note that *no* generated rationales are used during the subsequent RL stage.

Cold start with SFT. Following the approach of Guo et al. [48], we initialize the behavior simulation model f through supervised fine-tuning (SFT) on annotated trajectories, where each rationale is generated by Claude 3.5 Sonnet [21] via Amazon Bedrock, without leveraging any user profile information. This SFT phase acts as a cold start for subsequent RL, grounding the model in realistic rationale and action patterns. During this phase, the model is trained to jointly generate rationales and corresponding actions. The training objective is to maximize the likelihood of the ground truth rationale-action pairs, conditioned on the the input query $q_t = c_{1\dots t}, a_{1\dots t-1}, r_{1\dots t-1}$:

$$L_{\text{sft}} = -\sum_{t=1}^{N} \log p(r_t, a_t \mid q_t), \tag{2}$$

This supervised initialization plays a crucial role in helping the model internalize the structural dependencies among context, rationale, and action early in the training pipeline. By grounding the model in these patterns upfront, we significantly enhance both the stability and sample efficiency of subsequent RL stages. More importantly, it provides an explicit signal for what constitutes a high-quality long-text output, such as correctly naming a clicked button or specifying a meaningful search query. These capabilities that are otherwise difficult to acquire solely through RL, especially given the sparse and delayed reward structure.

Shop-R1. To better guide policy optimization in the human behavior simulation setting, we decompose each step into two sub-tasks: rationale generation and action prediction. Each sub-task is assigned a tailored reward to improve alignment and interpretability. To ensure the ease and correctness of parsing predicted rationales and actions from model outputs, we introduce a *binary format reward*, which encourages the model to produce responses in a structured JSON format. This format adheres to a dictionary schema with two keys: *rationale* and *action*. For rationale generation, we employ a *self-certainty score* [30, 31], which quantifies the model's confidence in its generated rationale. Specifically, we compute the KL divergence between the model's predictive distribution over the vocabulary and a uniform distribution, averaged over the entire output sequence:

$$s(r_t \mid q_t) = \frac{1}{N|V|} \sum_{j=1}^{N} \sum_{i=1}^{|V|} p_{ij} \log\left(\frac{p_{ij}}{U_i}\right), \tag{3}$$

where N is the number of tokens in the generated rationale r_t, p_{ij} is the predicted probability of token i at position j, and $U_i = \frac{1}{|V|}$ is the uniform distribution over the vocabulary V. Higher values of $s(\cdot)$ indicate greater certainty and consistency in the model's reasoning. For action prediction, we replace the brittle binary signal with a *hierarchical reward scheme* that credits both the coarse-grained action type and its fine-grained sub-actions to stabilize training and discourage degenerate reward hacked policies. This hierarchical scheme densifies the reward landscape: it expands the set of profitable trajectories, lifts the agent out of the 'no-reward' plateau that typically stalls policy search, and makes reward hacking uneconomical. Concretely, every action, easy or hard, earns the *same* coarse-level reward once its high-level type is correct; only the more complex actions can unlock *additional* gains through their long-text sub-components. As a result, naively spamming the trivial '*terminate*' action no longer yields a competitive payoff, while executing the full ('*click*', '*type_and_submit*') sequence becomes the most lucrative strategy. Concretely, a '*click*' action containing a sub-action,

Table 1: Hierarchical reward schedule with Difficulty-Aware Reward Scaling (DARS). A response earns a format reward of 0.5 if it is in a valid JSON format; otherwise, it gains no format reward. A valid response can further gain partial credit for (i) the correct action type, (ii) the presence of the required sub-action attribute, and (iii) any long-text value prediction, whose reward equals the DARS factor multiplied by its ROUGE-L similarity to the ground truth.

Action Type	Type Reward	Sub-action Attribute Reward	Text-Similarity Value Reward
terminate	0.3	None	None
click	0.3	$+0.2$ (if name $\neq \varnothing$)	+DARS $\times$ ROUGE-L(name)
type_and_submit	0.3	$+0.1$ (if name $\neq \varnothing$) $+0.1$ (if text $\neq \varnothing$)	$+0.1 \times$ ROUGE-L(name) +DARS $\times$ ROUGE-L(text)

specifying the button name to be clicked; partial rewards are granted for the correctly predicted components. Likewise, '*type_and_submit*' contains sub-action, providing the intended textual content. In contrast, '*terminate*' has no sub-actions and is scored only at the action-type level. Prediction accuracy is measured with task-specific metrics: discrete action types use an exact-match criterion, whereas free-form sub-actions are evaluated with ROUGE-L. A text-based sub-action, such as a button label or search query, earns a *soft* reward proportional to its ROUGE-L similarity to the ground truth, but only when that similarity exceeds a preset threshold (e.g., 0.75). Because long-text sub-actions are substantially harder, where modern webpages can expose thousands of candidate elements, we introduce a *difficulty-aware reward scaling* (DARS) factor that amplifies rewards for correctly predicting these components. This prevents reward hacking behaviors in which the agent repeatedly selects the trivial '*terminate*' action to secure easy points. The proposed hierarchical reward scheme is summarized in **Tab. 1**. Bringing these components together, the objective of *Shop-R1*is to maximize the combined reward signal derived from multiple sources, while regularizing with a KL divergence to a reference policy:

$$\max_{\pi_\theta} \mathbb{E}_{r,a\sim\pi_\theta(q)} \left[v(a) + \alpha s(r) + -\beta \, \mathrm{KL} \left(\pi_\theta(r, a \mid q) \,\|\, \pi_{\text{ref}}(r, a \mid q) \right) \right], \tag{4}$$

where π_{ref} denotes a fixed reference policy, $v(a_t)$ denotes the reward for action prediction and α and β are hyperparameters that control the strength of the corresponding regularization terms.

4 Experiments

4.1 Experiment Setups

Datasets and models. Our study is built on a proprietary corpus of 52,137 real-world shopping sessions collected from a leading global e–commerce service. Each session logs the multi-turn interaction between a human customer and the website interface. We enrich each recorded action with a natural language *rationale* automatically generated by Claude 3.5 Sonnet (see Appendix B for the prompting details). The provided observation context is formatted as simplified HTML [13], which retains essential structural elements while filtering out irrelevant content such as scripts, styling information, and user-specific data. For SFT dataset, we keep each session intact. The model is asked to produce the assistant response, which contains both the rationales and the structured action prediction. For RL dataset, we convert a session into a sequence of <context, action> pairs. The context is the concatenation of (i) all previously observed contexts and (ii) the actions already taken; the target is the next action only. Because every session begins on the home page, there is always at least one observed <context, action> pair before the first prediction step, eliminating the open-world ambiguity of the very first move. To provide the model with slightly richer supervision on the harder behaviors, the two complex actions (*click* and *type_and_submit*) each occur about 10% more frequently than the simple *terminate* action. This mild skew prevents the learner from over-fitting to the trivial case while still maintaining near-uniform coverage, thereby supporting fair and informative per-class evaluation. All experiments fine-tune the publicly available `Qwen-2.5-3B-Instruct` model. The default 3B parameter backbone offers a favourable compute–performance trade-off.

Baselines for comparison. We evaluate our approach against several baseline schemes: (a) **Zero-shot prompting**, where the model generates outputs based solely on instruction prompts without additional training; (b) **RL (Binary)**, where the base model is optimized directly with RL, using only a sparse binary reward signal; (c) **SFT-only**, where the model is trained via supervised fine-tuning on data with LLM-generated rationales; (d) **SFT + RL (Binary)**, which extends SFT with reinforcement learning using a binary reward based on exact action match; and (e) **Shop-R1**, our proposed RL framework with hybrid reward design for the simulation-oriented human behavior modeling task.

Training setups. Our codebase is built on verl [59], and all experiments were conducted on NVIDIA A100 GPUs (80 GB). We leveraged Fully Sharded Data Parallelism (FSDP) in PyTorch [60] to maximize training efficiency. The default policy optimization algorithm is Group Relative Policy Optimization (GRPO) [50]. Input sequences were padded or truncated to a maximum context length of 32k tokens, and the default sampling temperature is 0.6. We set the per-device batch size to 1, yielding a global batch size of 64. For supervised fine-tuning (SFT) we trained for 4 epochs with a learning rate of 2×10^{-5}; for reinforcement learning (RL) we trained for 500 steps with a learning rate of 1×10^{-7}. By default, we set the DARS factor to 1000, and use $\alpha = 0.005$ and $\beta = 0.001$ to weight the corresponding reward terms.

Evaluation metrics. We apply an exact match criterion for the accuracy evaluation of predicted user actions. A prediction is deemed correct only when every relevant component exactly matches the ground truth. For instance, in the case of '*click*' actions, both the specific subtype (such as clicking on a filter, search area, or another UI element) and the selected target must align with the true label. Similarly, for '*type_and_submit*' actions, the model should reproduce the similar meaning of input text. Additionally, We report accuracy and F1 on the coarse-grained *action type* alone. Comparing these scores with exact-match accuracy highlights whether residual errors stem from misclassifying the high-level action type or from mistakes in the fine-grained label (button name or query text).

4.2 Experimental Results

Table 2: Simulation accuracy under different fine-tuning methods across models of different sizes. There are three complementary metrics: exact action accuracy (all sub-fields must match the label); action type accuracy, and action type F1 to disentangle mistakes in coarse intent classification from those in long-text arguments.

Model	Settings	Exact Action	Action Type	
		Acc.	**Acc.**	**F1**
Qwen-2.5-3B-Instruct	Zero-shot prompting	0.32%	15.33%	16.15%
	RL (Binary)	1.01%	6.17%	9.92%
	SFT	16.76%	22.25%	24.52%
	SFT + RL (Binary)	16.55%	23.74%	28.07%
	Shop-R1 (Ours)	**27.72%**	**36.40%**	**31.28%**
Qwen-2.5-1.5B-Instruct	Zero-shot prompting	0.53%	3.94%	6.16%
	SFT	10.86%	23.58%	29.02%
	Shop-R1 (Ours)	24.11%	34.54%	29.19%
Qwen-2.5-0.5B-Instruct	Zero-shot prompting	6.76%	12.88%	15.55%
	SFT	9.90%	17.72%	21.61%
	Shop-R1 (Ours)	27.72%	31.83%	21.20%

Performance comparison with baselines. Main performance comparison results are shown in **Tab.** 2. Firstly, zero-shot prompting yields low performance: without any task-specific adaptation Qwen-2.5-3B-Instruct achieves only 0.32% exact-action accuracy, confirming that long-horizon web behavior cannot be recovered from generic instruction tuning alone. Second, RL with sparse binary rewards on their own still fail to give the agent meaningful guidance. When we train the policy from scratch under this signal, it reaches only 1.01% exact-match action accuracy and 6.17% type accuracy. Third, a straightforward round of SFT is more effective, boosting performance to 16.76% exact match accuracy and 22.25% type accuracy. This confirms that dense, teacher-forced trajectories are crucial for injecting the structural knowledge (context $\rightarrow$ rationale $\rightarrow$ action) and illustrating the shape of the long-text fields (button labels or search queries) that the binary signal alone cannot convey. Fourth, appending an additional binary-reward RL phase after SFT delivers only mixed results: exact-match action accuracy actually slips to 16.55%, while type-level F1 rises to 28.07%. The agent thus learns to guess the coarse intent better, but it still struggles to reproduce the long-text values that drive the exact-match metric. In other words, the policy becomes better at guessing the coarse action type, but slightly worse at reproducing the fine-grained, long-text values required for an exact match. Lacking finer-grained credit assignment, the binary objective cannot push the model beyond what SFT already achieves and in some respects even pulls it backwards. Training with this objective is furthermore prone to instability and converges substantially more slowly than optimization with richer, structured rewards. Our proposed *Shop-R1* framework closes most of this gap. By combining hierarchical rewards, self-certainty signals, format rewards and difficulty-aware scaling, it delivers 27.72% exact-action accuracy (+65% relative to SFT) and pushes action-type accuracy and F1 to 36.40% and 31.28%, respectively. The simultaneous rise of both the coarse (type-level) and fine-grained (exact-match) metrics indicates that Shop-R1 not only identifies the correct intent more often, but also reproduces the long-text values (button labels, text queries) with higher fidelity.

Table 3: Exact action accuracy and action type accuracy for each action type: '*click*', '*type_and_submit*', and '*terminate*', across different models and finetuning methods.

Models	**Settings**	**Exact Action Acc. Per Action Type**			**Action Type Acc. Per Action Type**		
		click	*type_and_submit*	*terminate*	*click*	*type_and_submit*	*terminate*
Qwen-2.5-3B-Instruct	Zero-shot prompting	0.58%	0.15%	0.00%	38.7%	1.62%	0.00%
	SFT	4.93%	3.84%	49.80%	8.55%	15.36%	49.80%
	SFT + RL (Binary)	8.12%	3.25%	45.51%	17.25%	13.88%	45.51%
	Shop-R1 (Ours)	7.39%	7.53%	81.84%	10.29%	28.66%	81.84%
Qwen-2.5-1.5B-Instruct	Zero-shot prompting	1.01%	0.15%	0.39%	10.00%	0.44%	0.39%
	SFT	4.49%	7.83%	23.44%	15.07%	32.35%	23.44%
	Shop-R1 (Ours)	3.62%	8.12%	72.85%	6.52%	34.12%	72.85%
Qwen-2.5-0.5B-Instruct	Zero-shot prompting	0.43%	0.15%	24.02%	12.90%	4.43%	24.02%
	SFT	3.19%	7.68%	21.88%	5.94%	26.59%	21.88%
	Shop-R1 (Ours)	0.72%	3.99%	97.07%	1.01%	17.87%	97.07%

As shown in **Tab. 3**, we decompose accuracy by action type. *Zero-shot prompting* shows the classic "intent–content" split. For example, it can guess that a '*click*' is needed (38.7% type accuracy) yet almost never names the exact UI target (0.58 % exact). Even if *SFT* can boost the performance but the gain remains uneven, which suggests that teacher forcing alone does not give the model enough credit assignment signal for predicting high-entropy arguments such as search queries. Appending a sparse binary RL phase after SFT still fails to boost these harder text-generation cases. *Shop-R1* reshapes those incentives, higher exact match accuracy is achieved, indicating that the model is no longer satisfied with merely selecting correct type but is learning to identify the correct widget and query text as well. To be summarized, dense and structured feedback is essential: it overcomes the no-reward plateau, and makes reward hacking uneconomical.

4.3 Ablation Study And Analysis

Model size. Tab. 2 and **Tab. 3** reveal a consistent scaling trend. In the zero-shot regime, the 3B backbone already outperforms its 1.5B and 0.5B counterparts by a factor of $\times 4 \sim 5$ on coarse action–type accuracy, confirming that larger models possess stronger out-of-the-box priors for human behavior simulation at the setting of website shopping. After SFT, all sizes gain, yet the improvement is more pronounced for the two smaller backbones than for the 3B model, suggesting that demonstration learning compensates for limited capacity. *Shop-R1* lifts every backbone to its best operating point, but the shape of the gains differs by scale. The 3B variant reaches the highest overall numbers while distributing its improvements evenly across the two complicated action types. By contrast, the 0.5B model achieves a comparable headline exact match accuracy (27.72%) almost entirely by over-predicting the easiest '*terminate*' action (97.07% exact) and largely ignoring the more semantically demanding classes. The 1.5B backbone sits in between, recovering moderate fidelity on '*click*' and '*type_and_submit*' while retaining a strong but not overwhelming bias toward '*terminate*'. In short, scaling primarily augments the model's ability to handle *long-text, high-entropy* actions; smaller networks can still match aggregate accuracy by exploiting the high-reward termination branch, but they do so at the cost of behavioral diversity. These findings underscore that, although Shop-R1 markedly mitigates capacity limitations, genuine mastery of simulation-oriented web-shopping action prediction tasks continues to benefit from larger backbones.

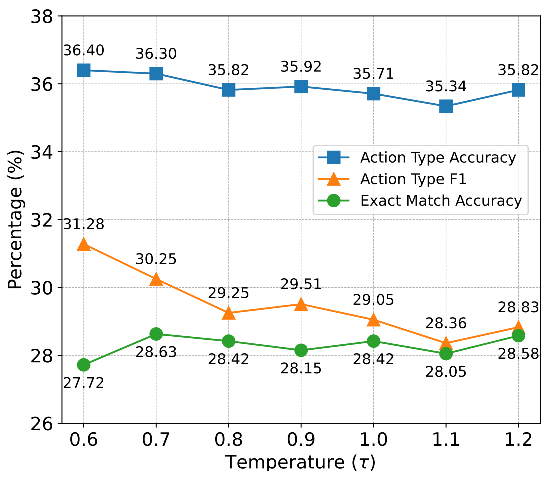


Figure 2: Sampling temperature ablation study.

Sampling temperature. Fig. 2 shows that *Shop-R1* is robust to sampling temperature, yet the three evaluation metrics react in distinct ways that reveal how temperature sampling propagates through the decision hierarchy. Action-type accuracy remains almost constant (36%) across the entire temperature sweep because this metric aggregates all predictions: small mis-classifications in one direction are largely offset by fixes in another, leaving the overall hit rate unchanged. By contrast, the F1 score declines steadily (31.28% $\rightarrow$ 28.36%) as temperature rises; class-averaging penalizes any asymmetric increase in confusion. Interestingly, a modest boost from the default $\tau = 0.6$ to $\tau = 0.7$ improves exact-match accuracy to its peak of 28.63%: a trace of stochasticity helps the generated response escape local maxima and occasionally assemble the full long-text argument that greedy decoding would miss. When $\tau > 0.8$ the added entropy no longer uncovers new correct completions; instead it corrupts fine-grained fields faster than it fixes them, so exact-match plateaus while F1 continues to erode. This pattern is expected since the SFT stage already anchors the model to dataset-specific behavior, privileging faithful simulation over creativity. Taken together, these trends indicate that temperatures in the 0.6–0.8 band offer the best trade-off, preserving robust intent classification, maximizing strict exact-match, and avoiding the metric degradation that emerges once the sampler becomes overly exploratory.

Table 4: Ablation study on different training component configurations, evaluated by exact match action accuracy and action type accuracy / F1.

Model	Training Scheme Components					Exact Action	Action Type	
	SFT	Format Reward	Rationale Reward	Reward Scale	Action Reward	Acc.	Acc.	F1
Qwen-2.5-3B-Instruct	✗	✓	✓	✓	`hierarchical`	4.63%	36.56%	21.92%
	✓	✗	✓	✓	`hierarchical`	2.87%	3.19%	5.04%
	✓	✓	✗	✓	`hierarchical`	26.93%	37.25%	33.74%
	✓	✓	✓	✗	`hierarchical`	27.83%	27.20%	11.70%
	✓	✓	✓	✓	`binary`	27.41%	27.46%	12.11%
	✓	✓	✓	✓	`hierarchical`	**27.72%**	**36.40%**	**31.28%**

Training component. Tab. 4 makes clear that every element of *Shop-R1* addresses a different pathology. Removing the SFT warm-start cripples the agent: despite having all RL signals, exact-match drops to 4.63%, underscoring that a supervised prior is indispensable for learning the shape of long-text arguments. Omitting the format reward is even more destructive, where exact accuracy plunges to 2.87% and type-level metrics fall below 6% since unparseable JSON outputs earn zero credit, starving the learner of gradient signal. When the *self-certainty (rationale) reward* is ablated, coarse intent prediction remains strong but exact-match lags the full system by 0.8%, indicating that explicit feedback on the generated rationales mainly tightens the long-text portion of an action rather than its top-level label. Disabling the *difficulty-aware reward scaling* or reverting to a *binary* action reward leads to a different failure mode: the model still attains around 27% exact accuracy, yet type-level F1 degrades to 11–12%. Inspection shows that, without either scaling or hierarchical credit, the agent gravitates toward the easy high-reward '*terminate*' action and rarely ventures into harder '*click*' or '*type_and_submit*' cases, which is a classic reward-hacking pattern. The full configuration combines all signals and delivers the best balance demonstrating that *each component is necessary*: SFT injects linguistic priors, the format reward safeguards parsability, the self-certainty term refines long-text precision, and hierarchical difficulty-scaled rewards prevent degenerate policies while promoting fine-grained action fidelity.

Whole-session v.s. latest-step context. Tab. 5 isolates the impact of including the *simplified HTML* of each visited page in the action history. Removing this structural cue slashes exact-match accuracy from 27.72% to 14.74%, a nearly 50% relative loss, while coarse action-type accuracy drops more modestly. The sharp divergence indicates that, although the model can still infer *which action type* of interaction is likely next from the dialogue trace alone, it struggles to generate the *fine-grained arguments*, the precise button label or query string, without access to the page's detailed context. Interestingly, the class-balanced F1 score rises slightly, suggesting that the only latest-step context variant compensates by spreading probability mass more evenly across action types, however, this redistribution does not translate into correct long-text completions. In short, supplying even a token-efficient, pruned HTML view is critical for high-fidelity simulation: it grounds the language model in the concrete UI affordances required for exact replay, and although it imposes a substantial overhead on the context window, this cost is justified by its necessity for accurate simulation.

Table 5: Comparison of model performance when using either the whole-session context or only the latest-step context as input.

Context Settings	Exact Action	Action Type	
	Acc.	Acc.	F1
`whole-session`	27.72%	36.40%	31.28%
`latest-step`	14.74%	30.46%	33.48%

5 Conclusion

In this work, we introduced *Shop-R1*, a novel reinforcement learning framework tailored for simulating real human behavior in web-based environments using LLMs. By decomposing the task into two sub-problems, rationale generation and action prediction, and equipping each with carefully designed, structured reward signals, *Shop-R1* addresses key limitations of prior approaches relying solely on supervised fine-tuning or sparse binary rewards. Our hybrid reward scheme incorporating self-certainty scoring, hierarchical credit assignment, format regularization, and difficulty-aware scaling leads to substantial improvements in exact match accuracy and robustness across model sizes. Extensive experiments demonstrate that *Shop-R1* not only surpasses existing baselines by wide margins, but also mitigates common pathologies such as reward hacking and over-reliance on trivial actions. These findings highlight the promise of structured RL frameworks in enabling language agents to perform fine-grained, interpretable, and high-fidelity behavior simulation, paving the way for more realistic and personalized virtual user modeling in future interactive systems.

References

[1] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations (ICLR)*, 2023.

[2] B. Jin, H. Zeng, Z. Yue, J. Yoon, S. Arik, D. Wang, H. Zamani, and J. Han, "Search-r1: Training llms to reason and leverage search engines with reinforcement learning," *arXiv preprint arXiv:2503.09516*, 2025.

[3] X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, and E. Chen, "Understanding the planning of llm agents: A survey," *arXiv preprint arXiv:2402.02716*, 2024.

[4] F. Xu, Q. Hao, Z. Zong, J. Wang, Y. Zhang, J. Wang, X. Lan, J. Gong, T. Ouyang, F. Meng *et al.*, "Towards large reasoning models: A survey of reinforced reasoning with large language models," *arXiv preprint arXiv:2501.09686*, 2025.

[5] Y. Zhang, S. Mao, T. Ge, X. Wang, A. de Wynter, Y. Xia, W. Wu, T. Song, M. Lan, and F. Wei, "Llm as a mastermind: A survey of strategic reasoning with large language models," *arXiv preprint arXiv:2404.01230*, 2024.

[6] C. Sun, S. Huang, and D. Pompili, "Llm-based multi-agent decision-making: Challenges and future directions," *IEEE Robotics and Automation Letters*, 2025.

[7] Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun *et al.*, "Personal llm agents: Insights and survey about the capability, efficiency and security," *arXiv preprint arXiv:2401.05459*, 2024.

[8] J. Gu, X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, W. Li, Y. Shen, S. Ma, H. Liu *et al.*, "A survey on llm-as-a-judge," *arXiv preprint arXiv:2411.15594*, 2024.

[9] J. Jia, Y. Zhang, Y. Zhang, J. Liu, B. Runwal, J. Diffenderfer, B. Kailkhura, and S. Liu, "Soul: Unlocking the power of second-order optimization for llm unlearning," *arXiv preprint arXiv:2404.18239*, 2024.

[10] Y. Zhang, P. Li, J. Hong, J. Li, Y. Zhang, W. Zheng, P.-Y. Chen, J. D. Lee, W. Yin, M. Hong *et al.*, "Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark," *arXiv preprint arXiv:2402.11592*, 2024.

[11] Y. Chen, S. Pal, Y. Zhang, Q. Qu, and S. Liu, "Unlearning isn't invisible: Detecting unlearning traces in llms from model outputs," *arXiv preprint arXiv:2506.14003*, 2025.

[12] C. Chen, B. Yao, R. Zou, W. Hua, W. Lyu, Y. Ye, T. J.-J. Li, and D. Wang, "Towards a design guideline for rpa evaluation: A survey of large language model-based role-playing agents," *arXiv preprint arXiv:2502.13012*, 2025.

[13] Y. Lu, B. Yao, H. Gu, J. Huang, J. Wang, Y. Li, J. Gesi, Q. He, T. J.-J. Li, and D. Wang, "Uxagent: A system for simulating usability testing of web design with llm agents," *arXiv preprint arXiv:2504.09407*, 2025.

[14] D. Wang, T.-Y. Hsu, Y. Lu, L. Cui, Y. Xie, W. Headean, B. Yao, A. Veeragouni, J. Liu, S. Nag *et al.*, "Agenta/b: Automated and scalable web a/btesting with interactive llm agents," *arXiv preprint arXiv:2504.09723*, 2025.

[15] A. Kasuga and R. Yonetani, "Cxsimulator: A user behavior simulation using llm embeddings for web-marketing campaign assessment," in *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024, pp. 3817–3821.

[16] S. Khatuya, R. Vij, P. Koley, S. Datta, and N. Ganguly, "Expert: Modeling human behavior under external stimuli aware personalized mtpp," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 17, 2025, pp. 17 822–17 830.

[17] M. Yao, S. Zhao, S. Sahebi, and R. Feyzi Behnagh, "Stimuli-sensitive hawkes processes for personalized student procrastination modeling," in *Proceedings of the Web Conference 2021*, 2021, pp. 1562–1573.

[18] X. Pan, C. Han, K. H. Law, and J.-C. Latombe, "A computational framework to simulate human and social behaviors for egress analysis," in *Proceedings of the joint international conference on computing and decision making in civil and building engineering*, 2006, pp. 1206–1215.

[19] A. Kong, S. Zhao, H. Chen, Q. Li, Y. Qin, R. Sun, X. Zhou, E. Wang, and X. Dong, "Better zero-shot reasoning with role-play prompting," *arXiv preprint arXiv:2308.07702*, 2023.

[20] Y. Lu, J. Huang, Y. Han, B. Bei, Y. Xie, D. Wang, J. Wang, and Q. He, "LLM Agents That Act Like Us: Accurate Human Behavior Simulation with Real-World Data," Apr. 2025.

[21] Anthropic, "Claude 3.5 sonnet technical overview," https://www.anthropic.com/news/claude-3-5-sonnet, June 2024, https://www.anthropic.com/news/claude-3-5-sonnet.

[22] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier, "A survey of reinforcement learning from human feedback," *arXiv preprint arXiv:2312.14925v2*, 2024.

[23] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 728–53 741, 2023.

[24] T. Mu, A. Helyar, J. Heidecke, J. Achiam, A. Vallone, I. Kivlichan, M. Lin, A. Beutel, J. Schulman, and L. Weng, "Rule based rewards for language model safety," *Advances in Neural Information Processing Systems*, vol. 37, pp. 108 877–108 901, 2024.

[25] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon *et al.*, "Constitutional ai: Harmlessness from ai feedback," *arXiv preprint arXiv:2212.08073*, 2022.

[26] A. Glaese, N. McAleese, M. Trębacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker *et al.*, "Improving alignment of dialogue agents via targeted human judgements," *arXiv preprint arXiv:2209.14375*, 2022.

[27] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried *et al.*, "Webarena: A realistic web environment for building autonomous agents," *arXiv preprint arXiv:2307.13854*, 2023.

[28] Q. Dong, L. Dong, Y. Tang, T. Ye, Y. Sun, Z. Sui, and F. Wei, "Reinforcement pre-training," *arXiv preprint arXiv:2506.08007*, 2025.

[29] Z. Wang, Y. Lu, W. Li, A. Amini, B. Sun, Y. Bart, W. Lyu, J. Gesi, T. Wang, J. Huang, Y. Su, U. Ehsan, M. Alikhani, T. J.-J. Li, L. Chilton, and D. Wang, "Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation," in *arxiv*, 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:279244562

[30] Z. Kang, X. Zhao, and D. Song, "Scalable best-of-n selection for large language models via self-certainty," *arXiv preprint arXiv:2502.18581*, 2025.

[31] X. Zhao, Z. Kang, A. Feng, S. Levine, and D. Song, "Learning to reason without external rewards," *arXiv preprint arXiv:2505.19590*, 2025.

[32] I. Csiszár, "I-divergence geometry of probability distributions and minimization problems," *The annals of probability*, pp. 146–158, 1975.

[33] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," in *Proceedings of the 36th annual acm symposium on user interface software and technology*, 2023, pp. 1–22.

[34] J. S. Park, C. Q. Zou, A. Shaw, B. M. Hill, C. Cai, M. R. Morris, R. Willer, P. Liang, and M. S. Bernstein, "Generative Agent Simulations of 1,000 People," Nov. 2024.

[35] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Fan, X. Huang, Y. Lu, and Y. Yang, "Recmind: Large language model powered agent for recommendation," *arXiv preprint arXiv:2308.14296*, 2023.

[36] Y. Lu, B. Yao, H. Gu, J. Huang, J. Wang, L. Li, J. Gesi, Q. He, T. J.-J. Li, and D. Wang, "UXAgent: An LLM Agent-Based Usability Testing Framework for Web Design," Feb. 2025.

[37] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 8634–8652, 2023.

[38] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative Agents: Interactive Simulacra of Human Behavior," in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '23. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 1–22.

[39] I. Gur, H. Furuta, A. V. Huang, M. Safdari, Y. Matsuo, D. Eck, and A. Faust, "A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis," in *The Twelfth International Conference on Learning Representations*, Oct. 2023. [Online]. Available: https://openreview.net/forum?id=9JQtrumvg8

[40] X. Ma, Z. Zhang, and H. Zhao, "Coco-agent: A comprehensive cognitive mllm agent for smartphone gui automation," *arXiv preprint arXiv:2402.11941*, 2024.

[41] Z. Wang, H. Xu, J. Wang, X. Zhang, M. Yan, J. Zhang, F. Huang, and H. Ji, "Mobile-agent-e: Self-evolving mobile assistant for complex tasks," *arXiv preprint arXiv:2501.11733*, 2025.

[42] OpenAI. (2025) Introducing operator. [Online]. Available: https://openai.com/index/introducing-operator/

[43] C. Qian, W. Liu, H. Liu, N. Chen, Y. Dang, J. Li, C. Yang, W. Chen, Y. Su, X. Cong, J. Xu, D. Li, Z. Liu, and M. Sun, "ChatDev: Communicative Agents for Software Development," Jun. 2024.

[44] Q. Luo, Y. Ye, S. Liang, Z. Zhang, Y. Qin, Y. Lu, Y. Wu, X. Cong, Y. Lin, Y. Zhang, X. Che, Z. Liu, and M. Sun, "RepoAgent: An LLM-Powered Open-Source Framework for Repository-level Code Documentation Generation," Feb. 2024.

[45] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[46] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[47] L. Gao, J. Schulman, and J. Hilton, "Scaling laws for reward model overoptimization," in *International Conference on Machine Learning*. PMLR, 2023, pp. 10 835–10 866.

[48] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[49] Y. Su, D. Yu, L. Song, J. Li, H. Mi, Z. Tu, M. Zhang, and D. Yu, "Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains," *arXiv preprint arXiv:2503.23829*, 2025.

[50] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu *et al.*, "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," *arXiv preprint arXiv:2402.03300*, 2024.

[51] Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, T. Fan, G. Liu, L. Liu, X. Liu *et al.*, "Dapo: An open-source llm reinforcement learning system at scale," *arXiv preprint arXiv:2503.14476*, 2025.

[52] Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du, W. S. Lee, and M. Lin, "Understanding r1-zero-like training: A critical perspective," *arXiv preprint arXiv:2503.20783*, 2025.

[53] D. Alsagheer, A. Kamal, M. Kamal, and W. Shi, "Governance challenges in reinforcement learning from human feedback: Evaluator rationality and reinforcement stability," *arXiv preprint arXiv:2504.13972*, 2025.

[54] H. Lee, S. Phatale, H. Mansoor, T. Mesnard, J. Ferret, K. Lu, C. Bishop, E. Hall, V. Carbune, A. Rastogi *et al.*, "Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback," *arXiv preprint arXiv:2309.00267*, 2023.

[55] S. Casper, X. Davies, C. Shi, T. K. Gilbert, J. Scheurer, J. Rando, R. Freedman, T. Korbak, D. Lindner, P. Freire *et al.*, "Open problems and fundamental limitations of reinforcement learning from human feedback," *arXiv preprint arXiv:2307.15217*, 2023.

[56] T. Moskovitz, A. K. Singh, D. Strouse, T. Sandholm, R. Salakhutdinov, A. D. Dragan, and S. McAleer, "Confronting reward model overoptimization with constrained rlhf," *arXiv preprint arXiv:2310.04373*, 2023.

[57] Y. Mroueh, "Reinforcement learning with verifiable rewards: Grpo's effective loss, dynamics, and success amplification," *arXiv preprint arXiv:2503.06639*, 2025.

[58] X. Wen, Z. Liu, S. Zheng, Z. Xu, S. Ye, Z. Wu, X. Liang, Y. Wang, J. Li, Z. Miao *et al.*, "Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms," *arXiv preprint arXiv:2506.14245*, 2025.

[59] G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu, "Hybridflow: A flexible and efficient rlhf framework," *arXiv preprint arXiv: 2409.19256*, 2024.

[60] Y. Zhao, A. Gu, R. Varma, L. Luo, C.-C. Huang, M. Xu, L. Wright, H. Shojanazeri, M. Ott, S. Shleifer *et al.*, "Pytorch fsdp: experiences on scaling fully sharded data parallel," *arXiv preprint arXiv:2304.11277*, 2023.

Appendix

A System Prompt

```
<IMPORTANT>
Your task is to predict the next action and provide rationale for the action based
    on the previous actions and context.
You need to pretend that you are a user, browsing amazon.com and searching for a
    product to purchase.
The history action (with details described below) and context will be provided to
    you.
You need to predict the next action and provide rationale for the action.
</IMPORTANT>

# Action Space
An action is represented in JSON format, and there are three primary types of
    actions:
#### 1. `type_and_submit`:
Type text into an input field and immediately submit the form. Equivalent to typing
    text into an input and pressing enter key.
{
   "type": "type_and_submit",
   "name": "input_name",
   "text": "search_text"
}

#### 2. `click`:
Click on a button or clickable element identified by `name`.

{
   "type": "click",
   "name": "clickable_name"
}

#### 3. `terminate`:
When you are unsatisfied with the current search result and you don't want to buy
    anything, use `terminate` to indicate that you want to close the browser window
     and terminate the task.
{
   "type": "terminate"
}

# Context
Your context will be an **simplified version** of the raw HTML of the amazon page
    you are looking at. Some interactable elements will be added a unique "name"
    attribute, which you can use to identify the element to interact with (click or
     type_and_submit).

# Rationale
The rationale is a first-person sentence of what you are thinking when you make the
    action. It should be a short sentence that explains why you are making the
    action.

# Output Format
You need to predict the next action and provide rationale for the action. Your
    output should follow a strict JSON form:
{
   "rationale": "<rationale>", // rationale goes here, a string
   "action": {
      // action goes here
      "type": "<type>",
      ...
   },
}
```

```
<IMPORTANT>
OUTPUT A SINGLE JSON OBJECT, NOTHING ELSE.
</IMPORTANT>
```

B Reasoning Synthesize Prompt

```
You will be given a customer's shopping journey on one of the largest e-commerce
    services globally.
You will be given the context (what the user is looking at), the action (what the
    user did), and your job is to predict the user's rationale for the action.
The rationale should follow

Here is an example:
{example}

For each action in the input, output a rationale.

If the action is "terminate", it means that you didn't find any desired product and
    you decided to leave the website by closing the browser window.
```