AGENTIC CRITICAL TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Imitation learning (IL) teaches language-model agents to reproduce expert actions but not to distinguish them from plausible mistakes. Self-reflection methods expose models to alternatives yet use supervised fine-tuning (SFT) to imitate fixed rationales and actions. We introduce **Agentic Critical Training** (ACT), which uses reinforcement learning with verifiable rewards (RLVR) to train models to judge actions directly. At each expert-trajectory state, ACT pairs an expert action with an alternative sampled from the initial policy and randomizes their order. The model generates its own reasoning but is rewarded only for selecting the expert action. ACT reuses demonstrations, requires no reference rationales, and allows pair reuse across model sizes. ACT is a warm-up before IL, optionally followed by RL; inference requires no candidate comparison. Across Qwen3-8B and Olmo-3-7B-Instruct on ALFWorld-ID, WebShop, and ScienceWorld, ACT yields average gains of 5.85 points over IL and 4.12 points over IL→RL without ACT, while also improving ALFWorld-OOD. With Olmo on ScienceWorld, the full pipeline gains 15.36 points over CoT prompting and 9.23 points over IL→RL without ACT. Both ACT→IL and the full pipeline outperform supervised reflection baselines. Controls with fixed pairs or matched training durations show that the gains stem from the ACT objective rather than additional data or training. Without reasoning-specific post-training, the standalone ACT checkpoint achieves the highest mean among evaluated models on MATH-500 and GPQA-Diamond, showing that action comparison complements generation.

1 INTRODUCTION

Language-model agents make a sequence of choices while navigating websites (Yao et al., 2022), manipulating household environments (Shridhar et al.), conducting scientific experiments (Wang et al., 2022), and calling tools (Patil et al., 2024). A single plausible but incorrect action can derail the rest of a trajectory. Most agent post-training methods teach the model to generate the demonstrated next action, but do not explicitly teach it to compare that action with plausible alternatives. We study whether adding such a comparison objective can improve agent training.

This limitation is clearest in imitation learning (IL). IL maximizes the likelihood of expert actions, so each training example identifies what the agent should do at a given state. It does not show which competing actions should be rejected or what makes them worse (Ross et al., 2011; Hussein et al., 2017; Pomerleau, 1991). Early Experience (Zhang et al., 2025) adds information about alternatives by rolling out both the expert and alternative actions and observing their resulting next states. Its implicit-world-modeling variant trains the model to predict next states, while its self-reflection variant synthesizes a rationale that explains why the expert action is better. We compare with the self-reflection variant. Although its targets contain more information than standard behavioral cloning, training still uses next-token prediction to reproduce a fixed rationale and the expert action. The model is given both the explanation and the answer, rather than trained to arrive at the judgment itself, as illustrated in Figure 1(a).

We propose **Agentic Critical Training** (ACT), shown in Figure 1(b), to train this judgment directly. For each state-action pair in an expert trajectory, we sample an alternative action from the initial policy and pair it with the expert action. The two candidates are shown in random order, and the model is asked to reason about them and select the better one. We train the model with reinforcement learning (RL), using a reward based on whether it selects the expert action. The reward does not evaluate the chain-of-thought (CoT; Wei et al., 2022) or require the model to match a reference

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

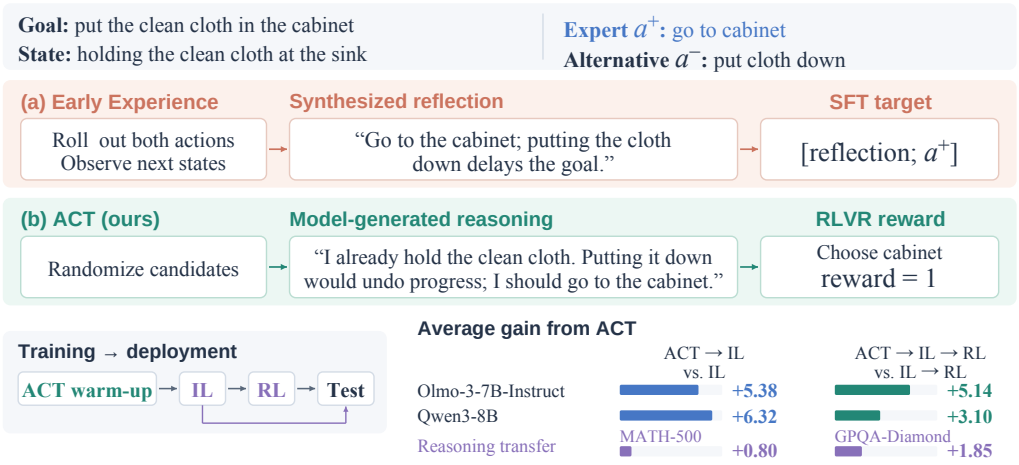

Figure 1: **ACT trains action judgment with a verifiable selection reward. (a)** Early Experience rolls out both candidates and uses SFT to imitate a synthesized reflection and expert action. **(b)** ACT randomizes the pair and rewards expert-action selection without a reference rationale. ACT precedes IL; RL continues from IL along the upper path, and the bypass goes directly to test. Blue and green bars show mean gains from adding ACT on ALFWorld-ID, WebShop, and ScienceWorld. Purple bars compare the ALFWorld-only Qwen3-8B ACT-stage checkpoint with the original Qwen3-8B model on general reasoning. All bars share a 0–7 point scale.

rationale. The supervision therefore evaluates the decision, while allowing the model to develop its own reasoning for that decision.

Building these pairs raises several practical issues. (1) The alternatives should resemble errors that the model is likely to make, (2) labels should be inexpensive to obtain, and (3) the input format should not provide shortcuts to the correct answer. We sample alternatives from the initial policy so that they reflect its own behavior, discard samples that match the expert action, and use the expert action as the preferred candidate under the assumption that expert actions are generally better. This construction requires no rationale generation. Randomizing the candidate order prevents the model from relying on a fixed position. We use one sampled alternative per state in our experiments and also find that pairs collected from an 8B model can train a 4B model, allowing the collection cost to be shared across model sizes.

ACT is a training stage, not an inference-time reranking procedure. It serves as a warm-up before IL; RL, when used, continues from the resulting IL checkpoint. We use **ACT stage** for the action-judgment warm-up, **full ACT pipeline** for ACT→IL→RL, and **IL→RL w/o ACT** for the corresponding baseline. In this sense, ACT bridges the pretrained model and downstream policy training by first building the capacity to evaluate actions. At test time, the resulting policy generates actions directly without candidate pairs.

Across Qwen3-8B and Olmo-3-7B-Instruct (Olmo et al., 2025), ACT warm-up improves both IL and IL→RL on ALFWorld, WebShop, and ScienceWorld, including ALFWorld-OOD. ACT→IL and the full ACT pipeline also outperform supervised reflection baselines. Because these methods use the same expert and alternative actions, the comparison suggests that rewarding a correct judgment is more effective than reproducing a fixed rationale. Same-action-pair and matched-continuation controls show that ACT's gains are not explained by the data or additional training alone (Table 3).

The learned action judgment also transfers beyond the main setting. ACT pairs collected from Qwen3-8B effectively train Qwen3-4B, and the standalone ACT-stage checkpoint achieves the highest reported mean on MATH-500 and GPQA-Diamond without mathematical or scientific reasoning data during post-training. Together, these results suggest that expert trajectories can provide more than actions to imitate: paired with model-generated alternatives, they can also teach the model to distinguish better decisions from worse ones.

In summary, our contributions are:

1. **Training objective.** We propose ACT, which uses RL to train an agent to select the better of an expert action and a model-generated alternative. The reward evaluates the selected action without requiring a reference rationale.
2. **Data construction.** We build contrastive pairs by sampling alternatives from the initial policy, filtering exact matches with the expert action, and randomizing candidate order, without generating reference rationales.
3. **Agent performance.** ACT→IL and the full ACT pipeline improve over their no-ACT counterparts across three agent benchmarks and two model families, while outperforming supervised reflection baselines. Same-pair and matched-continuation controls isolate the benefit of the ACT objective.
4. **Generalization and analysis.** ACT warm-up improves agentic OOD performance and transfers across model sizes, while the ACT-stage checkpoint improves general reasoning without reasoning-specific post-training data. Causal interventions connect reasoning to action selection.

2 Agentic Critical Training

We present Agentic Critical Training (ACT), our approach to training LLM agents beyond pure imitation. We first describe the problem setting (Section 2.1), then detail ACT data construction (Section 2.2), and finally present the training pipeline (Section 2.3). Figure 2 provides an overview.

2.1 Problem formulation

We consider an agent operating in a sequential decision-making environment, formalized as a partially observable Markov decision process (POMDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \Omega, \mathcal{O}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, Ω is the observation space, $\mathcal{O} : \mathcal{S} \to \Delta(\Omega)$ is the observation function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. In practice, the agent conditions on a textual context constructed from the task description, a truncated window of the most recent k observation-action pairs, and the current observation (see prompt templates in Section A.6). We denote this observation-derived context as s_t and use s as shorthand throughout; the policy is thus $\pi_\theta(a_t|s_t)$.

Given an expert demonstration dataset $\mathcal{D}_{\text{expert}} = \left\{\tau^{(n)} = (s_1^{(n)}, a_1^{(n)}, \ldots, s_{T_n}^{(n)}, a_{T_n}^{(n)})\right\}_{n=1}^{N}$, the standard approach is imitation learning (IL), which maximizes the likelihood of expert actions:

$$\mathcal{L}_{\text{IL}}(\theta) = -\mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{expert}}}\left[\log \pi_\theta(a|s)\right], \tag{1}$$

where π_θ is the LLM policy. While effective at teaching agents to replicate expert behavior, IL provides no signal about the relative quality of different actions: the agent learns that a_i is correct at context s_i, but not *why* it is preferable to alternatives. Our goal is to train agents that develop this understanding.

2.2 Data construction

The core idea of ACT is to transform the learning objective from "imitate the expert action" to "identify the better action," requiring the model to develop discriminative understanding of action quality. For each expert state-action pair $(s_i, a_i) \in \mathcal{D}_{\text{expert}}$, we construct an ACT example as follows:

1. **Sample alternatives.** Draw K candidate actions $\{a_i^1, a_i^2, \ldots, a_i^K\}$ from an initial policy $\pi_{\theta_0}(\cdot|s_i)$.
2. **Filter duplicates.** Remove candidates identical to the expert action: $\mathcal{A}_i^{\text{neg}} = \{a_i^j : a_i^j \neq a_i, j \in [K]\}$.
3. **Construct pairs.** Pair the expert action $a_i^+ = a_i$ with each alternative $a_i^- \in \mathcal{A}_i^{\text{neg}}$ to form $|\mathcal{A}_i^{\text{neg}}|$ contrastive examples.

This yields the ACT dataset $\mathcal{D}_{\text{critic}} = \{(s_i, a_i^+, a_i^-)\}_{i=1}^{M}$, where each example contains one expert action and one sampled alternative action. The key assumption is that the initial policy π_{θ_0} generates actions that are, on average, inferior to expert actions. This assumption is well-grounded in our setting: the benchmarks use constrained, discrete action spaces where expert trajectories are collected

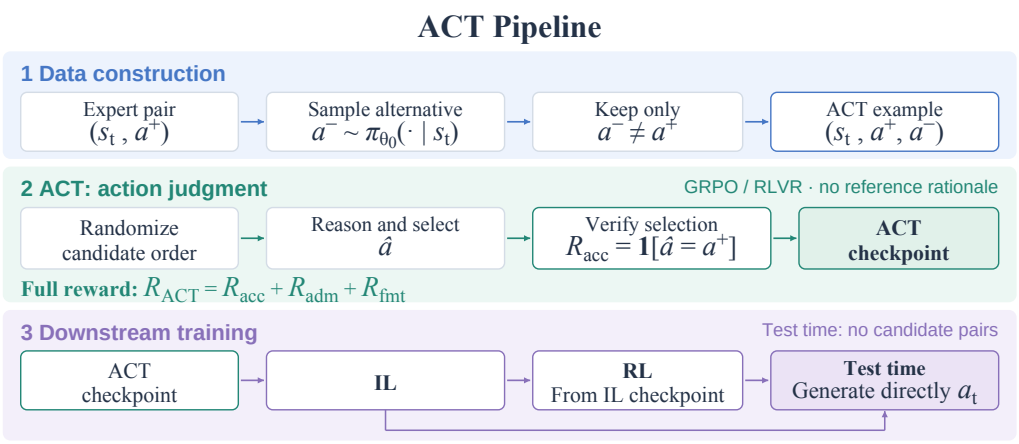

Figure 2: Overview of the ACT training pipeline. **Stage 1** constructs expert–alternative pairs from existing demonstrations. **Stage 2** trains action judgment with GRPO using the action-selection reward in eq. (2). Its accuracy component verifies expert-action selection; admissibility and format terms are detailed in Section A.1 (R_{adm} is disabled for WebShop). **Stage 3** applies IL from the ACT checkpoint, followed by RL from the resulting IL checkpoint or the bypass to test. Both paths produce a policy that generates actions directly without candidate pairs at test time.

from specialized agents or human demonstrations, whereas the initial policy is a general-purpose pretrained LLM without task-specific training. We empirically find that the majority of sampled alternatives differ from the expert action and are identifiably suboptimal given the task context, which is consistent with the large performance gap between the initial policy and the expert.

2.3 Training Pipeline

After constructing $\mathcal{D}_{\text{critic}}$, the model first undergoes ACT with Group Relative Policy Optimization (GRPO; Shao et al., 2024), followed by IL with supervised fine-tuning. Optional RL then continues from the IL or ACT→IL checkpoint. ACT uses an action-level selection reward; details of the ACT objective are provided in Section A.1.

Agentic Critical Training. The model is first trained on $\mathcal{D}_{\text{critic}}$ to identify the better action among two candidates. The ACT prompt is:

ACT Prompt

```
Task: {task_description}

Below are the most recent {k} observations and actions:
{history}

You are now at step {current_step} and your current observation is:
o_t

Your admissible actions of the current situation are:
A_admissible

Candidate Actions for the NEXT step:
Action 1: a_σ(1)
Action 2: a_σ(2)

Which action is better as the NEXT step? Think about why, then
output the chosen action inside <action>...</action> tags.
```

Here σ is a random permutation, so the expert action appears in either position with equal probability.

Given this prompt, GRPO samples G responses, each containing the model's reasoning and action selection. ACT provides no reference rationale: its verifiable reward depends on the final action selection rather than on matching reasoning content. This differs from the self-reflection variant of Early Experience (Zhang et al., 2025), which uses environment outcomes to synthesize rationales and then directly supervises the model to reproduce those rationales together with expert actions. ACT instead optimizes reasoning indirectly according to whether it supports correct selection. The resulting action-evaluation capability can then benefit direct action generation at test time. We examine whether the reasoning is behaviorally connected to selection through interventions in Section 4.4.

IL and RL. After ACT, the model first undergoes IL for direct action generation by maximizing the likelihood of expert actions. RL is optional and starts only from the IL or ACT→IL checkpoint. Importantly, ACT is purely a training-time warm-up: the final policy generates actions directly without the contrastive selection format at test time.

ACT Reward Design. The ACT stage uses the following composite action-selection reward:

$$R_{\text{ACT}}(s, y) = R_{\text{acc}}(a, a^+) + R_{\text{adm}}(a, \mathcal{A}_{\text{admissible}}) + R_{\text{fmt}}(y), \tag{2}$$

where y denotes the full ACT response and $a = \text{extract}(y)$ denotes the selected action extracted from the `<action>...</action>` tags. We set $R_{\text{acc}} = 1$ if a exactly matches the expert action a^+. We set $R_{\text{adm}} = 0.1$ if a is admissible but does not match a^+, which provides partial credit for valid actions. We set $R_{\text{fmt}} = -0.5$ if the response lacks proper `<action>` tags. This action-level reward is used only for the ACT stage. Detailed definitions are provided in Section A.1.

3 Related Work

LLM-based Agents. LLM agents combine language reasoning with sequential action for web navigation (Yao et al., 2022), tool use (Schick et al., 2023; Qin et al., 2024; Su et al., 2026), and multi-step tasks (Yao et al.; Wei et al., 2022). ReAct (Yao et al.) interleaves reasoning and acting, while Reflexion (Shinn et al., 2023) introduces verbal reflection at inference time. ACT instead trains action evaluation through RL, without relying on inference-time reflection prompts.

Training LLM Agents. Agent training commonly relies on imitation learning from expert demonstrations (Chen et al., 2023; Zeng et al., 2024). Early Experience (Zhang et al., 2025) augments demonstrations with outcomes from alternative actions, either predicting next states or synthesizing reflection targets; we compare with its self-reflection variant. SAND (Xia et al., 2025) synthesizes deliberation by critiquing alternatives, while Agent-R (Yuan et al., 2025) constructs revision trajectories from erroneous and corrected paths. These reflection and revision methods incorporate information beyond successful demonstrations, but still train the model to imitate fixed, constructed text targets. ACT instead uses RL to discriminate between expert and model-generated actions: the reward evaluates the final selection without prescribing a rationale.

Critique RL Training. RL has also been used to train critics for reward modeling (Zhang et al., 2026; Chen et al., 2026) or to improve policies through critique generation (Wang et al., 2025; Ruan et al., 2026). These methods primarily critique standalone outputs in single-turn settings. ACT instead learns to distinguish expert from suboptimal actions within a multi-turn decision process.

Agentic RL. Agentic RL extends post-training beyond chat alignment (Ouyang et al., 2022; Rafailov et al., 2023) to multi-turn, long-horizon interaction (Zhang et al.). RL with verifiable rewards can elicit reasoning without supervised chains of thought (Guo et al., 2025); GRPO (Shao et al., 2024), GiGPO (Feng et al.), and Search-R1 (Jin et al.) develop this idea for group-relative optimization, agent credit assignment, and interactive search. ACT contributes a complementary warm-up objective: learning to discriminate expert from suboptimal actions before policy training.

4 Experiments

We evaluate our approach on three diverse benchmarks spanning different agent capabilities. Unless otherwise noted, reported values are the mean ± standard deviation across three runs.

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on **ALFWorld** (Shridhar et al.) for embodied household tasks, **WebShop** (Yao et al., 2022) for web shopping, and **ScienceWorld** (Wang et al., 2022) for multi-step scientific experiments. We report ALFWorld's seen and unseen splits as in-distribution (ID) and out-of-distribution (OOD), respectively. Full evaluation protocols are provided in Section A.8.

Training methods. All training-based methods use the same expert trajectories $\mathcal{D}_{\text{expert}}$. We compare prompting, the standalone **ACT stage**, **IL**, **ACT→IL**, **IL→RL w/o ACT**, and the **full ACT pipeline** (ACT→IL→RL). Here IL uses next-token supervision on expert actions, whereas RL starts from an IL checkpoint and uses GRPO. Details, hyperparameters, and prompts are provided in Section A.

4.2 MAIN RESULTS

Tables 1 and 2 present our main results on Qwen3-8B (Yang et al., 2025) and Olmo-3-7B-Instruct (Olmo et al., 2025), respectively. The two tables use the same methods and evaluation settings across all three benchmarks.

ACT warm-up improves downstream performance across model families. The standalone ACT-stage checkpoint improves over prompting baselines but does not match IL or IL→RL w/o ACT in absolute agent performance, as it trains the model to *judge* actions rather than generate them directly. When used as a warm-up stage, adding ACT improves both downstream pipelines, and the full ACT pipeline achieves the highest mean in all four evaluation settings on both Qwen3-8B and Olmo-3-7B-Instruct. On Qwen3, adding ACT improves IL by 6.32 points and IL→RL w/o ACT by 3.10 points on average across ALFWorld-ID, WebShop, and ScienceWorld. On Olmo 3, the corresponding average gains are 5.38 and 5.14 points. The Olmo ScienceWorld gain is particularly large: the full ACT pipeline improves the mean score by 15.36 points over CoT prompting and by 9.23 points over IL→RL w/o ACT.

ACT→IL and the full ACT pipeline outperform supervised reflection baselines. We compare with Early Experience (Self-Reflection) (Zhang et al., 2025), SAND (Xia et al., 2025), and Agent-R (Yuan et al., 2025), which augment action supervision by training models to generate additional self-reflection, deliberation, or revision text. Across both model families, ACT→IL and the full ACT pipeline outperform all three baselines in every evaluation setting. This suggests that optimizing action-quality reasoning through outcome-based RL is more effective than learning additional synthesized reasoning text through supervised training. Baseline implementations appear in Section A.7.

ACT warm-up improves OOD generalization. On ALFWorld-OOD, adding ACT before IL improves performance from 79.85 to 83.58 on Qwen3 and from 56.97 to 67.41 on Olmo 3. The full ACT pipeline also improves over IL→RL w/o ACT from 90.80 to 91.54 on Qwen3 and from 77.11 to 83.83 on Olmo 3. Thus, the action judgment acquired during the ACT stage transfers to unseen task configurations on both backbones.

4.3 TRAINING-OBJECTIVE AND CONTINUATION CONTROLS

We conduct two sets of controls on Qwen3-8B. First, we vary how the ACT stage is trained while holding its contrastive pairs and downstream IL→RL pipeline fixed. Second, we compare ACT warm-up with additional IL or RL training with matched budgets. Each stage lasts three epochs.

ACT-stage objective controls. We replace ACT-stage RL with SFT-on-choice or DPO, or retain RL while removing the reasoning instruction. As shown in Table 3, the full ACT pipeline achieves the highest mean on ALFWorld-ID, WebShop, and ScienceWorld, while the SFT-on-choice pipeline leads on ALFWorld-OOD. The full ACT objective therefore provides the strongest overall performance across the four settings. Detailed control implementations are provided in Section A.7.

Matched IL continuation. We compare IL→IL, where a three-epoch IL checkpoint receives three additional IL epochs, with ACT→IL, where three ACT epochs are followed by three IL epochs.

Table 1: Main results on Qwen3-8B. ALFWorld and WebShop report success rates (%); ScienceWorld reports mean episode score (0–100).

Method	ALFWorld		WebShop	ScienceWorld
	ID	OOD		
Prompt w/o CoT thinking	32.62±2.63	24.88±2.88	18.20±0.85	16.99±2.85
Prompt w/ CoT thinking	50.00±4.77	39.30±7.72	15.53±0.34	18.68±0.34
ACT stage	62.14±7.72	59.45±9.23	21.53±4.58	30.04±2.13
Imitation Learning (IL)	85.71±0.58	79.85±2.79	28.00±1.31	21.94±2.65
Early Experience	86.43±1.55	79.10±10.57	30.93±1.23	22.82±1.30
SAND	85.71±1.16	83.09±0.93	20.27±1.36	29.41±0.46
Agent-R	87.86±1.17	81.09±0.93	24.07±4.20	16.95±1.79
ACT → IL	89.76±1.88	83.58±3.80	32.00±1.18	32.85±0.15
IL → RL w/o ACT	94.28±1.54	90.80±0.70	39.53±1.06	30.52±0.46
Full ACT pipeline	**96.19±0.67**	**91.54±0.35**	**42.80±0.59**	**34.63±0.33**

Table 2: Main results on Olmo-3-7B-Instruct. ALFWorld and WebShop report success rates (%); ScienceWorld reports mean episode score (0–100).

Method	ALFWorld		WebShop	ScienceWorld
	ID	OOD		
Prompt w/o CoT thinking	0.71±0.58	0.25±0.35	11.47±0.34	3.47±2.04
Prompt w/ CoT thinking	11.91±0.67	10.94±2.14	12.47±0.47	6.31±3.97
ACT stage	16.90±1.69	11.94±3.71	11.53±1.31	12.03±0.92
Imitation Learning (IL)	62.62±3.37	56.97±9.33	12.87±9.02	8.10±0.19
Early Experience	42.62±3.51	35.32±6.40	13.27±7.04	8.97±0.71
SAND	36.91±1.78	28.11±3.52	7.00±0.33	8.11±0.31
Agent-R	66.43±4.67	63.93±4.93	13.40±0.86	9.25±0.70
ACT → IL	74.52±2.21	67.41±4.06	14.80±7.27	10.40±0.89
IL → RL w/o ACT	81.67±2.21	77.11±5.91	23.40±1.23	12.44±1.54
Full ACT pipeline	**87.14±3.08**	**83.83±0.35**	**24.13±0.41**	**21.67±0.61**

ACT→IL exceeds IL→IL by 5.00 points on ALFWorld-ID, 4.47 on ALFWorld-OOD, 0.60 on WebShop, and 10.84 on ScienceWorld. Extending IL barely changes the ScienceWorld score (21.94 to 22.01), whereas ACT→IL reaches 32.85.

Matched RL continuation. We compare the full ACT pipeline (ACT→IL→RL) with IL→RL→RL, which adds three RL epochs to the standard IL→RL checkpoint. Both pipelines receive nine total epochs. The full ACT pipeline exceeds IL→RL→RL by 6.19 points on ALFWorld-ID, 1.74 on ALFWorld-OOD, 9.40 on WebShop, and 14.49 on ScienceWorld.

Additional training alone is insufficient. Three additional IL epochs lower ALFWorld-ID and OOD performance, while three additional RL epochs reduce the mean in all four settings. In contrast, ACT warm-up improves over the corresponding IL or IL→RL baseline in every setting. These controls show that allocating the additional stage to action judgment is more effective than repeating the downstream action-generation objective.

4.4 Causal rationale intervention

We evaluate whether the reasoning generated by the standalone ACT-stage checkpoint is behaviorally connected to its final action selection through inference-time interventions on 400 held-out ALFWorld action pairs. We compare unconstrained generation, direct selection without a rationale, and generation after injecting a plausible but incorrect rationale favoring the alternative action.

Table 3: Training-objective and continuation controls on Qwen3-8B. ALFWorld and WebShop report success rates (%); ScienceWorld reports mean episode score (0–100). Objective controls use the same action pairs and downstream IL→RL pipeline. Continuation comparisons match six total epochs for IL→IL versus ACT→IL and nine total epochs for IL→RL→RL versus the full ACT pipeline. Bold denotes the highest mean within each block.

Method	ALFWorld		WebShop	ScienceWorld
	ID	OOD		
(a) Matched IL continuation				
Imitation Learning (IL)	85.71±0.58	79.85±2.79	28.00±1.31	21.94±2.65
IL → IL	84.76±0.89	79.11±3.71	31.40±2.57	22.01±2.01
ACT → IL	**89.76±1.88**	**83.58±3.80**	**32.00±1.18**	**32.85±0.15**
(b) ACT-stage objective and matched RL controls				
IL → RL w/o ACT	94.28±1.54	90.80±0.70	39.53±1.06	30.52±0.46
IL → RL → RL	90.00±2.10	89.80±3.47	33.40±3.82	20.14±1.09
Full pipeline (SFT-on-choice)	94.76±1.34	**92.78±1.41**	37.73±1.48	30.10±0.19
Full pipeline (DPO)	93.57±0.58	90.05±1.53	41.40±0.99	33.56±0.27
Full pipeline (no-reasoning)	94.52±0.67	92.54±0.61	38.47±2.51	28.12±2.33
Full ACT pipeline	**96.19±0.67**	91.54±0.35	**42.80±0.59**	**34.63±0.33**

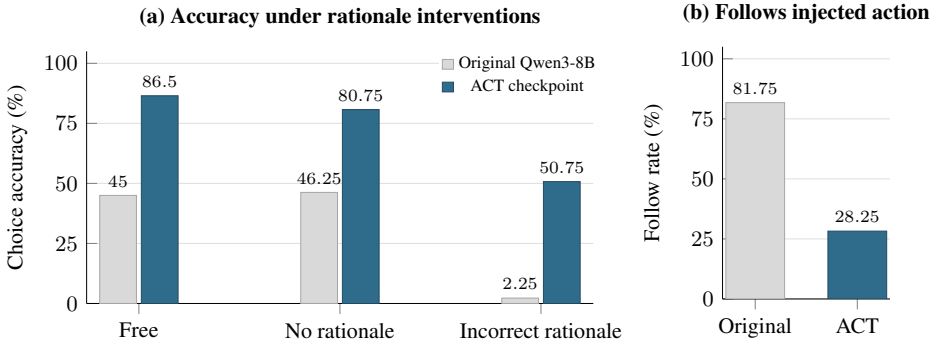

Figure 3: Causal rationale intervention on 400 held-out ALFWorld pairs. **(a)** Choice accuracy under free generation, direct selection without a rationale, and an injected incorrect rationale. **(b)** Fraction of examples in which the model follows the action favored by the injected incorrect rationale. Exact values are provided in Table 6.

As shown in fig. 3, the ACT checkpoint retains 50.75% accuracy while being actively guided toward the incorrect action, compared with 2.25% for the original model, and reduces the rate of following the injected action from 81.75% to 28.25%. Removing the rationale lowers its accuracy from 86.50% to 80.75%, showing that its generated reasoning is behaviorally relevant to the final choice. Full intervention details are provided in Section A.9.

Case studies: action evaluation supports agentic decision-making. On ALFWorld, an IL model repeats the same failed action until termination, while the model trained with the full ACT pipeline diagnoses the root cause and self-corrects. On WebShop, an IL trajectory proceeds to purchase without checking a violated price constraint, illustrating the need for state-aware action evaluation rather than rigid execution. Detailed traces are provided in Section B.

4.5 Generalization to general reasoning benchmarks

Beyond agentic tasks, we investigate whether the critical reasoning capabilities acquired through ACT transfer to general reasoning benchmarks. We directly evaluate the Qwen3-8B checkpoints trained on ALFWorld agentic data, including IL, IL→RL w/o ACT, and the standalone ACT-stage checkpoint, on MATH-500 (Hendrycks et al., 2021) and GPQA-Diamond (Rein et al.). None of these checkpoints receives mathematical or scientific reasoning data during our training.

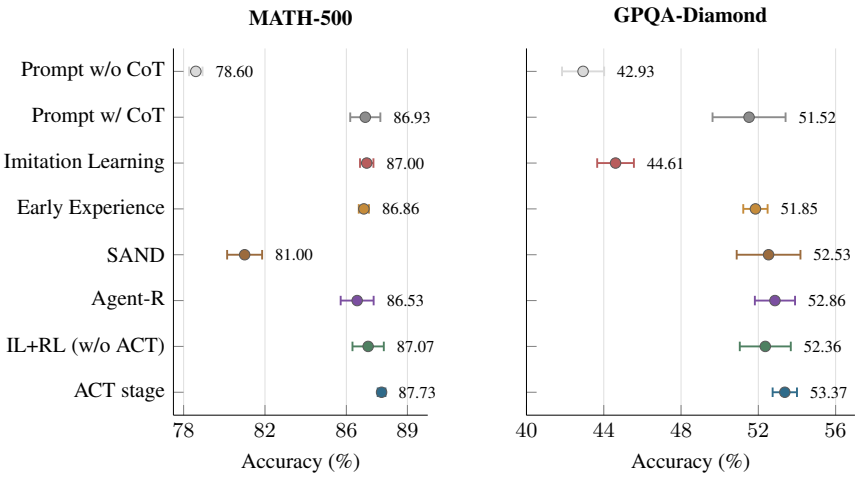

Figure 4: General-reasoning accuracy after training only on ALFWorld agent data. Exact values are provided in Table 8.

As shown in fig. 4, the training paradigms have different effects on general reasoning. IL and the supervised self-reflection baseline, both optimized with next-token objectives against fixed targets, do not yield consistent improvements from agentic training. On MATH-500, both maintain performance comparable to the original model. On GPQA-Diamond, IL degrades performance by 6.91 points compared with the CoT prompting baseline (44.61% vs. 51.52%), while the self-reflection baseline recovers to approximately the prompting baseline (51.85%). Thus, enriching supervised targets with reflection text does not produce the cross-domain reasoning gains observed with ACT.

Among the reflection baselines, SAND notably degrades MATH-500 to 81.00±0.86%, while Agent-R reaches 86.53±0.81% on MATH-500 and 52.86±1.04% on GPQA-Diamond. Neither method surpasses the CoT prompting baseline on both benchmarks simultaneously.

IL→RL w/o ACT roughly preserves the original model's performance on both benchmarks. The standalone ACT-stage checkpoint achieves the highest mean among the models tested on both MATH-500 (87.73±0.19%) and GPQA-Diamond (53.37±0.63%) despite being trained exclusively on agentic data, while IL degrades GPQA-Diamond by 6.91 points. This result indicates that RL in agentic environments, when combined with the ACT objective, can serve as a viable pathway for enhancing general reasoning capabilities.

Case studies: self-verification and reasoning collapse. We examine reasoning traces and observe two phenomena. First, the standalone ACT-stage checkpoint exhibits *self-verification*: on a difficult GPQA-Diamond problem, the model checks its answer by substituting back into the original equations, systematically eliminating incorrect options. Second, an IL trace on another GPQA-Diamond problem is substantially longer and more meandering than the corresponding standalone ACT-stage trace. Detailed examples are provided in Section C.

5 Conclusion

We introduced ACT, which uses RL to train action judgment by contrasting expert and model-generated actions. Unlike supervised reflection methods, ACT evaluates the selected action without prescribing a reference rationale. ACT serves as a warm-up before IL, with optional RL continuing from the resulting IL checkpoint. Across Qwen3-8B and Olmo-3-7B-Instruct on three agent benchmarks, adding ACT improves both downstream pipelines; the full ACT pipeline achieves the highest mean in every evaluation setting and outperforms supervised reflection baselines. Same-action-pair and compute-matched controls show that these gains depend on how the additional stage is trained, rather than additional optimization alone. The improvements extend to unseen agent tasks, while the standalone ACT-stage checkpoint attains the highest mean on both MATH-500 and GPQA-Diamond despite receiving only agentic training. Together, these results suggest that learning to compare actions is a useful training objective whose benefits transfer to agent policies and general reasoning.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and editing the manuscript and with developing and refining code for result visualizations. We also used LLMs to generate alternative actions, which were paired with expert actions from existing demonstrations to construct the synthetic action-comparison training data for ACT. The authors reviewed all AI-assisted text, code, and figures, verified them against the described methods and underlying experimental results, and take responsibility for the final content of this work.

ETHICS STATEMENT

This work uses existing benchmark environments and datasets and does not involve the collection of personal data or new studies with human subjects. Although our experiments are conducted in benchmark environments, improvements in autonomous language-model agents may create risks if such systems are deployed without appropriate safeguards. Real-world applications should therefore use domain-specific validation, restricted permissions, monitoring, and human oversight.

REPRODUCIBILITY STATEMENT

The ACT objective, data construction procedure, and training pipeline are described in Sections 2, 2.2 and 2.3. The appendix provides the complete reward definitions and training algorithm (Section A.1), implementation details and hyperparameters, dataset statistics and expert-trajectory sources (Section A.5), and full prompt templates (Section A.6). It also documents the evaluation and decoding protocol (Section A.8), causal intervention protocol and results (Section A.9), cross-size transfer results (Section A.10), and full general-reasoning results (Section A.11).

REFERENCES

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*, 2023.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 88313–88342, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Jian Hu, Xibin Wu, Wei Shen, Jason Klein Liu, Weixun Wang, Songlin Jiang, Haoran Wang, Hao Chen, Bin Chen, Wenkai Fang, et al. Openrlhf: A ray-based easy-to-use, scalable and high-performance rlhf framework. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 656–666, 2025.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024.

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, volume 2024, pp. 9695–9717, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: international conference for high performance computing, networking, storage and analysis*, pp. 1–16. IEEE, 2020.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Chi Ruan, Dongfu Jiang, Yubo Wang, and Wenhu Chen. Critique-coder: Enhancing coder models by critique reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 19678–19692, 2026.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

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

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

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

Junhao Su, Yuanliang Wan, Junwei Yang, Hengyu Shi, Tianyang Han, Yurui Qiu, and Junfeng Luo. Failure makes the agent stronger: Enhancing accuracy through structured reflection for reliable tool interactions. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 12712–12734, 2026.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, 2022.

Xiyao Wang, Chunyuan Li, Jianwei Yang, Kai Zhang, Bo Liu, Tianyi Xiong, and Furong Huang. Llava-critic-r1: Your critic model is secretly a strong policy model. *arXiv preprint arXiv:2509.00676*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yu Xia, Yiran Jenny Shen, Junda Wu, Tong Yu, Sungchul Kim, Ryan A Rossi, Lina Yao, and Julian McAuley. Sand: Boosting llm agents with self-taught action deliberation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3062–3077, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Siyu Yuan, Zehui Chen, Zhiheng Xi, Junjie Ye, Zhengyin Du, and Jiecao Chen. Agent-r: Training language model agents to reflect via iterative self-training. *arXiv preprint arXiv:2501.11425*, 2025.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077, 2024.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhong-Zhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for llms: A survey. *Transactions on Machine Learning Research*.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, et al. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*, 2025.

Yifan Zhang, Xingyu Lu, Xiao Hu, Chaoyou Fu, Bin Wen, Tianke Zhang, Changyi Liu, Kaiyu Jiang, Kaibing Chen, Kaiyu Tang, et al. R1-reward: Training multimodal reward model through stable reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 31037–31061, 2026.

A EXPERIMENTAL DETAILS

A.1 TRAINING FORMULATION AND ALGORITHM

Reward Function Design. Our composite reward function consists of three components. Given a full generated response y for state s, we first extract the action span inside the `<action>...</action>` tags and denote the extracted action by $a = \text{extract}(y)$. The semantic rewards are applied to the extracted action a, while the format reward is applied to the full response y:

$$R_{\text{ACT}}(s, y) = R_{\text{acc}}(a, a^+) + R_{\text{adm}}(a, \mathcal{A}_{\text{admissible}}) + R_{\text{fmt}}(y). \tag{3}$$

If the response does not contain a valid tagged action span, we set $a = \emptyset$, so the response receives zero semantic reward and only the format penalty applies.

The *accuracy reward* measures exact match between the extracted action and the expert action:

$$R_{\text{acc}}(a, a^+) = \begin{cases} 1.0 & \text{if normalize}(a) = \text{normalize}(a^+) \\ 0.0 & \text{otherwise} \end{cases} \tag{4}$$

The *admissible action reward* provides partial credit for extracted actions that are valid but suboptimal in environments with defined action spaces:

$$R_{\text{adm}}(a, \mathcal{A}_{\text{admissible}}) = \begin{cases} 0.1 & \text{if } a \neq a^+ \wedge a \in \mathcal{A}_{\text{admissible}} \\ 0.0 & \text{otherwise} \end{cases} \tag{5}$$

For ACT on WebShop, the action space includes open-ended search queries (e.g., `search[...]`) that cannot be enumerated, so R_{adm} is disabled and only R_{acc} and R_{fmt} are used.

The *format reward* penalizes full responses that lack proper action tags:

$$R_{\text{fmt}}(y) = \begin{cases} 0.0 & \text{if action tags present} \\ -0.5 & \text{otherwise} \end{cases} \tag{6}$$

GRPO Algorithm. The ACT stage and optional downstream RL stage use Group Relative Policy Optimization (GRPO) (Shao et al., 2024), while the intervening IL stage uses supervised fine-tuning. GRPO is a variant of proximal policy optimization (Schulman et al., 2017) that eliminates the need for a learned value function by estimating advantages from group-level reward statistics. Given a prompt (state) s, GRPO samples a group of G responses $\{y^{(1)}, y^{(2)}, \ldots, y^{(G)}\}$ from the current policy π_θ. Each response receives the reward defined for the current training stage, and the advantage for each response is computed relative to the group statistics:

$$\hat{A}^{(g)} = \frac{r^{(g)} - \bar{r}}{\sigma_r + \epsilon}, \tag{7}$$

where $\bar{r} = \frac{1}{G}\sum_{g=1}^{G} r^{(g)}$ is the group mean reward, $\sigma_r = \sqrt{\frac{1}{G}\sum_{g=1}^{G}(r^{(g)} - \bar{r})^2}$ is the group standard deviation, and ϵ is a small constant for numerical stability. The GRPO objective combines the policy gradient with KL regularization:

$$\begin{aligned} \mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{s\sim\mathcal{D}}\mathbb{E}_{y^{(g)}\sim\pi_\theta(\cdot|s)}\Big[\min\Big(&\rho^{(g)}\hat{A}^{(g)}, \\ &\text{clip}(\rho^{(g)}, 1-\epsilon_c, 1+\epsilon_c)\hat{A}^{(g)}\Big)\Big] \\ &+ \beta \cdot D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \end{aligned} \tag{8}$$

where $\rho^{(g)} = \frac{\pi_\theta(y^{(g)}|s)}{\pi_{\theta_{\text{old}}}(y^{(g)}|s)}$ is the importance sampling ratio, ϵ_c is the clipping threshold, β is the KL penalty coefficient, and π_{ref} is the reference policy. For compactness, the expression suppresses token indices; the clipped objective is applied tokenwise in implementation. We set $\beta = 0$ in all experiments (Table 4).

Algorithm 1 ACT with GRPO

Input: Expert dataset $\mathcal{D}_{\text{expert}}$, initial policy π_{θ_0}, number of candidate samples K, group size G
Output: Trained policy π_θ

// Phase 1: Data Collection
$\mathcal{D}_{\text{critic}} \leftarrow \emptyset$
for each $(s_i, a_i^+) \in \mathcal{D}_{\text{expert}}$ **do**
  Sample $\{a_i^1, \ldots, a_i^K\} \sim \pi_{\theta_0}(\cdot|s_i)$
  $\mathcal{A}_i^{\text{neg}} \leftarrow \{a_i^j : a_i^j \neq a_i^+\}$
  if $|\mathcal{A}_i^{\text{neg}}| > 0$ **then**
    for each $a_i^- \in \mathcal{A}_i^{\text{neg}}$ **do**
      $\mathcal{D}_{\text{critic}} \leftarrow \mathcal{D}_{\text{critic}} \cup \{(s_i, a_i^+, a_i^-)\}$
    end for
  end if
end for

// Phase 2: GRPO Training
Initialize $\theta \leftarrow \theta_0$, $\pi_{\text{ref}} \leftarrow \pi_{\theta_0}$
for each training iteration **do**
  Set $\theta_{\text{old}} \leftarrow \theta$
  Sample batch $\mathcal{B} \subset \mathcal{D}_{\text{critic}}$
  for each $(s, a^+, a^-) \in \mathcal{B}$ **do**
    Construct ACT prompt p with randomized positions
    Sample $\{y^{(1)}, \ldots, y^{(G)}\} \sim \pi_{\theta_{\text{old}}}(\cdot|p)$
    Compute $r^{(g)} = R_{\text{ACT}}(s, y^{(g)})$ for $g = 1, \ldots, G$
    Compute $\hat{A}^{(g)}$ via eq. (7) for $g = 1, \ldots, G$
  end for
  Update θ by minimizing $\mathcal{L}_{\text{GRPO}}(\theta)$
end for
return π_θ

Training Algorithm. Algorithm 1 summarizes the complete ACT procedure. The algorithm consists of two phases: data collection, where we construct the ACT dataset by sampling alternatives from the initial policy, and GRPO training, where we optimize the policy using verifiable rewards based on action correctness.

A.2 Implementation details

We use OpenRLHF (Hu et al., 2025) for GRPO training with DeepSpeed ZeRO-3 (Rajbhandari et al., 2020) for memory efficiency. Training uses 16 NVIDIA GH200 GPUs. Table 4 lists the Qwen3 and Olmo 3 training hyperparameters.

A.3 Data statistics

Table 5 summarizes the expert-data statistics across all benchmarks. Each sample corresponds to a single expert state–action pair extracted from a successful trajectory. ACT and IL methods use the same pairs to ensure a fair comparison; RL starts from the corresponding IL checkpoint. Each ACT, IL, and RL stage uses three epochs; IL→IL and IL→RL→RL add three epochs to their respective baseline checkpoints (Section 4.3). For ScienceWorld, we randomly sample 10,240 state-action pairs for training from the full set of expert trajectories.

A.4 Choice of K

We set $K = 1$ in all experiments (Table 4). Since the expert trajectory datasets already provide a sufficient number of training samples (Table 5), a single alternative per state yields enough contrastive

Table 4: Training hyperparameters for Qwen3 and Olmo-3-7B-Instruct.

Hyperparameter	**Value**
Base models	Qwen3-4B, Qwen3-8B, Olmo-3-7B-Instruct
Learning rate	2×10^{-6}
Learning-rate scheduler	Cosine
Warmup ratio	0.1
Batch size	64
Group size (G)	16 (Qwen3-4B); 8 (Qwen3-8B and Olmo 3)
Candidate samples (K)	1
Epochs per stage (ACT / IL / RL)	3
Prompt max length	4,096 tokens
Generation max length	4,096 tokens
Temperature	1.0
Top-p	0.95
KL coefficient (β)	0.0
Optimizer	AdamW with offload
Precision	bfloat16 (BF16)

Table 5: Dataset statistics for all training. Train samples are state-action pairs. *ID*: In-Distribution, *OOD*: Out-of-Distribution.

Benchmark	**Domain**	**Train Pairs**	**Task Types**	**Test Samples**
ALFWorld	Embodied	10,240	6	140 (ID) / 134 (OOD) episodes
WebShop	Web	3,071	N/A	500 episodes
ScienceWorld	Science	10,240	30	149 episodes

pairs for effective training without the need for $K > 1$. In settings where the expert dataset is smaller, increasing K can be used to produce additional contrastive pairs per state to compensate.

A.5 Expert trajectory collection

For ALFWorld, expert trajectories are collected by running the model released by Feng et al. on the ALFWorld training set. For WebShop, expert trajectories come from the official human demonstration data released with the benchmark. For ScienceWorld, expert trajectories come from the official gold trajectories released with the benchmark.

A.6 ACT PROMPT TEMPLATES

The following templates are used during the ACT stage. In each template, `action_1` and `action_2` contain the expert and alternative actions in randomized order.

ALFWorld. The ACT prompt for ALFWorld is shown in Figure 5.

ACT Prompt (ALFWorld)

```
You are an expert agent operating in the ALFWorld Environment.

Task: {task_description}

Prior to this step, you have already taken {n} step(s).
Below are the most recent {k} observations and actions:
{history}

You are now at step {current_step} and your current observation is:
{observation}

Your admissible actions of the current situation are:
{admissible_actions}

You need to decide which action is better for completing the task.

Output format requirements:
1. First, think about which action is better and why.
2. Then, output the chosen action content inside
<action>...</action> tags.
3. You must output the actual action text, NOT 'Action 1' or 'Action
2'.

Candidate Actions for the NEXT step:
Action 1: {action_1}
Action 2: {action_2}

Which action is better as the NEXT step? Think about why, then
output the chosen action inside <action>...</action> tags.
```

Figure 5: The ACT prompt for ALFWorld. The model is presented with the full context followed by two candidate actions and is asked to select the better one with reasoning.

WebShop. The ACT prompt for WebShop is shown in Figure 6.

ACT Prompt (WebShop)

```
You are an expert shopping assistant navigating an e-commerce
website.

Task: {task_description}

Prior to this step, you have already taken {n} step(s).
Below are the most recent {k} observations and actions:
{history}

You are now at step {current_step} and your current observation is:
{observation}

Your admissible actions of the current situation are:
{admissible_actions}

You need to decide which action is better for completing the task.

Output format requirements:
1. First, think about which action is better and why.
2. Then, output the chosen action content inside
<action>...</action> tags.
3. You must output the actual action text, NOT 'Action 1' or 'Action
2'.

Candidate Actions for the NEXT step:
Action 1: {action_1}
Action 2: {action_2}

Which action is better as the NEXT step? Think about why, then
output the chosen action inside <action>...</action> tags.
```

Figure 6: The ACT prompt for WebShop. The model is presented with the full context followed by two candidate actions and is asked to select the better one with reasoning.

ScienceWorld. The ACT prompt for ScienceWorld is shown in Figure 7.

ACT Prompt (ScienceWorld)

```
You are an expert science tutor guiding a student through hands-on
experiments in the ScienceWorld environment.

Task: {task_description}

Prior to this step, you have already taken {n} step(s).
Below are the most recent {k} observations and actions:
{history}

You are now at step {current_step} and your current observation is:
{observation}

Your admissible actions of the current situation are:
{admissible_actions}

You need to decide which action is better for completing the task.

Output format requirements:
1. First, think about which action is better and why.
2. Then, output the chosen action content inside
<action>...</action> tags.
3. You must output the actual action text, NOT 'Action 1' or 'Action
2'.

Candidate Actions for the NEXT step:
Action 1: {action_1}
Action 2: {action_2}

Which action is better as the NEXT step? Think about why, then
output the chosen action inside <action>...</action> tags.
```

Figure 7: The ACT prompt for ScienceWorld. The model is presented with the full context followed by two candidate actions and is asked to select the better one with reasoning.

A.7 Baseline and control implementations

Reflection baselines.

- **Early Experience**: following the self-reflection variant of Zhang et al. (2025), executes both expert and alternative actions in the environment, uses the resulting next-state differences to synthesize reflection targets, and trains the model through next-token prediction to reproduce each reflection together with the expert action. The resulting data is mixed with the standard expert-action data.
- **SAND**: samples $N = 5$ candidate actions at each expert decision point, executes them in the environment to obtain feedback, synthesizes an action-deliberation target from execution-grounded critiques, and applies SFT to the deliberation followed by the expert action (Xia et al., 2025).
- **Agent-R**: rolls out the current policy, identifies its first deviation from the expert trajectory, and constructs revision trajectories by combining the erroneous prefix, a reflection, the expert correction, and the remaining expert suffix; these trajectories are mixed with expert trajectories for SFT (Yuan et al., 2025).

All three baselines optimize next-token cross-entropy rather than a policy-gradient objective.

Same-action-pair controls. Each control uses ACT's expert–alternative pairs, changes only the stage-1 objective, and then follows the same IL→RL downstream pipeline:

- **Full pipeline (SFT-on-choice)**: replaces the ACT-stage RL objective with supervised fine-tuning on the correct action only, without training a rationale.
- **Full pipeline (DPO)**: replaces the ACT-stage RL objective with DPO, treating the expert action as chosen and the alternative as rejected.
- **Full pipeline (no-reasoning)**: keeps the ACT-stage RL objective and reward but removes the reasoning instruction, requiring the model to select an action directly.

A.8 Additional evaluation protocol

ALFWorld. We evaluate each checkpoint on 140 unique valid-seen episodes and 134 unique valid-unseen episodes. The two splits are reported as ID and OOD success rates, respectively. Each episode runs for at most 50 interaction steps.

WebShop. We evaluate each checkpoint on 500 episodes and report the task success rate.

ScienceWorld. We use the official test variations with at most five variations per task, producing 149 interactive episodes. Each episode runs for at most 50 steps. The primary metric is the mean of the highest environment score reached in each episode, on a 0–100 scale, which retains credit for completed subgoals. We additionally record full task completion when the score reaches 100.

Unified CoT protocol. For the prompting baselines, both Qwen3 and Olmo 3 receive the same prompt-level instruction: "Think step by step about which action best advances the task, then give your choice." The CoT and no-CoT prompts differ only by this instruction.

Seeded evaluation. We sample with temperature 0.6 and top-p 0.95 and repeat each evaluation with three runs. All reported values with error bars are the mean $\pm$ standard deviation across these runs.

A.9 Causal rationale intervention protocol

We sample 400 held-out ALFWorld action pairs and compare the original Qwen3-8B checkpoint with the standalone ACT-stage checkpoint under three inference conditions. In the *free* condition, the model generates its reasoning and choice normally. In the *no-rationale* condition, the reasoning block is pre-closed and the model directly selects an action. In the *incorrect-rationale* condition, we inject

a plausible rationale favoring the alternative action as if it had already been generated by the model, then ask the model to complete only the final choice. We report choice accuracy and the fraction of examples whose final choice follows the injected alternative. The intervention changes only the rationale context; the state, candidate actions, checkpoint, and decoding protocol remain fixed.

Table 6: Exact results for the causal rationale intervention in fig. 3 (%).

Condition	Original Qwen3-8B	ACT checkpoint
Free choice accuracy	45.00	**86.50**
No-rationale accuracy	46.25	**80.75**
Incorrect-rationale accuracy	2.25	**50.75**
Follows injected action ↓	81.75	**28.25**

A.10 Cross-size transfer to Qwen3-4B

ACT requires sampling alternative actions to construct its contrastive pairs. We test whether pairs collected from a larger model can be reused by training Qwen3-4B on ALFWorld with ACT pairs collected entirely from Qwen3-8B, without re-collection or adaptation. The 4B experiments use the same method definitions as the Qwen3-8B main experiments: we compare IL with ACT→IL, and IL→RL w/o ACT with the full ACT pipeline, ACT→IL→RL.

Table 7: Cross-size transfer results on Qwen3-4B for ALFWorld. Values are ID and OOD success rates (%). All ACT pairs used by the ACT-based methods were collected from Qwen3-8B.

Method	ID	OOD
Prompt w/o CoT thinking	13.57	8.96
Prompt w/ CoT thinking	50.71	29.85
ACT stage	71.43	62.69
Imitation Learning (IL)	85.00	83.58
Early Experience	88.57	88.06
ACT → IL	88.57	91.04
IL → RL w/o ACT	91.43	88.81
Full ACT pipeline	**92.14**	**91.79**

As shown in Table 7, transferred ACT data improves both downstream pipelines on Qwen3-4B in both ID and OOD settings. Thus, ACT pairs collected from Qwen3-8B remain effective for a smaller model without re-collection or adaptation.

A.11 Full general-reasoning results

Table 8: Full general-reasoning results corresponding to fig. 4. ACT denotes the standalone ACT-stage checkpoint, without downstream IL or RL.

Method	MATH-500	GPQA-Diamond
Prompt w/o CoT thinking	78.60±0.33	42.93±1.09
Prompt w/ CoT thinking	86.93±0.74	51.52±1.89
Imitation Learning	87.00±0.33	44.61±0.95
Early Experience (Self-Reflection)	86.86±0.25	51.85±0.63
SAND	81.00±0.86	52.53±1.65
Agent-R	86.53±0.81	52.86±1.04
IL → RL w/o ACT	87.07±0.77	52.36±1.32
Standalone ACT stage	**87.73±0.19**	**53.37±0.63**

B ADDITIONAL CASE STUDIES: AGENTIC TASK PERFORMANCE

We provide case studies illustrating failure recovery by the full ACT pipeline and characteristic decision-making failures of IL.

B.1 FAILURE RECOVERY THROUGH SELF-CRITIQUE

In ALFWorld, the environment returns "Nothing happens." when an action fails. In the example below, the IL model repeats the same failed action until termination. In contrast, the model trained with the full ACT pipeline diagnoses the root cause through its internal reasoning and issues the correct corrective action. This behavior is consistent with the action-evaluation capability introduced by ACT. Figure 8 shows the two representative traces.

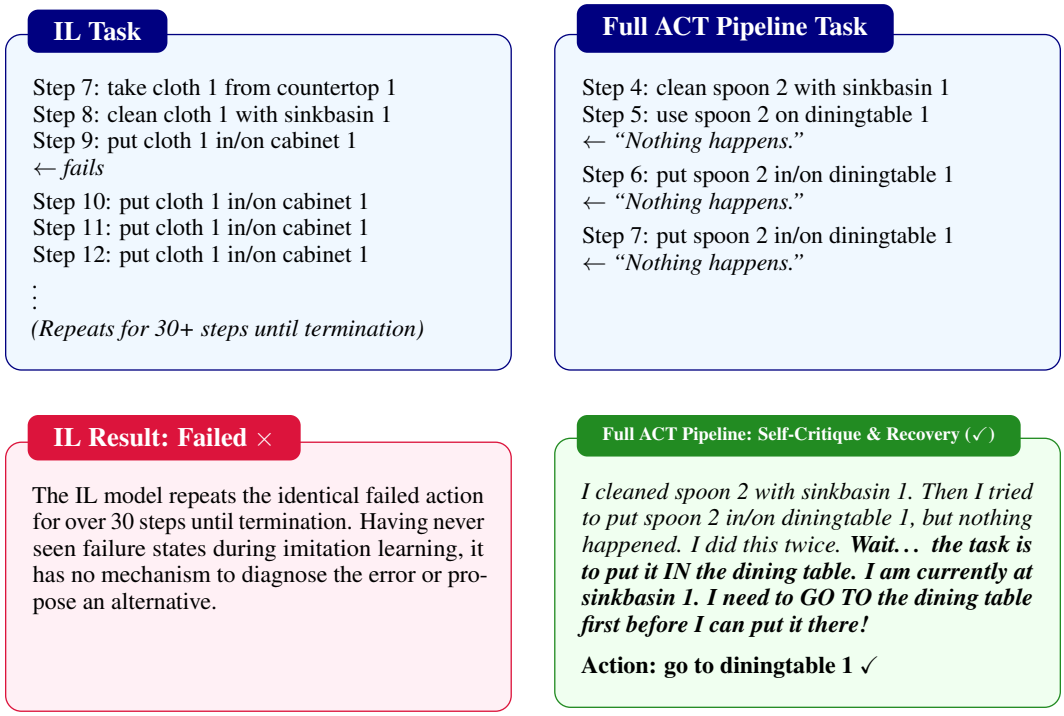

Figure 8: Failure recovery on ALFWorld. **Left:** The IL model enters an infinite loop, repeating a failed action for over 30 steps until termination. **Right:** The model trained with the full ACT pipeline encounters the same type of failure but uses its internal reasoning to diagnose the root cause (wrong location), break the loop, and issue the correct navigation command.

B.2 RIGID EXECUTION WITHOUT STATE AWARENESS

IL trains agents to reproduce demonstrated action patterns. When the environment state deviates from those patterns, an IL model may fail to detect the mismatch and adjust. Figure 9 illustrates this on a WebShop task: the user requests men's shirts with specific attributes *priced below $50*. The IL model follows a rigid sequence (search → click item → select attributes → buy) without checking whether the item satisfies the constraints. It clicks "Buy Now" on a $55 item, violating the price requirement and receiving a score of 0.

This example highlights the type of state-aware action evaluation targeted by ACT: before proceeding, the model should assess whether the current product page satisfies the task constraints and search again when it does not.

WebShop Task

Find me wash cold, machine wash men's shirts with polyester heathers, heathers cotton… and **price lower than 50.00 dollars**.

IL Trajectory (Score: 0.0 ×)

Step 1: search[men's shirts polyester heathers]
Step 2: click[B07XJD…]
Observation: ***Price: $55.00*** *(exceeds $50 budget)*
Step 3: click[black]
Step 4: click[small]
Step 5: click[Buy Now] ← *buys item despite constraint violation*

Figure 9: Rigid execution on WebShop. The IL model follows a scripted sequence (search → click → buy) without checking whether the item satisfies the price constraint ($55 > $50 budget), resulting in a failed purchase.

C ADDITIONAL CASE STUDIES: GENERAL REASONING AFTER AGENTIC TRAINING

We provide case studies analyzing the standalone ACT-stage checkpoint's self-verification behavior and reasoning degradation after IL, based on Qwen3-8B traces from GPQA-Diamond.

C.1 SELF-VERIFICATION BEHAVIOR

On certain difficult problems, the standalone ACT-stage checkpoint exhibits *self-verification* behavior: after performing an initial derivation, the model checks its answer by substituting back into the original equations. Figure 10 shows one such example on a particle physics problem: after deriving the kinetic energies, the standalone ACT-stage checkpoint substitutes each answer option back into the energy conservation equation to verify consistency, systematically eliminating incorrect options. The corresponding original-model trace performs the initial derivation but does not systematically verify the options.

GPQA-Diamond #12: Pion Decay Kinetic Energy

Find the kinetic energy of product particles in $\pi^+ \to \mu^+ + \nu$, where π^+ is stationary. Rest mass of π^+ and μ^+ is 139.6 MeV and 105.7 MeV respectively.
Options: (A) $KE_\mu = 4.12$ MeV, $KE_\nu = 29.8$ MeV (B) $KE_\mu = 7.2$ MeV, $KE_\nu = 32.8$ MeV (C) $KE_\mu = 3.52$ MeV, $KE_\nu = 20.8$ MeV (D) $KE_\mu = 2.84$ MeV, $KE_\nu = 26.8$ MeV

Standalone ACT-Stage Trace: Self-Verification (Correct ✓)

[. . . derives kinetic energies via conservation of energy and momentum. . .]
Verification of each option against energy conservation ($E_{\text{total}} = 139.6$ MeV):
Option C: $KE_\mu = 3.52$ MeV, $KE_\nu = 20.8$ MeV. Then, total energy would be $105.7 + 3.52 = 109.22$ MeV for the muon. Neutrino energy is 20.8 MeV. Total energy: $109.22 + 20.8 = 130.02$ MeV. ***Which is less than 139.6. So that's not possible.***
Option D: $KE_\mu = 2.84$ MeV, $KE_\nu = 26.8$ MeV. Total energy: $105.7 + 2.84 = 108.54$ MeV for muon. Neutrino energy is 26.8 MeV. Total energy is $108.54 + 26.8 = 135.34$ MeV. ***Still less than 139.6.***
So, only option A gives the total energy matching. Therefore, the correct answer is A.

Answer: A ✓

Figure 10: Self-verification behavior observed in the standalone ACT-stage checkpoint on GPQA-Diamond. After deriving the kinetic energies, the checkpoint substitutes each answer option back into the energy conservation equation, eliminating inconsistent options.

C.2 REASONING COLLAPSE AFTER IL

IL on agentic data fine-tunes the model on short, action-heavy expert trajectories that contain minimal extended reasoning. By imitating these behavioral patterns, the model can lose some of its general reasoning capability: the chain-of-thought behavior that the original model possesses may be disrupted by the short-sequence, action-centric distribution of agentic data. We refer to this observed degradation as *reasoning collapse*; one representative trace is shown below.

Unfocused meandering. Even when the IL model produces a reasoning trace, its reasoning can become less focused. Figure 11 shows a high-energy physics problem (GPQA-Diamond #7) about gamma-ray annihilation with CMB photons. The standalone ACT-stage checkpoint produces a focused 10,669-character derivation that sets up the threshold-energy condition and arrives at the correct answer. The IL model generates a 37,924-character trace, 3.5× longer, but wanders through vague recollections and contradictory estimates before guessing incorrectly.

GPQA-Diamond #7: Gamma-Ray CMB Annihilation Threshold

The universe is filled with the Cosmic Microwave Background. Consider the annihilation of high-energy γ-rays with a CMB photon into electron-positron pairs, i.e., $\gamma\gamma \to e^+e^-$. Find the energy of γ-rays whose lifetimes are limited by this process. The average energy of a CMB photon is 10^{-3} eV.

Standalone ACT-Stage Trace (Correct ✓)

For $\gamma\gamma \to e^+e^-$, the total energy must be at least $2m_ec^2$. For a head-on collision, the threshold condition is $E_\gamma \cdot E_{CMB} \geq (m_ec^2)^2$. [...] So $E_\gamma \geq (0.511\,MeV)^2/(10^{-3}\,eV) = 2.61 \times 10^{14}\,eV \approx 2.6\times10^5\,GeV$. The gamma rays with energy just above the threshold would have their lifetimes limited by this process. Therefore, the answer is the threshold energy.

Answer: C (2.6×10^5 **GeV**) ✓ (10,669 chars)

IL Reasoning Trace (Wrong ×)

[...after 37,924 characters of unfocused derivation...]
However, I recall that the energy at which gamma rays can interact with CMB photons to produce e^+e^- pairs is approximately 10^4 GeV. [...] Given that, the answer is likely option B or C. But I'm not certain. [...]
Given the time I've spent and the lack of progress, I'll go with option B: 1.8×10^5 *GeV.*

Answer: B (1.8×10^5 **GeV**) × (37,924 chars)

Figure 11: Reasoning collapse after IL: unfocused meandering. On a high-energy physics threshold problem, the standalone ACT-stage checkpoint produces a focused derivation (10K chars), while IL generates 3.5× as much text (38K chars) yet wanders through contradictory estimates and guesses incorrectly.

This example is consistent with ACT preserving more of the original model's reasoning behavior than IL: ACT optimizes correct action selection via RL rather than imitating short, action-centric targets. We treat the trace as qualitative evidence rather than a general claim about every problem instance.