HEURISTIC-GUIDED BRANCHING-REPLAY GROUP RELATIVE POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training has proven highly effective in improving the reasoning capabilities of large language models (LLMs). However, growing evidence suggests that reinforcement-learning-based post-training encounters a pronounced performance ceiling on long-horizon reasoning tasks. This limitation is particularly acute in multi-turn interactive reasoning, where learning is hindered by sparse terminal rewards, ambiguous trajectory-level credit assignment, and a compounding state-distribution mismatch between fixed replay trajectories and the evolving policy. In this paper, we propose **HBR-GRPO**, or **Heuristic-guided Branching-Replay Group Relative Policy Optimization**, a natural and effective extension of GRPO designed to be broadly applicable to multi-turn interactive reasoning. We adopt a heuristic-search perspective that decomposes trajectory-level training into individual turns and evaluates each action by the reduction it induces in the heuristic distance from the current state to the answer. We then use on-policy branching replay to reconnect these turn-level examples through transitions generated by the current policy, yielding genuine multi-turn interaction trajectories. HBR-GRPO substantially improves the performance of Qwen3-8B across the evaluated tasks. Notably, on one dataset where nearly all models with approximately 8B parameters listed on the clembench leaderboard achieve 0% accuracy, our method attains a nontrivial success rate.

1 INTRODUCTION

Reinforcement-learning post-training has substantially improved the reasoning and decision-making capabilities of large language models (Ouyang et al., 2022), particularly when outcomes are verifiable and interaction horizons are short. However, multi-turn interactive reasoning remains more difficult because a model must follow an action protocol, incorporate environmental feedback, and terminate with a correct solution. Terminal rewards are sparse and assign the same return to decisions of unequal quality, obscuring which intermediate actions actually contribute to progress. A successful trajectory may contain poor intermediate actions, whereas a failed trajectory may still contain useful progress; broadcasting one outcome reward across both obscures this distinction. These problems are especially severe for Group Relative Policy Optimization (GRPO), whose learning signal can vanish when every response in a group fails. Reusing fixed trajectories further introduces state-distribution shift, because deviations by the current policy make later logged states increasingly unlikely to be reached. Since each successor state depends on the preceding interaction history, this mismatch compounds with depth (Yu et al., 2025; Wang et al., 2025).

Representative turn-level methods such as Information Gain-based Policy Optimization (IGPO) and A^{2}TGPO construct denser intermediate rewards, yet still aggregate rewards and perform policy updates over complete sampled trajectories (Wang et al., 2026a; Chen et al., 2026). Credit assignment therefore remains coupled across turns, and existing approaches do not jointly provide externally verifiable state progress, state-conditioned relative optimization, goal-directed termination, and training on states reachable by the current policy.

We propose **HBR-GRPO (Heuristic-guided Branching-Replay Group Relative Policy Optimization)**. Given a heuristic distance $h(s)$ from state s to the success condition, HBR-GRPO rewards an action by its externally verifiable progress, $h(s_t) - h(s_{t+1})$. A trajectory is decomposed into history-conditioned states, and candidate actions sampled from the same state form an

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.

independent GRPO group. This converts one sparse terminal signal into multiple turn-level training examples while avoiding comparisons across states of different difficulty. Since local distance reduction alone may encourage continued search near the goal, we further introduce a finite-horizon **search-value function** that increases the value of answering when successful termination becomes preferable to additional information gathering.

To reduce the mismatch between historical and current-policy trajectories, **On-Policy Branching Replay (OPBR)** samples a new action from a historical prefix, obtains genuine environmental feedback, and stores the resulting reachable successor state for replay. Unlike replaying a fixed offline suffix, each branch follows a transition induced by the current policy. Recursive branching consequently shifts training toward states that the policy can actually visit, making late-turn optimization more relevant at inference time.

We evaluate HBR-GRPO on Wordle and GuessWhat from clembench using the Playpen pipeline (Horst et al., 2025). The method consistently improves accuracy and Clemscore over supervised fine-tuning and conventional outcome-reward post-training, while controlled ablations isolate the effects of its reward and replay components. We further evaluate generalization to multi-hop evidence search on HotpotQA.

Our contributions are:

1. A heuristic, externally verifiable turn-level reward with state-conditioned group-relative optimization.
2. A search-value objective that balances information gathering with timely termination.
3. OPBR, which constructs and reuses current-policy-reachable successor states to mitigate long-horizon distribution shift.

2 Heuristic-guided Branching-Replay GRPO

Figure 1 summarizes the complete training pipeline and the relationship among its supervised initialization, turn-level reward, state-conditioned optimization, and on-policy replay components. Algorithm 1 in Appendix A gives the corresponding pseudocode.

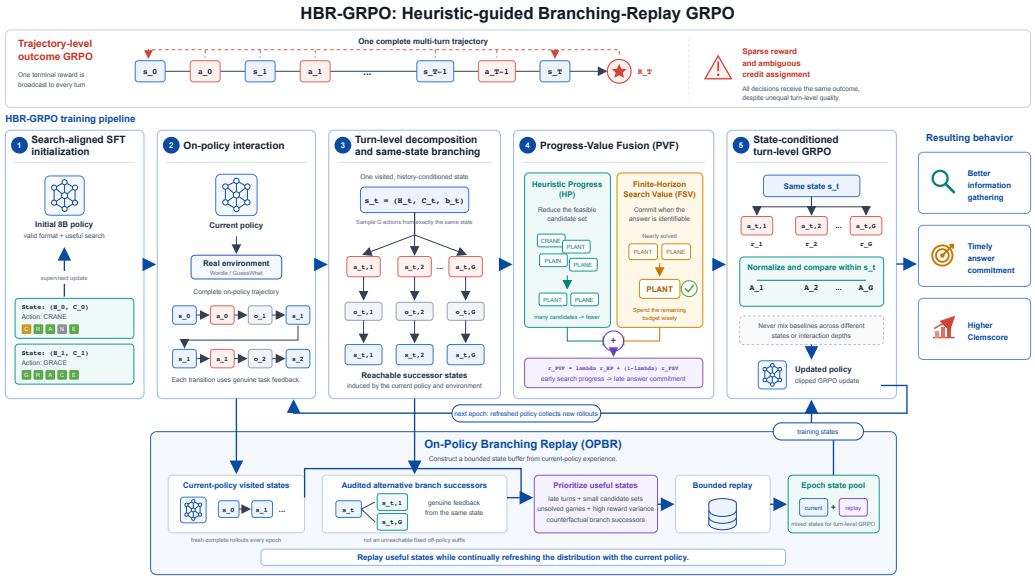


Figure 1: Overview of HBR-GRPO. In contrast to trajectory-level outcome GRPO, our method decomposes on-policy interactions into state-conditioned turn-level groups, combines heuristic progress and finite-horizon search value through Progress-Value Fusion (PVF), and refreshes useful states with OPBR.

2.1 Problem Formulation

We formulate multi-turn interactive reasoning as search over history-conditioned states. At turn t, the state

$$s_t = (\mathcal{H}_t, C_t, b_t) \tag{1}$$

contains the interaction history $\mathcal{H}_t$, the set C_t of hypotheses consistent with that history, and the remaining interaction budget b_t. The policy produces an action a_t, the environment returns deterministic, verifiable feedback $o_{t+1} = O(s_t, a_t)$, and the resulting transition is

$$s_{t+1} = T(s_t, a_t, o_{t+1}). \tag{2}$$

In Wordle, C_t is the set of words consistent with all observed color feedback; in GuessWhat, it is the set of candidates whose labels under a fixed semantic Answerer agree with the observed yes/no history. Further details on both benchmarks are provided in Section 3.1. The Answerer's weights and decoding configuration remain frozen throughout training, and its responses are cached by candidate–question pair to make repeated transitions deterministic. Under a fixed game instance, the reachable states form a directed search graph: each node is a history-conditioned state s_t, and each edge is an action-induced deterministic transition $s_t \xrightarrow{a_t, o_{t+1}} s_{t+1}$. A complete trajectory is one path from the initial state to a terminal state, while alternative actions from the same node define outgoing branches leading to different successor states. HBR-GRPO exploits this structure through four components: a heuristic progress reward, a finite-horizon search-value reward, independent turn-level GRPO, and On-Policy Branching Replay (OPBR).

2.2 Heuristic Progress (HP) Reward

The cardinality of the feasible set gives a direct and externally verifiable distance to the goal. We define the conceptual heuristic

$$h(s_t) = \frac{|C_t|}{|C_0|}, \tag{3}$$

as a normalized heuristic proxy for the distance from the current state to the correct answer. It measures residual ambiguity rather than the exact number of remaining turns: fewer feasible hypotheses indicate that the state is closer to uniquely identifying the answer. Paralleling potential-based reward shaping (Ng et al., 1999), the progress induced by an action is

$$\Delta h_t = h(s_t) - h(s_{t+1}) = \frac{|C_t| - |C_{t+1}|}{|C_0|}. \tag{4}$$

During optimization, we use its state-normalized counterpart

$$r_{\mathrm{HP}}(s_t, a_t) = \mathrm{clip}\left(\frac{|C_t| - |C_{t+1}|}{Z_t}, 0, 1\right), \tag{5}$$

where Z_t is constant within the GRPO group sampled from s_t. We use $Z_t = |C_t|$ for Wordle and $Z_t = \max(1, |C_t| - 1)$ for GuessWhat. GuessWhat normalizes by the number of removable distractors, so a question that isolates the target attains unit HP; Wordle normalizes by all candidates, keeping nonterminal information gain below the unit reward assigned to an exact guess. Malformed or non-partitioning actions (i.e., actions that leave the feasible set unchanged) receive 0, and the state-constant normalization preserves the HP component's within-group ordering.

This objective has an intuitive search interpretation. Consider a Wordle history in which the guess `slate` establishes that `e` is green in the fifth position and that `s`, `l`, `a`, and `t` are absent. A later guess `prone` further shows that `n` is present but not in the fourth position, while `p`, `r`, and `o` are absent. At the next turn, a legal guess partitions the remaining candidates into groups that would produce the same feedback pattern; after feedback is observed, its corresponding group becomes C_{t+1}. A partition without one dominant group therefore tends to leave fewer candidates and yields greater expected reduction. In GuessWhat, each question produces two groups, yes and no, and is most informative when the split is nearly even. Thus, optimizing Eq. (5) teaches the policy to use accumulated evidence when selecting informative actions.

2.3 FINITE-HORIZON SEARCH VALUE (FSV) AND PROGRESS-VALUE FUSION (PVF)

Heuristic Progress measures immediate information gain but does not represent the value of terminating. Once $|C_t|$ is small, a policy trained only with r_{HP} may continue gathering information even when the answer is already identifiable. On the frozen seed-13 diagnostics, HP-only trajectories frequently reach singleton candidate states without successful termination; full curves and counts are reported in Appendix E.1. This gap shows that local progress is not, by itself, a sufficient training objective. We therefore introduce a finite-horizon search value. Let $\mathcal{A}_{\mathrm{ref}}(s)$ be a fixed reference action set containing information-gathering actions and admissible direct answers, and let $\gamma \in (0, 1]$ be a search discount. For any action with possible observable outcomes o, we define

$$Q_b(s,a) = \sum_o p(o \mid s,a)\big[\mathbb{1}\{o = \text{success}\} + \gamma\, \mathbb{1}\{o \neq \text{success}\} V_{b-1}(T(s,a,o))\big], \tag{6}$$

$$Q_{\max,b}(s) \equiv V_b(s) = \max_{a \in \mathcal{A}_{\mathrm{ref}}(s)} Q_b(s,a), \qquad V_0(s) = 0, \tag{7}$$

where b is the remaining turn budget. The branch probabilities are induced by the current feasible set. Since C_t records consistency with the observed history but provides no relative likelihoods among the surviving candidates, we assign equal mass to each of them; consequently, $p(o \mid s, a)$ is the fraction that produce outcome o. Wordle uses its exact feedback matrix, with bounded exact search for small feasible sets. For GuessWhat, the numerator is evaluated on the policy-generated question itself: the fixed Answerer labels every candidate in C_t as yes or no, thereby inducing the question's partition and its branch probabilities. To compute $Q_{\max,b}(s)$, we enumerate the reference actions in $\mathcal{A}_{\mathrm{ref}}(s)$ from frozen libraries of high-value Wordle guesses and semantically verified GuessWhat questions; admissible direct answers are included in the same reference maximization. We normalize the generated action value against this fixed reference maximum:

$$r_{\mathrm{FSV}}(s_t,a_t) = \begin{cases} 0, & Q_{\max,b_t}(s_t) = 0, \\ \mathrm{clip}\left(\dfrac{Q_{b_t}(s_t,a_t)}{Q_{\max,b_t}(s_t)}, 0, 1\right), & Q_{\max,b_t}(s_t) > 0, \end{cases} \in [0,1]. \tag{8}$$

When $Q_{\max,b_t}(s_t) = 0$, no reference action can reach success within the remaining budget, so FSV is set to 0. Occasional exceedances of the finite reference maximum are clipped to preserve a common reward scale with HP. This value accounts for both eventual success and the cost of consuming another turn: when the answer is uniquely identified, answering retains terminal value while further information gathering does not. FSV therefore supplies the closing pressure absent from HP. Empirically, clipping is activated only rarely during training; Appendix F reports the counts and rates.

Neither component is sufficient across the full search horizon: HP provides a dense and stable signal in large candidate spaces, while FSV becomes most informative near the goal. We combine them through PVF:

$$r_{\mathrm{PVF}}(s_t,a_t) = \lambda(s_t)\, r_{\mathrm{HP}}(s_t,a_t) + \big(1 - \lambda(s_t)\big)\, r_{\mathrm{FSV}}(s_t,a_t), \tag{9}$$

$$\lambda(s_t) = w(|C_t|)\, \mathrm{clip}\left(\frac{b_t - 1}{\tau}, 0, 1\right). \tag{10}$$

Here, $w(|C_t|)$ is a monotone binned weight that is larger for broad candidate sets, and τ scales turn urgency for each environment. Consequently, early states with many candidates emphasize immediate reduction, whereas small candidate sets and nearly exhausted budgets shift weight toward finite-horizon value and direct termination.

2.4 INDEPENDENT TURN-LEVEL GRPO

After collecting complete interaction trajectories, we decompose each trajectory into its visited states rather than collapsing all turn rewards into one trajectory-level return (as in IGPO and A^{2}TGPO (Wang et al., 2026a; Chen et al., 2026)). Each state retains the full preceding action-feedback history, but it becomes an independent training example. For a state s_t, the current policy samples a group of

G candidate actions $\{a_{t,i}\}_{i=1}^{G}$, all conditioned on exactly the same history. Their state-conditioned relative advantages are

$$\hat{A}_{t,i} = \frac{r_{\mathrm{PVF}}(s_t, a_{t,i}) - \mu_t}{\sigma_t + \epsilon}, \quad \mu_t = \frac{1}{G}\sum_{j=1}^{G} r_{\mathrm{PVF}}(s_t, a_{t,j}), \tag{11}$$

where σ_t is the within-state reward standard deviation. The policy is optimized with the clipped group-relative objective

$$\mathcal{L}_{\mathrm{turn}} = -\mathbb{E}_{t,i}\left[\min\left(\rho_{t,i}\hat{A}_{t,i}, \mathrm{clip}(\rho_{t,i}, 1-\varepsilon, 1+\varepsilon)\hat{A}_{t,i}\right)\right], \tag{12}$$

where $\rho_{t,i} = \pi_\theta(a_{t,i} \mid s_t)/\pi_{\mathrm{old}}(a_{t,i} \mid s_t)$. Here, π_{old} is the pre-update behavior policy that generated the current action group; it is frozen for that update and refreshed after each new rollout batch. Crucially, rewards from different states or interaction depths are never included in the same GRPO group-wise reward normalization; μ_t and σ_t are computed only over actions sampled from the same state. This differs from assigning a weighted sum of turn scores back to an entire trajectory: Eq. (11) directly compares alternative actions under the same evidence and remaining budget. It therefore converts one sparse trajectory-level outcome into multiple updates with localized credit while avoiding the confounding effect of comparing intrinsically easy and difficult states.

2.5 On-Policy Branching Replay

Reusing stored transitions, as in classical experience replay (Lin, 1992), creates a distribution-shift problem here. If the updated policy chooses an action different from the behavior trajectory, the resulting feedback changes every later state; suffixes extracted from the historical trajectory may then be unreachable under the current policy. The mismatch compounds with interaction depth, making late-turn offline states progressively less useful.

OPBR refreshes the state distribution at every training epoch. Starting from the current policy, we execute complete interactions against the actual task transition function and record every visited pre-action state. Each state becomes a branch point from which GRPO samples a fresh action group. In GuessWhat, each current-policy question is applied directly through the same fixed Answerer, irrespective of whether it appears in the frozen reference library; the resulting yes/no label supplies genuine environment feedback and updates the feasible set. We additionally select a high-value alternative question from the same audited, frozen reference library used for $Q_{\max}$, excluding the current policy action. We apply this question through the fixed Answerer and materialize its successor only when it yields a reliable, non-degenerate partition. These successors are adjacent to states visited by the current policy rather than continuations copied from an incompatible historical suffix.

The epoch buffer combines newly collected states with a bounded fraction of states from previous epochs. Replay is prioritized toward later turns, small candidate sets, unsuccessful trajectories, high-reward-variance states, and counterfactual branch successors. Training then applies the same state-conditioned PVF-GRPO objective to the selected buffer. This procedure preserves useful difficult states while continually moving the training distribution toward states reachable by the improving policy.

Since OPBR bootstraps from its own rollouts, it requires a sufficiently competent starting policy. A weak initializer that frequently violates the action format or produces degenerate actions yields few valid transitions and a poor replay distribution; replay alone cannot repair the absence of usable search behavior. We therefore initialize the main method with search-aligned supervised fine-tuning (S-SFT). For Wordle, demonstrations are constructed from exact feedback transitions and information-gain scores over legal, non-repeated guesses, allowing both candidate answers and diagnostic probes to be selected according to their search value. For GuessWhat, the initial action pool is filtered for protocol compliance, candidate-name leakage, non-degenerate semantic partitions, and finite-horizon planner coverage. S-SFT supplies valid and informative behavior before PVF and OPBR are applied to current-policy-reachable states.

3 EXPERIMENTS

3.1 DATASETS

We evaluate HBR-GRPO on WORDLE and GUESSWHAT, two multi-turn dialogue games distributed with clembench (Chalamalasetti et al., 2023), a framework for evaluating language models as situated conversational agents through automatically scored gameplay. The task instances and official evaluation rules follow clembench, while the reinforcement-learning interaction environments are instantiated with Playpen (Horst et al., 2025). Playpen exposes each game as a stateful learning environment in which a Game Master enforces the protocol, executes task transitions, returns environment feedback, and records automatically verifiable outcomes.

Wordle. The model must identify a hidden five-letter English word within at most six guesses. After each valid guess, the environment returns letter-level feedback indicating whether each letter is correct and in the correct position, present in the target but misplaced, or absent from the target. Solving the task therefore requires the model to maintain constraints across turns, use feedback to reduce the feasible search space, and produce a valid final answer under a strict response protocol. An episode is aborted if the model repeatedly violates the required format or game rules.

GuessWhat. Each instance presents eight candidate words, one of which is hidden as the target. The model asks one semantic yes/no question per turn to eliminate incompatible candidates and has only one opportunity to submit its final guess. The instances span several levels of semantic relatedness and include both concrete and abstract concepts. The task jointly tests protocol compliance, information-seeking question generation, belief-state refinement from natural-language feedback, and the decision of when to stop asking questions and commit to an answer.

3.2 METRICS

We report **Accuracy (ACC)**, **Quality**, **Played**, and **Clemscore** following clembench. ACC is the fraction of all episodes solved, Played is the fraction not aborted by protocol or parsing failures, and Quality measures solution efficiency over played episodes. Clemscore combines Quality with protocol adherence:

$$\text{Clemscore} = \text{Quality} \times \frac{\text{Played}}{100}. \tag{13}$$

We therefore treat Clemscore as the primary metric and report ACC, Quality, and Played as complementary diagnostics. Appendix B provides the complete task-specific scoring definitions.

3.3 DATASET SPLITS

Wordle uses mutually disjoint target sets for search-aligned SFT, RL, development, and testing; checkpoint selection uses 124 development targets, followed by one evaluation on 168 clean-test targets. GuessWhat is stratified across its six experiment types into 120 training, 30 development, and 60 test instances, with disjoint targets and complete candidate sets across splits. In both tasks, development data are used only for checkpoint selection and test data are evaluated only after selection. Appendix C provides the complete construction procedure, turn-level state counts, replay composition, hyperparameters, and runtime configuration.

3.4 MAIN RESULTS

We post-trained Qwen3-8B (Yang et al., 2025) with HBR-GRPO and compared it with SFT and SFT-GRPO baselines across six open-weight model families: Qwen3-8B, Gemma-2-9B-IT (Gemma Team et al., 2024), Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), Falcon-7B-Instruct (Almazrouei et al., 2023), GLM-4-9B-Chat (GLM Team et al., 2024), and DeepSeek-LLM-7B-Chat (DeepSeek-AI et al., 2024). Except for the four models marked with †, all baseline post-training experiments follow the Playpen setup (Horst et al., 2025), including the game environments, interaction protocol, supervised initialization, GRPO baseline, parsers, scorers, and held-out evaluation procedure. The four marked rows are transcribed from the clembench v2.0 leaderboard.

At each training state, one GRPO rollout group contains eight independently sampled responses. HBR-GRPO is evaluated over five random seeds, and Table 1 reports the corresponding mean and

sample standard deviation; Quality is computed separately for each seed before aggregation. Full optimization hyperparameters, model-specific SFT settings, software versions, and hardware configurations are provided in Appendix C; per-seed development scores and checkpoint selections are reported in Appendix D.

Table 1: Main results on Wordle and GuessWhat. The HBR-GRPO row reports the five-seed mean ± sample standard deviation; the four rows marked with † are copied directly from the clembench leaderboard. Clemscore, shown in bold for both tasks, is the primary metric.

	Wordle				GuessWhat			
Model and training method	ACC (%)	Quality	Played (%)	**Clemscore**	ACC (%)	Quality	Played (%)	**Clemscore**
Qwen3-8B (SFT)	1.79	3.92	30.36	**1.19**	13.33	15.09	88.33	**13.33**
Qwen3-8B (SFT-GRPO)	1.79	3.92	30.36	**1.19**	18.33	20.00	91.67	**18.33**
Gemma-2-9B-IT (SFT)	7.14	5.92	58.33	**3.45**	48.33	14.12	98.33	**13.89**
Gemma-2-9B-IT (SFT-GRPO)	7.74	6.76	60.71	**4.11**	46.67	16.36	91.67	**15.00**
Mistral-7B-Instruct-v0.3 (SFT)	0.00	0.00	70.83	**0.00**	0.00	0.00	100.00	**0.00**
Mistral-7B-Instruct-v0.3 (SFT-GRPO)	0.00	0.00	71.43	**0.00**	21.67	8.89	100.00	**8.89**
Falcon-7B-Instruct (SFT)	0.00	0.00	67.86	**0.00**	0.00	0.00	0.00	**0.00**
Falcon-7B-Instruct (SFT-GRPO)	0.00	0.00	76.79	**0.00**	0.00	0.00	0.00	**0.00**
GLM-4-9B-Chat (SFT)	0.60	0.35	83.93	**0.30**	26.67	24.14	96.67	**23.33**
GLM-4-9B-Chat (SFT-GRPO)	0.60	0.35	83.93	**0.30**	41.67	27.01	96.67	**26.11**
DeepSeek-LLM-7B-Chat (SFT)	0.00	0.00	56.55	**0.00**	5.00	5.56	90.00	**5.00**
DeepSeek-LLM-7B-Chat (SFT-GRPO)	0.00	0.00	56.55	**0.00**	18.33	12.28	95.00	**11.67**
OLMo-2-1124-7B-Instruct†	–	0.00	0.00	**0.00**	–	20.34	98.33	**20.00**
Meta-Llama-3.1-8B-Instruct†	–	2.00	50.00	**1.00**	–	13.33	91.67	**12.22**
Llama-3.3-70B-Instruct†	–	2.39	70.00	**1.67**	–	37.74	88.33	**33.34**
Llama-4-Maverick†	–	7.14	46.67	**3.33**	–	52.20	88.33	**46.11**
HBR-GRPO (ours)	**15.71** ±2.75	**11.14** ±1.06	**76.31** ±6.57	**8.50** ±1.10	**48.33** ±8.08	**48.33** ±8.08	**100.00** ±0.00	**48.33** ±8.08

† The clembench v2.0 leaderboard reports Played and Quality, but not ACC, for individual games; unavailable ACC values are therefore marked "–". We compute the displayed Clemscore as $\mathrm{Quality} \times \mathrm{Played}/100$ using the leaderboard values.

3.5 Overall Performance

Among the publicly available clembench v2.0 leaderboard results for models with fewer than 25B parameters, HBR-GRPO achieves the strongest reported Clemscore on both Wordle and GuessWhat among the entries considered in Table 1. Moreover, the resulting 8B model is competitive with substantially larger systems: it exceeds the reported scores of Llama-3.3-70B-Instruct and Llama-4-Maverick on both tasks. More broadly, these results place HBR-GRPO in the performance range of some much larger or proprietary systems represented on the public leaderboard.

The gains over the Qwen3-8B SFT baseline are substantial in the primary metric. On Wordle, HBR-GRPO raises Clemscore from 1.19 to 8.50, an absolute improvement of 7.31 points and a $7.14\times$ increase. On GuessWhat, Clemscore rises from 13.33 to 48.33, an absolute gain of 35.00 points and a $3.63\times$ increase. In contrast, conventional trajectory-level outcome-based GRPO provides much smaller gains: the Qwen3-8B SFT-GRPO baseline remains unchanged at 1.19 on Wordle and reaches only 18.33 on GuessWhat, a 5.00-point improvement over SFT. The same pattern appears across several other model families in Table 1, where outcome-based GRPO is often limited or produces no improvement at all. We evaluate each epoch checkpoint of these conventional GRPO runs on the held-out development split using a frozen selection rule. The selected checkpoints frequently occur in early epochs, including epoch 1 in several runs, because later checkpoints yield little or no development-set improvement; this behavior is consistent with the limited gains observed in the final results. For transparency, Appendix D reports complete development-set checkpoint scores and selections for all five HBR-GRPO seeds.

HBR-GRPO also improves protocol compliance, although not uniformly across tasks or comparison models. Relative to Qwen3-8B SFT, Played increases from 30.36% to 76.31% on Wordle and from 88.33% to 100.00% on GuessWhat, corresponding to gains of 45.95 and 11.67 percentage points, respectively. These improvements do not uniformly dominate the baselines: the Wordle Played score remains below several comparison models, while the GuessWhat score reaches the ceiling but ties existing results. The Clemscore gains therefore reflect a combination of stronger task-solving quality and higher rates of valid play, with their relative contributions differing across the two tasks.

3.6 Ablation Study

Table 2: Ablation results with search-aligned turn-level SFT (S-SFT) initialization (seed 13). Each entry reports Wordle / GuessWhat.

OPBR setting	Reward	ACC (%)	Quality	Played (%)	**Clemscore**
SFT-only	–	7.7380 / 35.0000	6.2166 / 35.0000	50.5952 / 100.0000	**3.1453 / 35.0000**
w/o OPBR	HP	10.7143 / 48.3333	11.6279 / 48.3333	51.1905 / 100.0000	**5.9524 / 48.3333**
	FSV	10.7143 / 45.0000	11.2087 / 45.0000	54.1667 / 100.0000	**6.0714 / 45.0000**
	PVF	8.9286 / 46.6667	8.4810 / 46.6667	47.0238 / 100.0000	**3.9881 / 46.6667**
w/ OPBR	HP	11.9048 / 46.6667	12.0931 / 46.6667	51.1905 / 100.0000	**6.1905 / 46.6667**
	FSV	11.3095 / 41.6667	14.6574 / 41.6667	43.4524 / 100.0000	**6.3690 / 41.6667**
w/ OPBR	**PVF**	**12.5000 / 51.6667**	**10.0870 / 51.6667**	**68.4524 / 100.0000**	**6.9048 / 51.6667**

Table 3: Ablation results with Playpen's original trajectory-level SFT initialization (seed 13). Each entry reports Wordle / GuessWhat.

OPBR setting	Reward	ACC (%)	Quality	Played (%)	**Clemscore**
SFT-only	–	1.7857 / 13.3333	3.9217 / 15.0943	30.3571 / 88.3333	**1.1905 / 13.3333**
w/o OPBR	HP	4.1667 / 20.0000	5.1389 / 22.6415	42.8571 / 88.3333	**2.2024 / 20.0000**
	FSV	2.9762 / 18.3333	4.1177 / 20.7547	40.4762 / 88.3333	**1.6667 / 18.3333**
	PVF	2.9762 / 18.3333	3.2051 / 20.7547	46.4286 / 88.3333	**1.4881 / 18.3333**
w/ OPBR	HP	4.7619 / 21.6667	5.8431 / 25.0000	42.8571 / 86.6667	**2.5042 / 21.6667**
	FSV	2.9762 / 18.3333	6.2501 / 20.7547	42.8571 / 88.3333	**2.6786 / 18.3333**
	PVF	5.9523 / 23.3333	7.3128 / 26.4151	43.4524 / 88.3333	**3.1776 / 23.3333**

The ablations reveal a division of labor rather than uniform additivity. HP addresses sparse terminal supervision through verifiable feasible-set reduction; its robustness across initializers, particularly on semantic GuessWhat partitions, shows that local information gain supplies credit before success is frequent. FSV targets the complementary closing failure: reducing ambiguity without committing before budget exhaustion. The combination of stronger Wordle Quality and less stable Played indicates sharper termination but also greater brittleness in isolation. PVF therefore implements a phase-dependent handoff from exploration to closing.

OPBR solves the distribution problem required for that handoff. Its inconsistent effect with HP or FSV alone rules out a generic "more states" explanation, whereas PVF+OPBR is consistently strongest. As the policy changes, fixed suffixes become stale and underrepresent late, low-ambiguity states where FSV should dominate. Regenerating reachable histories aligns replay with current-policy search: PVF specifies what to optimize, while OPBR supplies where to optimize it.

Search-aligned S-SFT addresses replay cold start by producing legal, informative branches; the weaker trajectory initializer remains recoverable but yields poorer states, showing that replay cannot manufacture signal from protocol failures. With the same S-SFT, standard GRPO and IGPO (Wang et al., 2026a) remain behind, so initialization is insufficient. HBR-GRPO's same-state groups and decision-level credit avoid cross-state difficulty confounding and trajectory-level reward dilution. Although the full comparison changes reward, grouping, and replay jointly, the component ablations support their complementarity.

Table 4: Comparison of RL algorithms after turn-level SFT initialization (seed 13). Each entry reports Wordle / GuessWhat.

Method	ACC (%)	Quality	Played (%)	**Clemscore**
Standard GRPO	8.9286 / 35.0000	6.8269 / 35.0000	61.9048 / 100.0000	**4.2262 / 35.0000**
IGPO	9.5238 / 41.6667	7.5258 / 41.6667	57.7381 / 100.0000	**4.3452 / 41.6667**
HBR-GRPO (ours)	**12.5000 / 51.6667**	**10.0870 / 51.6667**	**68.4524 / 100.0000**	**6.9048 / 51.6667**

3.7 Generalization to Multi-Hop Search

To test transfer beyond language-based reasoning games, we evaluate HBR-GRPO on HotpotQA (Yang et al., 2018), which requires repeated retrieval and evidence selection across multiple Wikipedia documents before answering. Complete experimental details are provided in Appendix G.

Table 5: Results on the held-out HotpotQA FullWiki split (3,680 examples; seed 13). Metrics are percentages; EM denotes exact match; best results are bold.

Method	Answer EM	Answer F1	Supporting Fact EM	Supporting Fact F1	Joint EM	Joint F1
SFT-only	35.87	46.13	14.74	40.43	10.79	28.01
GRPO	37.27	49.03	14.96	42.11	11.49	29.88
IGPO (Wang et al., 2026a)	40.79	51.89	17.28	46.08	12.74	32.65
GiGPO (Feng et al., 2025)	39.56	51.42	16.71	44.79	12.70	32.35
Search-R1-base (Jin et al., 2025)	39.14	51.65	15.09	44.21	12.22	32.28
HBR-GRPO (ours)	**41.96**	**52.60**	**20.24**	**48.91**	**15.43**	**35.15**

HBR-GRPO achieves the best result on all six metrics in Table 5, reaching 35.15 Joint F1 versus 32.65 for the strongest baseline and demonstrating transfer to multi-hop corpus search.

3.8 Failure Analysis

Residual failures primarily involve delayed answer commitment after the state identifies a solution. GuessWhat also suffers from poorly partitioning questions that consume budget without sufficient information gain. Appendix E provides representative trajectories and detailed analysis.

4 Related Work

GRPO replaces the learned critic in Proximal Policy Optimization (PPO) with within-group comparisons and improves mathematical reasoning (Shao et al., 2024); DeepSeek-R1 further shows that large-scale RL can elicit adaptive reasoning without annotated trajectories (Guo et al., 2025). Dr. GRPO removes an objective bias favoring longer responses (Liu et al., 2025), while DAPO adds decoupled clipping, dynamic sampling, and token-level losses (Yu et al., 2025). Process supervision provides denser intermediate feedback (Lightman et al., 2024), but this line largely targets static solutions.

Static benchmarks such as GSM8K and MATH expose verifiable final answers (Cobbe et al., 2021; Hendrycks et al., 2021), whereas interactive settings additionally require protocol adherence and environment feedback, as illustrated by ReAct, WebShop, clembench, and Playpen (Yao et al., 2023; 2022; Chalamalasetti et al., 2023; Horst et al., 2025). Recent agent-RL methods study long-horizon instability (RAGEN), confidence progress (IGPO), stage-specific shopping rewards (Shop-R1), and simulator-derived emotion rewards (RLVER) (Wang et al., 2025; 2026a; Zhang et al., 2026; Wang et al., 2026b). HBR-GRPO instead uses externally verifiable state progress, same-state action comparisons, finite-horizon closing pressure, and current-policy branching replay, avoiding sole reliance on terminal outcomes, internal confidence, learned reward proxies, or fixed trajectories.

5 Conclusion, Limitations, and Future Work

We introduced HBR-GRPO for multi-turn interactive reasoning. Its combination of heuristic progress, finite-horizon search value, state-conditioned optimization, and on-policy branching replay provides dense credit beyond sparse trajectory outcomes. On Wordle and GuessWhat, Qwen3-8B outperforms the considered small-model clembench entries and remains competitive with substantially larger systems. Its lead over the evaluated HotpotQA baselines across all six metrics further demonstrates transfer beyond games.

HBR-GRPO currently requires externally auditable progress signals, such as candidate-set reduction or labeled supporting evidence. Extending it beyond such environments will require reliable learned or induced heuristics.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing, code implementation and debugging, experiment orchestration, and the inspection and summarization of experimental logs and outputs. They were also used to provide feedback on the presentation of the methodology and experimental setup. All research decisions, experimental configurations, and reported claims were reviewed and approved by the authors. AI-assisted code was tested against the stated protocols, and quantitative results were verified against the original evaluation artifacts. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work studies reinforcement-learning methods on publicly available language-game and question-answering benchmarks and does not involve human participants, newly collected personal data, or decisions about individuals. The evaluated language models may nevertheless reproduce biases or generate inaccurate or inappropriate content inherited from their pretraining data. Our experiments are confined to automatically verifiable research environments and should not be interpreted as validating deployment in safety-critical or user-facing settings. We report the evaluated tasks, data splits, and scoring procedures to support transparent interpretation of the results.

REPRODUCIBILITY STATEMENT

Section 3 specifies the datasets, metrics, data splits, baselines, and evaluation protocol. Appendix A gives the complete HBR-GRPO procedure, and Appendices B–D provide the metric definitions, dataset construction, prompts, reward and search settings, training configuration, software and hardware environment, and checkpoint-selection procedure. Appendix G separately documents the frozen retrieval environment, prompts, rewards, initialization, hyperparameters, and evaluation setup for HotpotQA. All reported experiments use fixed data splits and explicitly stated random seeds, and test sets are evaluated only after checkpoint selection.

REFERENCES

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*, 2023. URL https://arxiv.org/abs/2311.16867.

Kranti Chalamalasetti, Jana Götze, Sherzod Hakimov, Brielen Madureira, Philipp Sadler, and David Schlangen. clembench: Using game play to evaluate chat-optimized language models as conversational agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 11174–11219, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.689. URL https://aclanthology.org/2023.emnlp-main.689/.

Dingwei Chen, Zefang Zong, Zhipeng Ma, Leo Luo, Yang Li, Chengming Li, Peng Chen, and Jie Jiang. A^2TGPO: Agentic turn-group policy optimization with adaptive turn-level clipping. *arXiv preprint arXiv:2605.06200*, 2026. URL https://arxiv.org/abs/2605.06200.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI, Xiao Bi, et al. DeepSeek LLM: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024. URL https://arxiv.org/abs/2401.02954.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing*

Systems, volume 36, pp. 10088–10115, 2023. doi: 10.52202/075280-0441. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025.

Gemma Team et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. URL https://arxiv.org/abs/2408.00118.

GLM Team, Aohan Zeng, et al. ChatGLM: A family of large language models from GLM-130B to GLM-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024. URL https://arxiv.org/abs/2406.12793.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Nature*, 645: 633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Nicola Horst, Davide Mazzaccara, Antonia Schmidt, Michael Sullivan, Filippo Momentè, Luca Franceschetti, Philipp Sadler, Sherzod Hakimov, Alberto Testoni, Raffaella Bernardi, Raquel Fernández, Alexander Koller, Oliver Lemon, David Schlangen, Mario Giulianelli, and Alessandro Suglia. Playpen: An environment for exploring learning from dialogue game feedback. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 29854–29891, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1517. URL https://aclanthology.org/2025.emnlp-main.1517/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. URL https://arxiv.org/abs/2310.06825.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Conference on Language Modeling*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992. doi: 10.1007/BF00992699. URL https://doi.org/10.1007/BF00992699.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287. Morgan Kaufmann, 1999. URL https://ai.stanford.edu/~ang/papers/icml99-shaping.pdf.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract.html.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=qkWP6phrvZ.

Peisong Wang, Ruotian Ma, Bang Zhang, Xingyu Chen, Zhiwei He, Kang Luo, Qingsong Lv, Qingxuan Jiang, Zheng Xie, Shanyi Wang, Cixing Li, Yuan Li, Fanghua Ye, Jian Li, Yifan Yang, Jia Li, Zhaopeng Tu, and Xiaolong Li. RLVER: Reinforcement learning with verifiable emotion rewards for empathetic agents. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=P7wBg0vPTh.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025. URL https://arxiv.org/abs/2504.20073.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pp. 20744–20757, 2022. doi: 10.52202/068431-1508. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://openreview.net/forum?id=2a36EMSSTp.

Yimeng Zhang, Tian Wang, Jiri Gesi, Ziyi Wang, Yuxuan Lu, Jiacheng Lin, Sinong Zhan, Vianne Gao, Ruochen Jiao, Junze Liu, Kun Qian, Yuxin Tang, Ran Xue, Houyu Zhang, Qingjun Cui, Yufan Guo, and Dakuo Wang. Shop-r1: Rewarding LLMs to simulate human behavior in online shopping via reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fkIePp9YEO.

A HBR-GRPO ALGORITHM

Algorithm 1 HBR-GRPO training with state-conditioned PVF and OPBR

Require: Initial policy π_{init}; search-aligned demonstrations $\mathcal{D}_{\text{SFT}}$; fixed-state pool $\mathcal{D}_{\text{fix}}$; initial-state distribution p_0; environment $\mathcal{E}$ with transition T; heuristic h; HP normalizer Z; reference action sets $\mathcal{A}_{\text{ref}}$; fixed-state and OPBR epochs $E_{\text{fix}}, E_{\text{OPBR}}$; group size G; replay fraction η; discount γ; normalization constant ϵ; clipping parameter ε

$\pi_\theta \leftarrow \text{S-SFT}(\pi_{\text{init}}, \mathcal{D}_{\text{SFT}})$; $\mathcal{R} \leftarrow \varnothing$
for $e = 1, \ldots, E_{\text{fix}} + E_{\text{OPBR}}$ **do**
  $\pi_{\text{old}} \leftarrow \pi_\theta$
  if $e \le E_{\text{fix}}$ **then**
    $\mathcal{B}_e \leftarrow \text{Sample}(\mathcal{D}_{\text{fix}})$
  else
    $\Xi_e \leftarrow \text{Rollout}(\pi_{\text{old}}, p_0, \mathcal{E})$ ▷ complete current-policy interactions
    $\mathcal{S}_{\text{cur}} \leftarrow \text{Decompose}(\Xi_e)$ ▷ all visited pre-action states
    $\mathcal{S}_{\text{br}} \leftarrow \text{AuditedBranches}(\mathcal{S}_{\text{cur}}, \mathcal{E})$
    $\mathcal{B}_e \leftarrow \text{PrioritizedMix}(\mathcal{S}_{\text{cur}} \cup \mathcal{S}_{\text{br}}, \mathcal{R}, \eta)$
    $\mathcal{R} \leftarrow \text{BoundedUpdate}(\mathcal{R}, \mathcal{S}_{\text{cur}} \cup \mathcal{S}_{\text{br}})$
  end if
  for each minibatch $\mathcal{M} \subset \mathcal{B}_e$ **do**
    for each state $s = (\mathcal{H}, C, b) \in \mathcal{M}$ **do**
      Sample $\{a_i\}_{i=1}^{G} \sim \pi_{\text{old}}(\cdot \mid s)$
      $Q_{\max,b}(s) \leftarrow \max_{a \in \mathcal{A}_{\text{ref}}(s)} Q_b(s, a)$
      for $i = 1, \ldots, G$ **do**
        $o_i \leftarrow \mathcal{E}(s, a_i)$; $s_i' \leftarrow T(s, a_i, o_i)$
        $r_{\text{HP},i} \leftarrow \text{clip}((|C| - |C_i'|)/Z(s), 0, 1)$
        if $Q_{\max,b}(s) = 0$ **then**
          $r_{\text{FSV},i} \leftarrow 0$
        else
          $r_{\text{FSV},i} \leftarrow \text{clip}(Q_b(s, a_i)/Q_{\max,b}(s), 0, 1)$
        end if
        $\lambda_s \leftarrow w(|C|)\,\text{clip}((b-1)/\tau, 0, 1)$
        $r_i \leftarrow \lambda_s r_{\text{HP},i} + (1 - \lambda_s) r_{\text{FSV},i}$
      end for
      $\widehat{A}_i \leftarrow (r_i - \mu_s)/(\sigma_s + \epsilon)$ for $i = 1, \ldots, G$
      Assign $\widehat{A}_i$ to the generated tokens of a_i; store $(s, a_i, \widehat{A}_i)$
    end for
    Update θ with the clipped turn-level GRPO objective in Eq. (12)
  end for
end for
return π_θ

For every sampled action, the environment supplies genuine feedback before the successor state is scored. Wordle uses the exact color-feedback transition. GuessWhat applies every policy-generated question to the fixed Answerer over all candidates in C, while the frozen question library is used only for reference maximization over $\mathcal{A}_{\text{ref}}$. AUDITEDBRANCHES materializes genuine successor states for available high-value alternative actions; PRIORITIZEDMIX favors late, low-cardinality, unsuccessful, high-variance, and branch-successor states while enforcing the bounded replay fraction.

B COMPLETE METRIC DEFINITIONS

All metrics follow the clembench evaluation protocol. Let N be the total number of episodes, N_{succ} the number solved successfully, and N_{abort} the number aborted because of protocol or parsing failures. Accuracy and Played are

$$\text{ACC} = 100\frac{N_{\text{succ}}}{N}, \qquad \text{Played} = 100\frac{N - N_{\text{abort}}}{N}. \tag{14}$$

Quality is computed from an episode-level efficiency score $q_i \in [0, 100]$. Aborted episodes are excluded, whereas a non-aborted but unsuccessful episode receives $q_i = 0$. For a successful Wordle

episode completed in t_i guesses, the official speed score is

$$q_i^{\text{WORDLE}} = \begin{cases} 100, & t_i \in \{1,2,3\}, \\ 50, & t_i = 4, \\ 30, & t_i = 5, \\ 20, & t_i = 6. \end{cases} \tag{15}$$

For a successful GuessWhat episode completed in t_i requests, the official scorer uses a maximum horizon $T = 8$ and a lower-bound threshold $b = 5$:

$$q_i^{\text{GUESSWHAT}} = \begin{cases} 100, & t_i \le b, \\ 100\dfrac{T - t_i}{T - b}, & b < t_i \le T. \end{cases} \tag{16}$$

Let $\mathcal{P} = \{i : i \text{ is not aborted}\}$ denote the set of played episodes. Quality is the arithmetic mean of the episode scores over played episodes:

$$\text{Quality} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} q_i. \tag{17}$$

Thus, Quality rewards reaching the correct answer with fewer interaction turns while assigning zero efficiency to valid but unsuccessful play. Since Quality excludes aborted episodes, it does not capture the protocol failures reflected by Played. Clemscore corrects for this distinction through the relation given in Section 3.2.

C Experimental Details and Reproducibility

C.1 Dataset Construction and Split Details

Wordle. Search-aligned SFT uses 2,000 state–action demonstrations constructed from 971 unique target words. The fixed-state RL stage contains 1,000 states generated from 534 unique targets, which are a subset of a separate 700-word RL target universe. Each of the three OPBR epochs samples from this universe and materializes exactly 1,000 states, yielding 3,000 OPBR training rows per seed. Checkpoint selection uses 124 full-game development targets, and the selected checkpoint is evaluated once on a clean test set of 168 targets. The SFT targets, RL universe, development targets, and test targets are mutually disjoint.

GuessWhat. We allocate 20 training, 5 development, and 10 test instances from each of the six experiment types, giving 120/30/60 instances in total. Each instance contains eight candidate targets. Enumerating every target in the training instances produces 960 successful teacher trajectories; splitting them at every supervised action yields 3,908 turn-level SFT examples, comprising 2,948 questions and 960 final guesses. The same interaction histories define a fixed RL pool of 3,908 states. Targets and complete candidate sets are disjoint across splits, although ordinary distractor words may recur because the public ontology does not support strict vocabulary-level disjointness across all six experiment types.

OPBR samples only from the corresponding training split. In each epoch, it collects at least 1,250 fresh on-policy states, adds audited counterfactual branch successors, and selects exactly 1,000 states for optimization. The first epoch contains only current-policy states; each subsequent epoch combines 750 current states with 250 replay states. Thus, each seed uses 3,000 materialized OPBR rows in addition to its fixed-state training pool. Development and test instances never contribute optimization samples.

C.2 Prompt Templates

All bracketed fields below are populated at runtime. The rendered messages use the native Qwen3 chat template with thinking disabled; the empty no-thinking control tokens inserted by the tokenizer are omitted for readability.

Wordle. The following user-message template constructs each S-SFT and RL state from the complete interaction history:

```
You are a language wizard who plays Wordle by following
all public game rules.
The hidden target is an English word with exactly five
lowercase letters.
You have at most six total guesses.  Use the complete
history below to choose only the next guess.

Previous guesses, all incorrect answers:
[for each previous turn t]
t.  guess:  [guess_t]
  guess_feedback:  [letter_1]<[color_1]> ...
[letter_5]<[color_5]>
[end history]

Remaining guesses including this one:
[remaining_turns].
Wordle guidance for the next guess:
- Do not repeat any previous guess.
- Use feedback from every previous turn, not only the
latest turn.
- Choose a valid lowercase English word with exactly
five letters.
- Usually preserve confirmed clues, but a legal
information-gathering probe may depart from them when it
gives better search value.

Provide your response only in this format:
explanation:  concise reason based on the public
feedback
guess:  word
```

When the history is empty, the history block is replaced by `(none)`. Formal game-level evaluation retains the native clembench Wordle task message and Game Master feedback; the required visible response schema remains `explanation: ...` followed by `guess: word`.

GuessWhat. The following initial user message is shared by S-SFT construction, RL rollouts, and formal evaluation:

```
Let's play a guessing game.  Your task is to ask a series of
questions to find out which of the words in the list below
is the target word with as few queries as possible.  The
maximum number of questions you can ask to help you narrow
down the options is [max_questions].  Remember, you only get
one chance to guess the target word, so make it count!

The rules of the game are:

1.  Only ask one question per turn.  DO NOT ASK MORE THAN
ONE QUESTION.

2.  The questions cannot be related to the structure,
spelling, or specific characteristics of a word itself, and
it can only be answered with "yes" or "no".

3.  Use the following format for asking questions and DO NOT
ADD ANY TEXT OTHER THAN THIS:

QUESTION: <your question>

4.  You have only one opportunity to guess the target word.
To make your guess, use the following format and DO NOT ADD
ANY TEXT OTHER THAN THIS:

GUESS: <target word>

Remember you cannot ask for a hint.  Make your question
brief.  Let's start!

List of candidate words:  [candidate_list]
```

In all experiments, `[max_questions]` is set to 7. After each valid question, the fixed Answerer supplies the next user message as `ANSWER: [yes/no]`. The full message history is retained, and the policy either emits one further `QUESTION: ...?` or terminates with `GUESS: [target word]`.

C.3 Reward and Search Hyperparameters

For both tasks, the PVF coefficient is

$$\lambda(s_t) = w(|C_t|)\,\mathrm{clip}\left(\frac{b_t - 1}{\tau}, 0, 1\right), \tag{18}$$

where b_t is the number of actions remaining, including the current action. Table 6 gives the task-specific search discount, urgency scale, and complete candidate-count weighting functions used in every main-method seed.

Table 6: Task-specific FSV and PVF hyperparameters.

Setting	Wordle	GuessWhat
Search discount γ	0.95	0.92
Urgency scale τ	4	7
$w(\lvert C\rvert)$	0.10 for $1 \le \lvert C\rvert \le 2$; 0.30 for 3–8; 0.60 for 9–32; 0.80 for $\lvert C\rvert \ge 33$	0 for $\lvert C\rvert = 1$; 0.15 for $\lvert C\rvert = 2$; 0.35 for 3–4; 0.55 for 5–6; 0.70 for $\lvert C\rvert \ge 7$
Reference search	Exact feedback matrix; bounded exact recursion when $\lvert C\rvert \le 8$	Frozen verified question library for $Q_{\max}$; policy questions partitioned by the fixed Answerer

C.4 Training Configuration

All primary experiments start from `Qwen/Qwen3-8B` with hidden thinking disabled in SFT rendering, RL rollouts, and evaluation. Both tasks use low-rank adaptation (LoRA) (Hu et al., 2022) with rank 64, scaling factor 32, and dropout 0.05. GuessWhat targets the query, key, value, and output projections; Wordle additionally targets the gate, up, and down projections. Wordle SFT uses 4-bit quantized LoRA (QLoRA) (Dettmers et al., 2023), whereas RL loads the base model in bfloat16 and optimizes the inherited LoRA adapter. GuessWhat uses bfloat16 LoRA training throughout.

The cross-model baselines use the same frozen trajectory-level SFT protocol: a 7,110/291 train/validation split, 700 optimizer steps, batch size 4, maximum length 1,024, learning rate 2×10^{-4}, weight decay 0.01, five warmup steps, and LoRA with rank 64, scaling factor 32, and dropout 0.05. LoRA target modules follow the corresponding model architecture, and the native tokenizer and chat template are retained through rollout and evaluation.

Table 7: Primary training hyperparameters. RL settings are shared by Wordle and GuessWhat unless stated otherwise.

Setting	Wordle SFT	GuessWhat SFT	HBR-GRPO
Training rows	2,000	3,908	1,000 states/epoch
Schedule	700 steps	3 epochs	2 fixed + 3 OPBR epochs
Optimizer steps	700	735	$500 + 750 = 1{,}250$
Batch size	4	4	4 states
Gradient accumulation	1	4	8 response microbatches
Rollout group size	–	–	8 responses/state
Learning rate	2×10^{-4}	1×10^{-4}	1×10^{-6}
Optimizer	AdamW 8-bit	AdamW	AdamW
Weight decay	0.01	0.01	0.01
Maximum sequence length	1,024	1,024	–
Maximum generated tokens	–	–	128
Sampling temperature	–	–	0.75

Each fixed-state epoch contains 250 optimizer steps. Each OPBR epoch also contains 250 steps and exactly 1,000 selected states; the on-policy collection multiplier is 1.25 and the replay fraction is

0.25 after the first epoch. A training batch therefore contains four states and $4 \times 8 = 32$ sampled responses. Main-method results use seeds 20260711–20260715, while the ablation tables use seed 20260713.

C.5 GRPO Objective and Regularization

Table 8 reports the optimization details shared by the fixed-state and OPBR stages. The probability ratio is computed once per response from the mean log-probability of its generated tokens. Each rollout group is updated once; there are no additional PPO-style optimization epochs over the same response batch.

Table 8: Complete HBR-GRPO optimization and regularization configuration.

Setting	Wordle	GuessWhat
Policy-ratio clipping	$\varepsilon = 0.2$; ratio clipped to $[0.8, 1.2]$	$\varepsilon = 0.2$; ratio clipped to $[0.8, 1.2]$
Advantage normalization	Within-state population mean/std; advantages set to 0 when $\sigma < 10^{-6}$	Same
Zero-variance handling	Up to 2 group resamples (3 draws total)	Same
State batch / rollout group	4 states / 8 responses per state	Same
Generation	Temperature 0.75, top-p 1.0, top-k 0, 128 new tokens, thinking disabled	Same
Optimizer	AdamW, lr $= 10^{-6}$, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon_{\mathrm{Adam}} = 10^{-8}$	Same
Schedule and weight decay	Constant learning rate, no warmup, weight decay 0.01	Same
Kullback–Leibler (KL)/reference regularization	Adapter-disabled base reference; KL coefficient $\beta = 0$ (inactive)	Same; reference log-probabilities are skipped because $\beta = 0$
Other regularization	Global gradient-norm clipping at 1.0; LoRA dropout 0.05	No additional gradient clipping; LoRA dropout 0.05
Precision and update scope	bfloat16; inherited LoRA parameters only	Same
Forward microbatch	4 responses; 8 chunks accumulated into one 32-response update	Same

No entropy bonus, reward-model regularizer, or active KL penalty is used. The only policy-objective clipping is the GRPO ratio clipping above; the separate FSV reward clipping in Equation 8 bounds the scalar reward before within-state advantage normalization.

C.6 Fixed GuessWhat Answerer

GuessWhat evaluates every valid policy-generated question by querying a fixed Answerer independently for each currently feasible candidate. The Answerer's yes/no labels induce the candidate partition used for $Q_b(s, a)$ and HP, while the frozen verified question library is used only to compute $Q_{\max,b}(s)$. Table 9 reports the exact API configuration used by all five seeds.

Table 9: Fixed Answerer configuration for GuessWhat reward computation.

Setting	Value
Provider/interface	DeepSeek OpenAI-compatible API (`https://api.deepseek.com`)
Model	`deepseek-v4-flash`
Decoding	Temperature 0; maximum 512 tokens; thinking enabled
Reliability	90-second timeout; at most 4 attempts with exponential backoff
Parallelism	8 Answerer workers
Prompt version	`clembench-guesswhat-v3.0-answerer-exact-v1-thinking`
Cache key	Model, prompt version, thinking mode, target candidate, and question
Accepted output	A parseable `ANSWER: yes/no` (or equivalent yes/no JSON field)

The Answerer receives one user message and no separate system message. With bracketed fields substituted at runtime, the exact template is:

```
Let's play a guessing game!  Your task is to answer the
other player's questions.  Based on your knowledge of
the word:  [TARGET WORD], respond to the following
questions or guesses.  Limit your response to only 'yes'
or 'no' with no explanation or other words.  Never
reveal the answer in your response.

You must reply using the format below and DO NOT ADD ANY
TEXT OTHER THAN THIS:

ANSWER: <some text>

Target Word:  [TARGET WORD]

QUESTION: [QUESTION]
```

Responses outside the binary schema are marked unresolved and retried up to the limit above. A question is scored only when the complete active candidate set receives a reliable binary partition; otherwise, the action receives zero reward and is retained in the audit counters.

C.7 Construction of High-Quality Reference Action Libraries

The reference actions are constructed before policy optimization and never use development or test targets. They provide the state-dependent maximum $Q_{\max,b}(s)$ and, when applicable, an audited alternative for branch materialization. They do not constrain the policy action space: every valid policy-generated Wordle guess is scored through the exact game transition, and every valid policy-generated GuessWhat question is partitioned by the fixed Answerer described above.

Wordle. We first remove all held-out clembench targets from the official answer list to obtain the training answer universe. The allowed-guess lexicon is the union of the public single-step candidate list and all official answers. We then precompute and freeze the exact color-feedback matrix between every allowed guess and every training answer. To identify broadly useful probes, each allowed word $g = g_1 \dots g_5$ receives the deterministic global score

$$S(g) = \sum_{j=1}^{5} f_j(g_j) + 0.35 \sum_{c \in \mathrm{uniq}(g)} f(c) - 0.15\big(5 - |\,\mathrm{uniq}(g)|\big)|\mathcal{Y}_{\mathrm{train}}|, \tag{19}$$

where $f_j(c)$ and $f(c)$ are respectively the positional and word-level frequencies of letter c in the training answer universe $\mathcal{Y}_{\mathrm{train}}$. The first 768 words under this ranking form the high-information core; a seed-fixed sample of at most 768 words from the remaining lexicon provides lexical diversity. At a state, the reference pool contains the feasible answer words (all of them when there are at most 256, otherwise a deterministic sample), up to 128 unused core probes, and up to 128 unused diversity probes. Actions are reranked by the finite-horizon value in Equation 6, using the exact feedback matrix and $\gamma = 0.95$. Continuation values are solved recursively when $|C| \le 8$ and use the discounted blind-search approximation for larger sets. The selected reference value and top eight actions are stored with every constructed state; the lexicon, answer universe, matrix, and generated state files are bound to SHA-256 manifests.

GuessWhat. We construct a separate library for each of the 120 training instances, each containing eight candidates. For every instance, `deepseek-v4-flash` first generates two batches of 12 questions at temperature 0 with thinking disabled. The batches emphasize complementary semantic axes, including category and function, contrasts between close concepts, and abstract relations or purposes. Questions that mention a candidate, refer to spelling or word form, contain multiple queries, or violate the context-free binary-question protocol are rejected and canonical duplicates are removed. The fixed Answerer then labels every candidate independently; questions with an unresolved label or an all-yes/all-no partition are discarded. A second `deepseek-v4-flash` matrix verifier independently returns the complete eight-way answer vector. Agreement is an audited preference rather than a hard filter, so disagreement cannot silently replace the Answerer's environment labels.

The surviving questions are scored by partition entropy and normalized expected candidate reduction. Partitions that are identical up to exchanging yes and no are collapsed, after which a greedy set-cover procedure selects at most 12 actions. At each selection step it prioritizes the number of

previously indistinguishable candidate pairs newly separated, followed by verifier agreement, expected reduction, entropy, and split balance. If any candidate pair remains unresolved, up to two targeted generation rounds of 12 questions explicitly seek a semantic distinction for the uncovered pairs. The final audit independently recomputes every answer count and partition statistic, rejects duplicate questions and answer vectors, and verifies that the planner can isolate each of the eight targets within the seven-question budget before the final guess. The resulting question vectors, model and prompt versions, raw-response hashes, build audit, and library SHA-256 are frozen and shared by all seeds. This library is used only for reference maximization and audited branch proposals; real policy questions are always executed through the fixed Answerer.

C.8 Software and Hardware

The frozen software environment is summarized in Table 10. GuessWhat uses the OpenAI-compatible client for the fixed Answerer interface. Cross-model SFT baselines use LLaMA-Factory at commit `7af909522a951e3ad9f022ea6f88b6755257eaa5` while preserving each model's native tokenizer and chat template.

Table 10: Primary software environment.

Component	Version	Component	Version
Python	3.12.3	PyTorch	2.8.0+cu128
CUDA	12.8	Transformers	5.11.0
PEFT	0.19.1	Accelerate	1.14.0
bitsandbytes	0.49.2	safetensors	0.8.0
OpenAI client	2.45.0	LLaMA-Factory	0.9.5

Training and evaluation jobs run on a single NVIDIA RTX PRO 6000 Blackwell Server Edition GPU with approximately 96 GiB of memory. Jobs are serialized per GPU; no experiment uses multi-GPU or distributed training. Model weights, split files, prompts, generated training data, run configurations, and evaluation outputs are bound to SHA-256 manifests in the released experiment records.

C.9 Checkpoint Selection and Evaluation

Training-time generation uses temperature 0.75 and at most 128 new tokens. Formal evaluation is greedy with temperature 0, the same no-thinking chat rendering used during training, and the native task parser and scorer. Wordle evaluates 14 frozen global checkpoints on the 124-target development set; GuessWhat evaluates 13 checkpoints on its 30-instance development set. Checkpoints are ranked by development success, then Clemscore, lower abort rate, and finally lower global step. The selected checkpoint is evaluated exactly once on the corresponding untouched test split: 168 Wordle targets or 60 GuessWhat instances. Quality is computed independently for each seed before the five-seed mean and sample standard deviation are calculated.

D Checkpoint Selection

Tables 11 and 12 report the complete development-set results used to select the final checkpoint for each HBR-GRPO seed. Each cell gives ACC/Clemscore in percent. Boldface marks the checkpoint selected by the frozen ranking rule described in Appendix C.

Table 11: Wordle checkpoint selection on the 124-target development set. Columns denote global training steps; cells report ACC/Clemscore (%).

Seed	100	200	300	400	500	600	700	750	800	900	1000	1100	1200	1250
20260711	8.06/4.27	**15.32/7.26**	14.52/6.29	12.90/5.56	11.29/5.00	12.10/5.08	10.48/4.68	11.29/4.84	12.10/4.84	8.87/3.63	11.29/4.68	12.10/4.84	10.48/4.35	11.29/4.03
20260712	12.90/5.56	12.10/5.32	12.90/5.89	11.29/4.92	10.48/4.76	16.13/5.89	15.32/6.69	14.52/6.69	13.71/6.53	13.71/6.45	13.71/6.29	12.90/6.13	15.32/6.45	**16.13/6.53**
20260713	9.68/4.76	**15.32/6.45**	12.10/5.40	12.90/5.56	13.71/5.56	12.10/5.00	11.29/5.32	11.29/4.68	13.71/5.40	11.29/4.44	12.10/4.92	12.90/5.24	13.71/5.89	13.71/5.89
20260714	12.10/5.89	12.90/5.81	10.48/4.76	11.29/5.16	10.48/4.52	9.68/4.27	14.52/6.53	13.71/6.13	15.32/6.45	16.13/6.53	13.71/5.65	16.13/6.13	16.13/6.37	**16.94/6.37**
20260715	7.26/4.68	7.26/4.68	13.71/6.21	11.29/5.32	10.48/4.92	11.29/4.92	11.29/5.00	13.71/6.05	10.48/4.52	12.10/4.52	12.10/5.00	14.52/5.40	**16.13/6.94**	13.71/6.37

Table 12: GuessWhat checkpoint selection on the 30-instance development set. Columns denote global training steps; cells report ACC/Clemscore (%).

Seed	100	200	300	400	500	600	700	800	900	1000	1100	1200	1250
20260711	46.67/46.67	**50.00/50.00**	46.67/46.67	40.00/40.00	46.67/46.67	50.00/50.00	40.00/40.00	36.67/36.67	40.00/40.00	36.67/36.67	40.00/40.00	40.00/40.00	40.00/40.00
20260712	46.67/46.67	50.00/50.00	53.33/53.33	**56.67/56.67**	53.33/53.33	53.33/53.33	56.67/56.67	56.67/56.67	56.67/56.67	56.67/56.67	53.33/53.33	50.00/50.00	56.67/56.67
20260713	50.00/50.00	46.67/46.67	43.33/43.33	50.00/50.00	50.00/50.00	43.33/43.33	43.33/43.33	40.00/40.00	**53.33/53.33**	50.00/50.00	43.33/43.33	46.67/46.67	50.00/50.00
20260714	40.00/40.00	46.67/46.67	50.00/50.00	50.00/50.00	46.67/46.67	53.33/53.33	53.33/53.33	50.00/50.00	53.33/53.33	53.33/53.33	53.33/53.33	**56.67/56.67**	50.00/50.00
20260715	50.00/50.00	50.00/50.00	46.67/46.67	53.33/53.33	**56.67/56.67**	50.00/50.00	50.00/50.00	50.00/50.00	53.33/53.33	50.00/50.00	50.00/50.00	50.00/50.00	43.33/43.33

E Detailed Failure Analysis

E.1 HP-only Closing Diagnostics

Figure 2 and Table 13 quantify the answer-commitment limitation motivating FSV. Although HP-only policies often reduce the feasible set to a singleton, successful termination remains substantially less frequent.

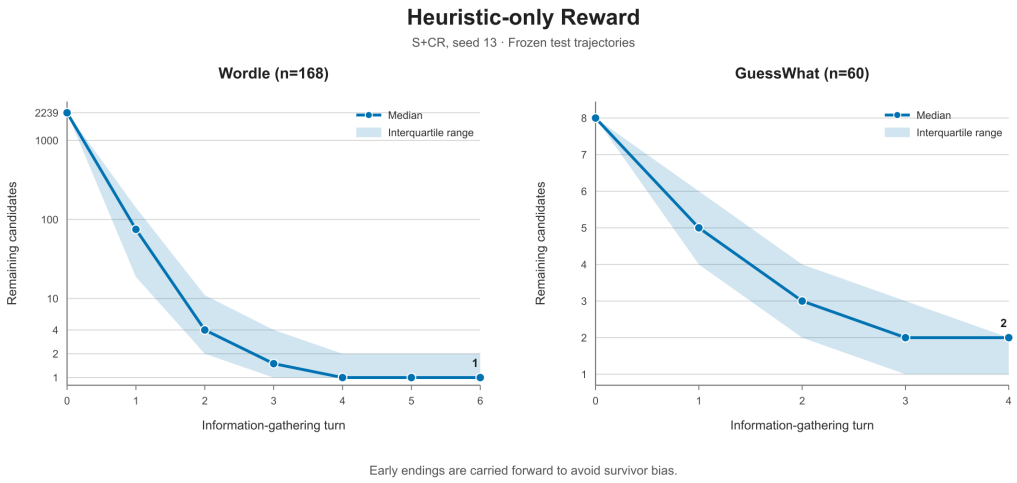


Figure 2: Candidate reduction under the heuristic-only reward on the frozen seed-13 test trajectories. Curves show the median and interquartile range after each information-gathering turn. Early endings are carried forward to avoid survivor bias.

Table 13: Closing behavior under the heuristic-only reward. "Singleton" means that one candidate remains immediately before the trajectory's final model turn.

Dataset	Games	Singleton	Success	Singleton, unsolved
Wordle	168	112 (66.7%)	18 (10.7%)	99
GuessWhat	60	22 (36.7%)	29 (48.3%)	6

E.2 Representative Failure Cases

Cases 1 and 3 reveal a persistent answer-commitment failure: even after the interaction history has reduced the feasible answer set to a singleton, the policy may still produce an incorrect or invalid final guess (Figures 3 and 5). As discussed above, HBR-GRPO substantially improves search and decision quality, but the final policy does not eliminate this failure mode entirely.

Case 2 highlights a complementary limitation in semantic partitioning (Figure 4). Unlike Wordle, where feedback directly constrains letters and positions, GuessWhat requires the model to formulate an abstract property that divides a heterogeneous candidate set efficiently. The policy sometimes fails to identify a property shared by roughly half of the remaining candidates and instead asks overly

specific questions that eliminate words one at a time, leading to inefficient use of the interaction budget.

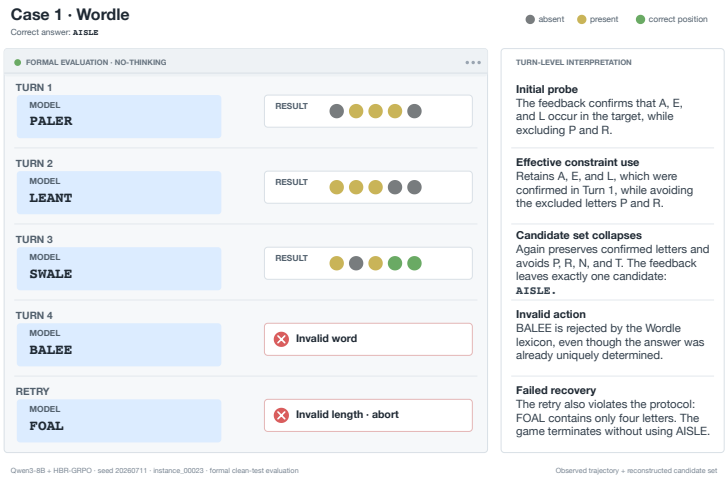


Figure 3: A representative Wordle failure case with `aisle` as the target word.

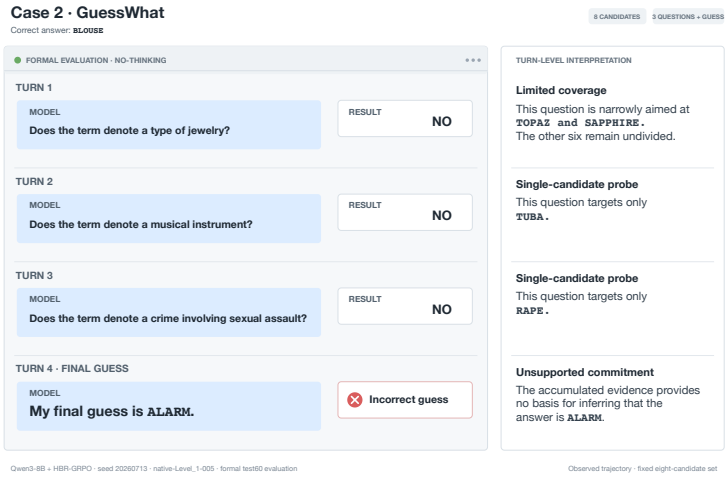


Figure 4: A representative GuessWhat failure case with `blouse` as the target word.

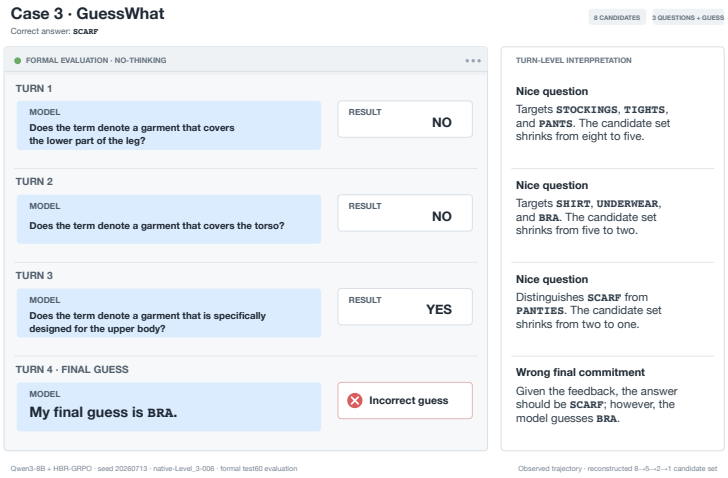


Figure 5: A representative GuessWhat failure case with `scarf` as the target word.

F FSV CLIPPING AUDIT

We audit the five-seed main-method training logs and count a substantive upper-clipping event when an action entering the FSV normalization branch satisfies $Q_b(s,a) - Q_{\max,b}(s) > 10^{-9}$. This tolerance excludes ratios that exceed one only through floating-point roundoff. Correct direct guesses in GuessWhat receive terminal reward 1 explicitly and bypass this normalization branch.

Table 14: Observed FSV upper-clipping frequency during five-seed main-method training. "Normalized" denotes audited actions that enter the $Q_b/Q_{\max,b}$ branch.

Dataset	Audited	Normalized	Clips	Normalized rate	Audited rate
Wordle	25,000	19,547	252	1.289%	1.008%
GuessWhat	200,000	75,801	67	0.0884%	0.0335%

GuessWhat retains action-level statistics for all 32 responses sampled at every optimizer step, so its row covers all 200,000 training actions. The Wordle logger retains four of the 32 responses per step; its row is therefore an exact count over the 25,000 retained diagnostic actions (12.5% coverage), rather than an estimate of the unlogged actions. In both cases, clipping is a low-frequency saturation safeguard rather than a primary source of reward.

G HOTPOTQA GENERALIZATION DETAILS

G.1 DATASET CONSTRUCTION AND FROZEN SEARCH ENVIRONMENT

We use the labeled HotpotQA FullWiki training and validation data (Yang et al., 2018). The 90,447 official training examples remain the training split. We deterministically partition the labeled validation set using `blake2b(question_id, digest_size=8) mod 2`, assigning bucket 0 to development and bucket 1 to held-out testing. This produces 3,725 development and 3,680 held-out examples. The latter is not the official hidden HotpotQA test set. Development labels are used only for checkpoint selection, and neither development nor held-out labels enter S-SFT or RL optimization.

All interactions use the immutable HotpotQA English Wikipedia abstracts snapshot dated 2017-10-01, containing 5,233,329 introductory article passages. We build a contentless SQLite FTS5 index over titles and paragraph bodies with the `unicode61` tokenizer and BM25 title/body weights $(4.0, 1.0)$. Queries are normalized, deduplicated in first-occurrence order, truncated to 32 tokens, and converted to OR queries. Ties are resolved by case-insensitive title and then row ID. Each search deterministically returns the top five articles and exposes at most 12 sentences per article. No live web request, embedding retriever, learned reranker, NLI model, or LLM judge is used.

At each turn, the policy observes the question, remaining budget, query history, and cumulatively selected evidence, but not gold answers, gold supporting facts, rewards, reference actions, or future results. It emits either a JSON search action `{"query": "..."}` or a terminal action `{"answer": "..."}`. After retrieval, the same policy selects exact `title::sentence_id` evidence identifiers from the current results. Query generation and evidence selection constitute one macro-action, and at most nine searches are allowed.

G.2 PROMPT TEMPLATES

All prompts use the native Qwen3 chat template with `enable_thinking=False`. The system message is:

```
You are an evidence-grounded multi-hop search agent. Use
only the frozen Wikipedia search results and selected
evidence shown by the environment. Return exactly one JSON
object. Do not reveal hidden reasoning or add prose outside
JSON.
```

The S-SFT query prompt is:

```
Choose one concise Wikipedia search query for the next
retrieval turn.  The query must be grounded in the question
or selected evidence.  Return {"query":  "..."}.
STATE:
[sorted public-state JSON]
```

The RL and evaluation action prompt is:

```
Choose the next action from the current observable state.
If the selected evidence is sufficient, submit the shortest
exact answer.  Otherwise, issue one concise Wikipedia search
query grounded in the question or selected evidence.  Return
exactly one of {"query":  "..."} or {"answer":  "..."}.
STATE:
[sorted public-state JSON]
```

After each search, evidence selection uses:

```
Select only non-duplicate sentences that are useful for
answering the question.  Use exact IDs from SEARCH_RESULTS.
Return {"evidence":  ["title::sent_id", ...]}.  Return an
empty list when no result is useful.
STATE:
[sorted public-state JSON]
SEARCH_RESULTS:
[sorted search-result JSON]
```

When the search budget is exhausted, the forced-answer prompt is:

```
Answer the question using only selected evidence.  Return
{"answer":  "..."}.
STATE:
[sorted public-state JSON]
```

Action and forced-answer generations are limited to 64 new tokens; evidence selection is limited to 192 new tokens. Formal evaluation uses greedy decoding and batch size 8.

G.3 HotpotQA HP, FSV, and PVF Rewards

Let E_t denote the selected evidence at state s_t and E^* the gold supporting-fact set. We define exact-evidence overlap as

$$I(E_t, E^*) = \frac{|E_t \cap E^*|}{|E_t \cup E^*|}, \tag{20}$$

with value 1 when both sets are empty. The HotpotQA heuristic-progress reward is the positive intersection-over-union (IoU) increment induced by an actual search transition:

$$r_{\mathrm{HP}}(s_t, a_t) = \mathrm{clip}(I(E_{t+1}, E^*) - I(E_t, E^*), 0, 1)\,. \tag{21}$$

FSV uses the same finite-horizon recursion as the main method with discount $\gamma = 0.95$. A policy search is valued from its actual retrieved-and-selected successor evidence. Frozen reference actions are used only to compute $Q_{\max,b_t}(s_t)$, after which

$$r_{\mathrm{FSV}}(s_t, a_t) = \begin{cases} \mathrm{clip}(Q_{b_t}(s_t, a_t)/Q_{\max,b_t}(s_t), 0, 1)\,, & Q_{\max,b_t}(s_t) > 0, \\ 0, & Q_{\max,b_t}(s_t) = 0. \end{cases} \tag{22}$$

A correct terminal answer receives 1 and an incorrect answer receives 0. Malformed and repeated search actions receive zero search value.

PVF is

$$r_{\mathrm{PVF}} = (1 - w_m) r_{\mathrm{HP}} + w_m r_{\mathrm{FSV}}, \tag{23}$$

where $m = |E^* \setminus E_{t+1}|$ is the number of missing supporting facts and $w_0 = 1.0$, $w_1 = 0.65$, $w_2 = 0.35$, and $w_m = 0.25$ otherwise. Rewards are normalized only among responses sampled from the identical pre-action state. The query and its evidence-selection tokens share one macro-action advantage; prompt and search-feedback tokens are masked.

G.4 S-SFT AND INITIAL RL-STATE CONSTRUCTION

Search-aligned SFT trajectories are constructed only from training examples with a visibility-constrained planner. A query may use the question, selected evidence, and entities already exposed by that evidence; hidden supporting titles cannot be injected before becoming visible. Every candidate query is executed against the same frozen BM25 retriever used for RL and evaluation. Examples without an audited visible path are rejected rather than repaired with hidden labels. The resulting S-SFT data contain 20,000 training rows (8,000 query, 8,000 evidence-selection, and 4,000 answer rows) and 2,000 evaluation rows. The construction accepts 4,000 training episodes, and all 9,241 supporting facts in those episodes resolve against the frozen index.

The same accepted training episodes produce 12,000 public RL states and 4,000 reference rows. Reference actions are used only for $Q_{\max}$ and are never inserted into the policy-visible state. The fixed PVF stage deterministically samples 1,000 states per epoch for two epochs. The subsequent three OPBR epochs also optimize 1,000 states each. OPBR stores states rather than actions or rewards, resamples current-policy actions upon reuse, has capacity 4,096 and maximum age two epochs, and uses four branches per state. Its first epoch uses only current states; later epochs contain at most 25% historical states.

G.5 HBR-GRPO OBJECTIVE AND REGULARIZATION

Table 15 reports the optimization details shared by the fixed-state and OPBR stages. The probability ratio is computed once per response from the mean log-probability of its generated tokens. Each rollout group is updated once, without additional PPO-style epochs over the same response batch.

Table 15: HotpotQA HBR-GRPO optimization and regularization configuration.

Setting	Configuration
Policy-ratio clipping	Response-level $\varepsilon = 0.2$; ratio clipped to $[0.8, 1.2]$
Advantage normalization	Population mean and standard deviation within the identical pre-action state; zero when variance vanishes
State batch / rollout group	4 states / 8 independently sampled responses per state
Generation	Temperature 0.75; thinking disabled; 64 action tokens and 192 evidence-selection tokens
Optimizer	AdamW, learning rate 10^{-6}, weight decay 0.01, constant schedule without warmup
KL/reference regularization	KL coefficient $\beta = 0$; frozen reference actions used only for $Q_{\max}$
Other regularization	Global gradient-norm clipping at 1.0; inherited LoRA dropout 0.05
Precision and update scope	bfloat16 with TensorFloat-32 (TF32); 87,293,952 inherited LoRA parameters only
Token credit assignment	One macro-action advantage shared by query and evidence-selection response tokens; prompt and tool tokens masked

No entropy bonus, reward-model regularizer, or active KL penalty is used. Historical replay stores states only, so actions, environment transitions, rewards, and advantages are regenerated under the current policy whenever a state is reused.

G.6 SOFTWARE AND HARDWARE

The frozen software environment is summarized in Table 16.

Table 16: Primary software environment for the HotpotQA experiment.

Component	Version	Component	Version
Python	3.12.3	PyTorch	2.8.0+cu128
CUDA	12.8	Transformers	4.51.3
PEFT	0.15.2	SQLite FTS5	Built-in extension

HBR-GRPO training and evaluation run on one NVIDIA RTX PRO 6000 Blackwell GPU with approximately 97 GB of memory. The model uses bfloat16 weights and scaled dot-product attention (SDPA). The contentless FTS5 index uses `unicode61 remove_diacritics 2` and is opened read-only with `immutable=1`. RL inputs are limited to 2,048 tokens, whereas formal evaluation allows 8,192 input tokens. The frozen 2017-10-01 Wikipedia corpus, deterministic BM25 index, split manifests, prompts, model initializer, and evaluation protocol are unchanged across the reported methods; no online search or external model API is invoked.

G.7 Hyperparameter Settings

All runs start from `Qwen/Qwen3-8B` with thinking disabled. HBR-GRPO continues training from the final seed-13 S-SFT LoRA adapter. Table 17 gives the complete stage-level training schedule. LoRA targets the query, key, value, output, gate, up, and down projections.

Table 17: HotpotQA S-SFT and HBR-GRPO training hyperparameters.

Setting	S-SFT	Fixed-state PVF	PVF+OPBR
Training / evaluation rows	20,000 / 2,000	1,000 states/epoch	1,000 states/epoch
Schedule	3 epochs	2 epochs	3 epochs
Optimizer steps	3,750	500	750
Batch size	8 examples	4 states	4 states
Gradient accumulation	2 SFT batches	One 4-state update	One 4-state update
Forward microbatch	–	2 response segments	2 response segments
Rollout group size	–	8	8
Learning rate	2×10^{-5}	1×10^{-6}	1×10^{-6}
Weight decay	0	0.01	0.01
Input length	2,048	2,048	2,048
Sampling temperature	–	0.75	0.75
Learning-rate schedule	Cosine; 3% warmup	Constant	Constant
LoRA rank / alpha / dropout	32 / 64 / 0.05	Inherited	Inherited
OPBR capacity / branches	–	–	4,096 / 4
Historical replay	–	–	0% in epoch 1; at most 25% later
Maximum replay age	–	–	2 epochs

We select among HBR-GRPO steps 900, 1,000, 1,100, 1,200, and 1,250 using development Joint F1, followed by Joint EM, Answer F1, and the earlier step as tie-breakers; step 1,250 is selected. The fixed checkpoint is then evaluated exactly once on all 3,680 held-out examples. We report the official HotpotQA Answer EM/F1, Supporting Fact EM/F1, and Joint EM/F1 metrics. All examples, including aborts, enter the average.