AGENTIC POLICY IMPROVEMENT THROUGH VALUE-GUIDED INFORMATIVE SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has substantially improved the reasoning capabilities of large language models. However, its learning signal is limited by the trajectories discovered by the current policy. When group rollouts repeatedly follow similar or low-value paths, the resulting verified signal provides little guidance for learning better behavior. Existing search-based rollout methods can increase trajectory diversity, but diversity alone neither guarantees a better rollout policy nor ensures that search improvements are transferred back to the model. We therefore study how to make search during RLVR a principled policy-improvement operator. We propose APIVIS, a rollout-and-learning framework that applies value-guided search to domain-native language-model decisions and combines searched and direct trajectories within each rollout group. APIVIS further distills search-improved decisions, preserving an improvement-aligned learning signal even when group-relative returns are uninformative. We prove statewise and trajectory-level policy improvement under exact values, together with an explicit degradation bound under bounded value errors. Across established benchmarks in mathematical reasoning and multi-turn interaction, APIVIS outperforms established RLVR and search-based baselines, demonstrating the effectiveness of search as a learnable policy-improvement operator.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) turns executable outcomes into training signals. It has shown promising results for improving mathematical reasoning (Shao et al., 2024; DeepSeek-AI, 2025; Lambert et al., 2024). The same principle extends to interactive agents, where environment success can be verified deterministically (Shridhar et al., 2021; Yao et al., 2022). Common RLVR algorithms, including GRPO (Shao et al., 2024), DAPO (Yu et al., 2025), and their variants, differ in optimization details but collect experience in the same way. They sample G trajectories independently from the current policy, verify them, and credit each trajectory by its group-relative return. At the trajectory level, the exploration operator is therefore i.i.d. ancestral sampling. When a group has a common final return, its relative advantages vanish; more generally, no optimizer can learn a useful behavior that the rollout distribution never reaches. Recent analyses likewise suggest that RLVR mainly sharpens behavior already accessible under the base model's sampling distribution (Yue et al., 2025).

A natural remedy is to make trajectory generation *principled* instead of relying on independent sampling. At inference time, value- and process-reward-guided search already improves returned trajectories by selecting among candidate continuations (Yao et al., 2023; Snell et al., 2024; Wang et al., 2025; Lightman et al., 2023). This success has motivated training-time methods that branch or resample rollouts to increase trajectory diversity (Xing et al., 2025; Zhao et al., 2026). Yet a more diverse batch does not by itself define a better rollout policy, and a better response found by search is not necessarily transferred back to the model. We therefore ask: *can value-guided search act as a guaranteed policy-improvement operator inside RLVR, and can its improvements remain learnable across language-model decision processes?*

AlphaZero and MuZero provide a principled template: search is not merely a decoder that returns one better action, but an improvement operator that defines an improved action distribution and distills it into the policy (Silver et al., 2018; Schrittwieser et al., 2020). Gumbel MuZero makes this

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.

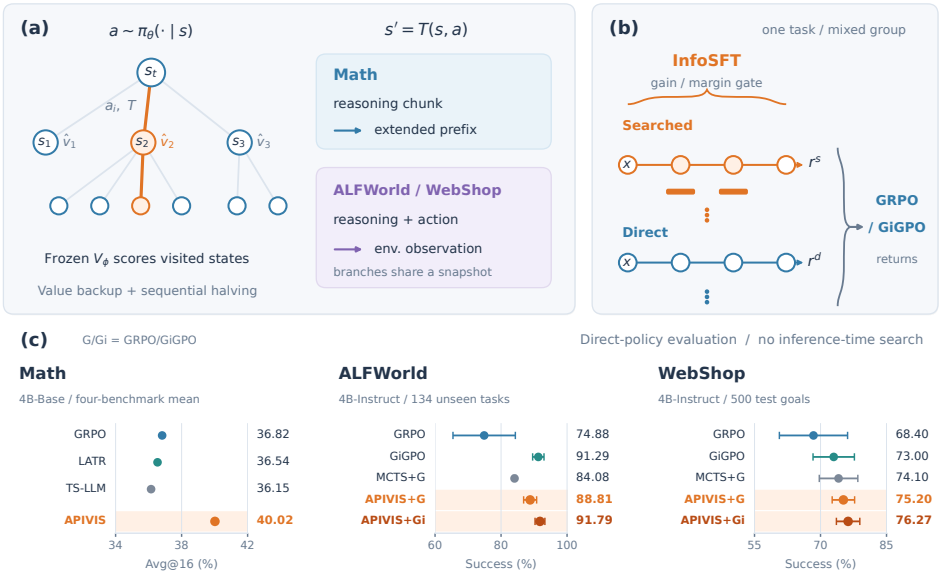

Figure 1: APIVIS at a glance. (a) At each searched state, frozen values guide Gumbel sequential halving over policy-sampled, domain-native decisions: reasoning chunks in math and reasoning–action steps in ALFWorld and WebShop. (b) Mixed rollout groups combine direct and searched trajectories; group-relative returns train GRPO or GiGPO, while InfoSFT distills search-improved decisions. (c) Direct-policy results without inference-time search on four math benchmarks (Avg@16), ALFWorld `valid_unseen` (134 tasks), and WebShop (500 goals), where APIVIS variants outperform the evaluated baselines.

view explicit by monotonically reweighting prior actions with completed action values, yielding a policy-improvement guarantee while sequential halving allocates a limited simulation budget (Danihelka et al., 2022). Translating this operator from board games to RLVR is nontrivial. Board-game actions form a finite, enumerable set; language-model decisions are variable-length objects sampled from the policy, their values are estimated from partial trajectories, and the RLVR objective need not provide a gradient for the improvement found by search. Bridging these settings raises three questions.

(i) What constitutes an action? Token-level search creates a long decision horizon, while branching only over complete trajectories cannot intervene at intermediate decisions. We search over the smallest domain-native decision that can be executed or evaluated as a unit: a reasoning chunk in mathematics or a reasoning-and-action step in ALFWorld or WebShop. This preserves the structure needed for search while sharing prefixes or environment states across candidates.

(ii) What policy does search improve? The candidate action set is itself sampled from the language-model policy, so a board-game guarantee over a fixed action set does not apply directly. We define the marginal policy induced jointly by candidate generation and monotone value-based selection. Under exact values, it improves expected continuation value at every searched state; the same inequality composes over the complete trajectory when only a subset of decisions is searched, while the guarantee degrades gracefully under bounded value error.

(iii) How is the searched policy learned? AlphaZero-style policy iteration explicitly distills the search policy. In RLVR, placing a searched trajectory in the batch is insufficient because the group-relative gradient can still vanish when final returns are uniform. APIVIS therefore mixes direct and searched trajectories within each group and adds Informative SearchSFT, a selective maximum-likelihood term on committed decisions whose backed-up value improves upon the policy's own candidates. This preserves an improvement-aligned signal when the group-relative term is zero.

We summarize the contributions as follows:

- We formulate value-guided selection over sampled, domain-native actions and prove state-wise and trajectory-level policy improvement, including an explicit degradation bound under value-estimation error.

- We propose APIVIS, which combines finite-budget search, mixed direct and searched rollout groups, and selective supervision so that improvements found by search remain learnable under group-relative RLVR.
- We evaluate APIVIS on mathematical reasoning, ALFWorld, and WebShop, where it consistently outperforms recognized direct-generation and search-based baselines across all three domains.

2 Related Work

RLVR and group-relative optimization. RLVR has emerged as an effective approach for improving LLM reasoning, using automatically verifiable outcomes as reward signals. Within this paradigm, GRPO (Shao et al., 2024) replaces a learned critic with rewards normalized across responses to the same prompt. The group-relative formulation has been adopted at scale in systems such as DeepSeek-R1 (DeepSeek-AI, 2025) and Tülu 3 (Lambert et al., 2024), with subsequent methods such as DAPO (Yu et al., 2025) refining the optimization procedure with clipping, prompt filtering, and length handling. These methods primarily improve optimization over collected rollouts, while leaving rollout generation unchanged. APIVIS instead improves the rollout policy itself through value-guided search and selectively distills the resulting improvements, while retaining the group-relative objective.

Search for reasoning and policy improvement. At inference time, Tree-of-Thought search (Yao et al., 2023), best-of-n and beam search with process reward models (Lightman et al., 2023), and compute-aware test-time scaling (Snell et al., 2024) show that guided selection can outperform direct sampling. VGS (Wang et al., 2025) performs block-wise reasoning search with a token-level value model. These approaches primarily use search to improve inference-time outputs rather than the underlying policy. AlphaZero (Silver et al., 2018), MuZero (Schrittwieser et al., 2020), and Gumbel MuZero (Danihelka et al., 2022) take a different perspective, where search constructs an improved policy that is subsequently distilled into the learned policy. Gumbel MuZero further supports policy improvement under limited simulation budgets through Gumbel-Top-k sampling (Kool et al., 2019) and sequential halving. This search-and-learn paradigm has also been adapted to LLMs. TS-LLM (Feng et al., 2023), ReST-MCTS* (Zhang et al., 2024), and rStar-Math (Guan et al., 2025) use tree search to improve LLM decoding and generate targets for self-training. Their MCTS variants primarily serve as decoders or offline trace generators. More recent work brings search-based exploration directly into online RLVR: LATR (Xing et al., 2025) and ROSE (Zhao et al., 2026) branch or resample rollouts to increase trajectory diversity. In contrast, APIVIS embeds a value-monotone Gumbel search operator inside each online RLVR group, mixes direct and searched trajectories, and distills selected decisions through InfoSFT. We evaluate TS-LLM's MCTS-α as a controlled tree-search baseline.

Long-horizon interactive tasks. ALFWorld and AgentBoard evaluate embodied, multi-step decision making (Shridhar et al., 2021; Ma et al., 2024), while WebShop tests language-mediated web interaction (Yao et al., 2022). Unlike text-only reasoning, these tasks couple model decisions with environment transitions, making credit assignment and value estimation inherently state-dependent. Recent work addresses these challenges through step-level grouping for agentic RL, as in GiGPO (Feng et al., 2025), and reward modeling over agent states, as in AgentRM (Xia et al., 2025). We use these domains to study whether value-guided policy improvement extends from mathematical reasoning to long-horizon, executable agent decisions.

3 Preliminaries

3.1 RLVR and the group-relative objective

Problem setup. For task $x \sim \mathcal{D}$, policy π_θ produces a tokenized trajectory $y = (y_1, \ldots, y_T)$ with terminal verifier reward $r(x, y) \in \{0, 1\}$. From G trajectories sampled for the same task, GRPO assigns trajectory i the group-normalized advantage $\hat{A}^{(i)} = (r(x, y^{(i)}) - \mu_x)/\sigma_x$ and shares it across its response tokens. Uniform-reward groups therefore provide no policy gradient.

Group-relative objective. We perform one PPO-style update pass per rollout batch. The likelihood ratio is clipped asymmetrically around one. A reference-policy KL term may be included with

coefficient β; the reported experiments set $\beta = 0$. For negative advantages, dual clipping further caps the loss at $-c\hat{A}^{(i)}$, preventing large ratios from dominating the update. Let $\ell_t^{(i)}$ denote this negative clipped surrogate, $m_t^{(i)}$ the response mask, and $\widehat{d}_{\mathrm{KL}}^{(i,t)}$ the sampled KL estimate. The loss is

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{Z_i}\sum_{t=1}^{T} m_t^{(i)}\left(\ell_t^{(i)} + \beta\,\widehat{d}_{\mathrm{KL}}^{(i,t)}\right), \qquad Z_i = \sum_{t=1}^{T} m_t^{(i)}. \tag{1}$$

3.2 A DECISION-LEVEL MDP FOR LANGUAGE-MODEL AGENTS

We factor a trajectory into domain-native *decision chunks* rather than individual tokens. A state s contains the task input, all committed model outputs, and any external environment state. An action a is the next executable or semantically complete model decision; the transition $s' = T(s, a)$ appends the generated tokens and, for interactive tasks, applies the selected action to the environment. The discount is $\gamma = 1$ and the only training reward is the terminal verifier outcome.

The action boundary depends on the domain. In mathematical reasoning, generation stops at a decoded blank line, an "</answer>" tag, EOS, or an emergency limit of L_{safe} tokens. In ALFWorld and WebShop, a chunk contains the reasoning prefix and one parsed environment action; speculative branches execute from the same environment snapshot, and only the selected branch becomes the next state. In every case, truncation is performed on token IDs, all candidates share the same model prefix and external state, and the value model scores the state produced by a complete candidate decision. The search operator and the policy-improvement result in Section 4 do not otherwise depend on the domain-specific boundary.

3.3 GUMBEL–MUZERO PLANNING AT SMALL SIMULATION BUDGETS

Gumbel MuZero (Danihelka et al., 2022) replaces PUCT with a scheme that guarantees policy improvement even when the number of simulations is smaller than the number of root actions. Root actions are sampled without replacement using the Gumbel-Top-k trick (Kool et al., 2019), the simulation budget is divided into $\lceil \log_2 C \rceil$ phases of *sequential halving*, and root actions are ranked by a Gumbel score combined with a monotone transform of the backed-up value:

$$\sigma(a) \;=\; \lambda_g\, g_a \;+\; \left(c_{\mathrm{visit}} + \max_{a'} N(a')\right) c_{\mathrm{scale}}\, \lambda_V\, Q(a), \tag{2}$$

where $g_a \sim \mathrm{Gumbel}(0, 1)$, $N(a)$ is the visit count of root action a, and $Q(a) = W(a)/N(a)$ is its mean backed-up value. With $\lambda_g = 0$ the selection reduces to a deterministic value-guided ranking of the sampled candidates; with $\lambda_g > 0$ it recovers stochastic Gumbel-Top-k selection. We retain this root scoring form, but its use in RLVR requires more than replacing the action labels. The candidate set is sampled from the language-model policy, values score partial trajectories, and the induced policy must marginalize over candidate generation. Section 4.1 defines the resulting search operator, and Theorem 4.1 establishes its improvement guarantee.

4 METHOD

APIVIS augments group-relative RLVR with two complementary components: value-guided search for constructing rollouts and Informative SearchSFT (InfoSFT) for learning from the improvements found by search. For each task, we split an even-sized group of G trajectories evenly between direct policy sampling and search-assisted generation. All trajectories are verified and enter the same group-relative objective. Mixing the two rollout sources increases the opportunity for return contrast within a group, while InfoSFT provides an additional learning signal on search-improved decisions. In particular, when the group has a common terminal return, accepted InfoSFT targets can still update the policy.

At a searched state, APIVIS samples domain-native candidate decisions from the policy, uses a frozen value model $V_\phi(s) \in [0, 1]$ to estimate their probability of eventual task success, and commits the continuation selected by Gumbel sequential-halving search (§4.1). The method searches the first K decisions of each searched trajectory in every domain. Figure 1 summarizes the method, and Algorithm 1 gives the rollout procedure. The domain-specific decision boundaries and value outputs are summarized in Table 6.

4.1 VALUE-GUIDED CHUNK SEARCH

A *search call* replaces one ordinary policy decision at state s. Its generation is controlled by the number of candidates C and simulations S; candidate paths follow the same response-length limit or environment horizon as ordinary rollouts.

Root proposal. We independently draw C candidate chunks $a_1, \ldots, a_C \sim \pi_\theta(\cdot \mid s)$ using the vanilla rollout sampler with temperature τ and nucleus threshold p. Sampling stops at the first domain-specific action boundary (§3.2). We do not oversample, reject, filter, or rerank the proposals before search. Each candidate becomes a root child storing its successor state $T(s, a)$, token IDs, cumulative log-probability, and termination status.

Simulations. Each root candidate receives an independent $g_a \sim \mathrm{Gumbel}(0, 1)$. We divide the S simulations across $m = \max(1, \lceil \log_2 C \rceil)$ sequential-halving phases, allocating visits uniformly among the root candidates that remain in each phase. A simulation first descends from its assigned root candidate by repeatedly choosing the least-visited child, with round-robin tie breaking, until it reaches a terminal or unexpanded leaf. The frozen value model then evaluates the full leaf state, $v = V_\phi(\text{leaf})$. If the leaf is nonterminal and has not been expanded, we sample C vanilla-policy chunks and attach their successor states as children. Finally, the value is backed up along the visited path by updating $N \mathrel{+}= 1$, $W \mathrel{+}= v$, and $Q = W/N$.

After each phase, Eq. (2) ranks the surviving root candidates; the lower-scoring half is removed, and the next phase reallocates its simulations across the smaller set. Unlike one-step value ranking, deeper expansions can change a root candidate's score through backed-up continuation values.

Policy-improvement guarantee. We compare this search-induced decision rule with ordinary chunk sampling. At state s, define

$$Q^{\pi_\theta}(s, a) \;=\; \mathbb{E}[r \mid s,\ a,\ \text{then follow } \pi_\theta]\,, \tag{3}$$

where r is the terminal verifier reward. Let $\mathcal{A}_s = (a_1, \ldots, a_C)$ be an ordered list of chunks sampled independently from $\pi_\theta(\cdot \mid s)$. Because the entries are identically distributed, selecting a slot uniformly has marginal distribution $\pi_\theta(\cdot \mid s)$. For candidate scores $\mathbf{z} = (z_1, \ldots, z_C)$ and a non-decreasing transform $h : \mathbb{R} \to \mathbb{R}$, define the improved distribution over candidate slots by

$$\rho_h^{\mathbf{z}}(i \mid \mathcal{A}_s) \;=\; \frac{\exp(h(z_i))}{\sum_{j=1}^{C} \exp(h(z_j))}. \tag{4}$$

Theorem 4.1 (Chunk-level policy improvement for RLVR)**.** *Fix s and draw $\mathcal{A}_s = (a_1, \ldots, a_C) \sim \pi_\theta(\cdot \mid s)^C$. Let $q_i = Q^{\pi_\theta}(s, a_i)$, let h be non-decreasing, and let $\rho_h^{\mathbf{q}}$ be the candidate-selection distribution in Eq.* (4)*. Then, for every fixed $\mathcal{A}_s$,*

$$\sum_{i=1}^{C} \rho_h^{\mathbf{q}}(i \mid \mathcal{A}_s)\, q_i \;\geq\; \frac{1}{C} \sum_{i=1}^{C} q_i. \tag{5}$$

Let $\pi_h(a \mid s)$ be the marginal chunk policy induced by sampling $\mathcal{A}_s$ and then selecting with $\rho_h^{\mathbf{q}}$. Averaging over candidate lists gives

$$\sum_a \pi_h(a \mid s)\, Q^{\pi_\theta}(s, a) \;\geq\; \sum_a \pi_\theta(a \mid s)\, Q^{\pi_\theta}(s, a) \;=\; V^{\pi_\theta}(s). \tag{6}$$

If ρ_h instead uses estimates $\widehat{Q}$ with $|\widehat{Q} - Q^{\pi_\theta}| \leq \epsilon$ on all candidates, the induced policy $\widehat{\pi}_h$ satisfies

$$\sum_a \widehat{\pi}_h(a \mid s)\, Q^{\pi_\theta}(s, a) \;\geq\; V^{\pi_\theta}(s) - 2\epsilon. \tag{7}$$

Proof. See Appendix A. □

Remark 4.1 (Trajectory-level improvement)**.** *Statewise improvement composes even when only some decisions are searched. Let $d(s) \in \{0, 1\}$ indicate whether state s is one of the searched positions, and define the rollout policy $\mu(\cdot \mid s) = \pi_h(\cdot \mid s)$ when $d(s) = 1$, and $\mu(\cdot \mid s) = \pi_\theta(\cdot \mid s)$ otherwise. At searched states, Eq.* (6) *gives $\mathbb{E}_{a \sim \mu}[Q^{\pi_\theta}(s, a)] \geq V^{\pi_\theta}(s)$; at unsearched states the*

same relation holds with equality. Hence, for the prompt state s_0, the finite-horizon performance-difference identity yields

$$V^{\mu}(s_0) - V^{\pi_\theta}(s_0) = \mathbb{E}_{\tau\sim\mu}\left[\sum_t \big(Q^{\pi_\theta}(s_t, a_t) - V^{\pi_\theta}(s_t)\big)\right] \geq 0. \tag{8}$$

Therefore, applying the improved decision rule only at selected boundaries still improves the expected verifier reward of the complete trajectory. With estimated values and at most K searched boundaries, the corresponding bound is $V^{\widehat{\mu}}(s_0) \geq V^{\pi_\theta}(s_0) - 2K\epsilon$.

Connection to our search. The Gumbel-Max trick samples Eq. (4) through $i^\star = \arg\max_i\{g_i + h(z_i)\}$, where each g_i is an independent standard Gumbel variable. The candidate prior in Eq. (2) is uniform because the root proposals have already been sampled from π_θ. For $\lambda_g > 0$ the corresponding transform is $h(z) = \beta z/\lambda_g$, with $\beta = (c_{\text{visit}} + \max_j N_j)c_{\text{scale}}\lambda_V$; $\lambda_g = 0$ is the greedy limit. Sequential halving is the finite-budget procedure for identifying this maximizer. The exact guarantee therefore applies when the backed-up values are exact and sequential halving recovers the maximizer; Eq. (7) quantifies degradation under value error. This result concerns the rollout policy induced by search. It does not assert that any single finite GRPO or distillation step monotonically improves the parameterized language model.

Commitment. After the final phase, search retains the highest-scoring evaluated continuation. Its tokens or executable action, together with the policy log-probabilities, are appended to the trajectory. We also record the resulting decision boundary for constructing the InfoSFT mask.

Cost. A search call evaluates multiple candidate branches but retains only the selected continuation. Its computation is controlled by the number of candidates C, simulations S, and searched decisions K; we study all three parameters in Table 2.

4.2 Learning from search

Group-relative term. The terminal verifier scores both direct and search-assisted trajectories, and all of them enter Eq. (1) over their complete response masks. The mixed group allows the relative objective to compare ordinary policy behavior with decisions selected by search. Because searched tokens come from the improved rollout policy rather than direct sampling from π_θ, the objective is defined on the collected mixed rollout distribution. We do not apply a search-selection importance correction; the complementary alignment role of InfoSFT is described in Remark 4.2.

Informative SearchSFT. Group-relative learning disappears when all trajectories receive the same return. InfoSFT complements it with token-level supervision on searched decisions for which the value model provides evidence of improvement. Here *informative* refers to selecting learning targets using this improvement evidence. For each committed searched decision, define

$$\Delta_{\text{gain}} = V_{\text{post}} - V_\phi(s), \qquad \Delta_{\text{margin}} = V_{\text{post}} - \max_{c\in\{1..C\}} V_\phi(T(s, a_c)), \tag{9}$$

where $V_{\text{post}} = Q(a^\star)$ is the backed-up value of the selected continuation. The gain compares this value with the state before search, while the margin compares it with the strongest one-step root proposal. A committed decision becomes a distillation target when either $\Delta_{\text{gain}} > \delta_{\text{gain}}$ or $\Delta_{\text{margin}} > \delta_{\text{margin}}$. A policy-confidence filter may remove targets that the policy already assigns high probability; this filter affects supervision only, not root proposal generation. The target rule can use all trajectories with a separate failed-trajectory weight w_{fail}, or the success-only setting used in the reported mathematical reasoning run.

Let $\mathcal{M}_i \subseteq \{1..T\}$ be the token positions belonging to selected searched decisions in trajectory i. Then

$$\mathcal{L}_{\text{InfoSFT}}(\theta) = -\frac{1}{Z_{\text{I}}}\sum_i w_i \sum_{t\in\mathcal{M}_i} \log \pi_\theta\left(y_t^{(i)} \mid x, y_{<t}^{(i)}\right), \qquad Z_{\text{I}} = \sum_i w_i|\mathcal{M}_i|. \tag{10}$$

If no decision passes the gate, InfoSFT contributes exactly zero gradient.

Remark 4.2 (Implicit policy–search alignment)**.** *Let $q_{\text{I}}(\cdot \mid s)$ denote the empirical distribution of searched decisions accepted by the InfoSFT gate. InfoSFT treats this distribution as a distillation target and minimizes $D_{\text{KL}}(q_{\text{I}}\|\pi_\theta)$ on accepted decisions. Because the objective is supervised*

Algorithm 1 APIVIS rollout for one trajectory

Require: task x; rollout mode $z \in \{\text{DIRECT}, \text{SEARCHED}\}$; policy π_θ; frozen value V_ϕ; candidates C; simulations S; search cap K; domain termination rule

$s \leftarrow x$; $c \leftarrow 0$; $\mathcal{E} \leftarrow \emptyset$
while not terminated under the domain rule **do**
  if $z = \text{SEARCHED}$ **and** $c < K$ **then**
    $v \leftarrow V_\phi(s)$
    $a_{1..C} \sim \pi_\theta(\cdot|s)$; draw $g_{1..C}$; build root children
    for $\text{sim} = 1$ **to** S **do**
      leaf, path $\leftarrow$ DESCEND(current root child)
      $v_\ell \leftarrow V_\phi$(leaf); EXPAND(leaf) with C candidates
      BACKUP(path,v_ℓ)
      on phase end: rank by Eq. (2), keep top half
    end for
    $a^\star \leftarrow$ top surviving root; $\alpha \leftarrow$ its principal path
    record $(s, v, \max_c V_\phi(T(s, a_c)), Q(a^\star))$ in $\mathcal{E}$; $c \leftarrow c + 1$
  else
    $\alpha \leftarrow$ one domain action sampled from $\pi_\theta(\cdot|s)$
  end if
  append α to the trajectory; $s \leftarrow T(s, \alpha)$
end while
$r \leftarrow \text{verifier}(x, s)$; select $\mathcal{M} \subseteq \mathcal{E}$ by Eq. (9)
return trajectory, r, searched-state boundaries, InfoSFT mask $\mathcal{M}$

distillation rather than an estimate of a direct-policy expectation, it requires no search-selection importance ratio. Since subsequent root proposals are drawn from the updated policy, the update also reduces the policy–search gap:

$$q_{\text{search}} \textit{ selects } a^\star \;\rightarrow\; \pi_\theta(a^\star \mid s) \uparrow \rightarrow\; D_{\text{KL}}(q_{\text{I}} \| \pi_\theta) \downarrow .$$

This is an implicit alignment mechanism, not a monotonic cross-iteration guarantee under changing search targets and value errors.

Total objective. We set the total loss as

$$\mathcal{L}(\theta) \;=\; w_{\text{G}}\, \mathcal{L}_{\text{GRPO}} \;+\; w_{\text{I}}\, \mathcal{L}_{\text{InfoSFT}}, \qquad w_{\text{G}} = 1. \tag{11}$$

Setting $w_{\text{I}} = 0$ retains search-based rollout generation without distillation. Comparing this setting with $w_{\text{I}} > 0$ isolates the effect of InfoSFT because the search operator, sampler, searched positions, data order, and evaluation protocol remain unchanged.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP ACROSS DOMAINS

To isolate the effect of training guided by search, all methods within each domain use the same task order, number of optimizer steps, and number of initial rollout trajectories per step. Methods that use search may generate or execute additional candidate branches, and we report this additional computation separately. All trained policies are evaluated directly without using search during inference. For the completed mathematical and multi-turn studies, validation metrics first identify a small checkpoint shortlist for each run; we then report the strongest locked full evaluation among those checkpoints.

Mathematical reasoning. The mathematical reasoning study trains Qwen3-1.7B-Base and Qwen3-4B-Base for 100 steps using the fixed ordering of the Open-RS training prompts (Dang & Ngo, 2025), with 64 prompts per step and six completions per prompt. For APIVIS, each group contains three directly sampled responses and three responses generated through search. The frozen VGS value model (Wang et al., 2025) guides the search. We compare the initial policy, GRPO, LATR (Xing et al., 2025), TS-LLM (Feng et al., 2023), and APIVIS with search applied to the first K reasoning chunks.

Table 1: Avg@16 verifier accuracy (%) across Qwen3 model scales. For each run, we report the strongest full evaluation among its validation-shortlisted checkpoints. Evaluation uses direct policy decoding without inference-time search.

Method	MATH-500	AIME-24	AIME-25	Olympiad	Average
Qwen3-1.7B-Base	47.60	4.38	2.29	18.61	18.22
GRPO	63.79	7.08	3.75	27.08	25.42
LATR	63.44	5.42	3.33	27.25	24.86
TS-LLM	64.31	8.75	3.33	28.20	26.15
APIVIS + InfoSFT	**66.18**	**9.58**	**4.58**	**30.80**	**27.79**
Qwen3-4B-Base	45.44	4.58	4.17	21.68	18.97
GRPO	78.83	13.75	10.62	44.07	36.82
LATR	79.74	12.29	10.83	43.29	36.54
TS-LLM	77.39	13.75	11.04	42.42	36.15
APIVIS + InfoSFT	**81.15**	**16.46**	**14.79**	**47.66**	**40.02**

For each problem in MATH-500 (Hendrycks et al., 2021), AIME-2024, AIME-2025, and Olympiad-Bench (He et al., 2024), we draw 16 samples at temperature 0.7 and top-$p = 0.95$. Avg@16 denotes the mean verifier accuracy across these samples. Every shortlisted checkpoint is evaluated on the same 1,234-problem manifest.

Multi-turn interaction. We evaluate ALFWorld household-interaction tasks (Shridhar et al., 2021) under the AgentBoard evaluation framework (Ma et al., 2024), and WebShop's web-interaction tasks (Yao et al., 2022) in its native environment. Every method starts from `Qwen/Qwen3-4B-Instruct-2507` and uses the `verl-agent` pipeline. The comparison includes the initial policy, GRPO, GiGPO (Feng et al., 2025), TS-LLM's MCTS-α search operator with GRPO (Feng et al., 2023), APIVIS with GRPO and InfoSFT, and APIVIS with GiGPO and InfoSFT. Search variants use a frozen agent reward model to score candidate states.

Each searched decision consists of model reasoning followed by one executable agent action. All candidate actions are executed from the same environment snapshot, and only the selected successor continues the trajectory. Search is applied to the first K interaction decisions of each searched trajectory. The ALFWorld full evaluation contains all 134 `valid_unseen` tasks, uses one direct-policy rollout per task at horizon 50, and reports overall and task-type success rates. The WebShop full evaluation uses the training-matched 1,000-product catalog, all 500 test goals, horizon 15, and one direct-policy rollout per goal; it reports task score and success rate.

Baselines and implementation. Within each domain, search variants use the same underlying GRPO or GiGPO learner as the corresponding non-search baseline and follow the same rule for constructing training data. Training retains only the final committed trajectory, including decisions selected by search, and excludes all discarded candidate branches. Environment execution, value scoring, and decision boundaries are adapted to each domain. Full configurations are provided in Appendix B.

5.2 Mathematical Reasoning Results

Table 1 presents results at two Qwen3 model scales. APIVIS with InfoSFT obtains the highest Avg@16 on every benchmark for both models. With Qwen3-1.7B, its average of 27.79 exceeds GRPO, LATR, and TS-LLM by 2.37, 2.93, and 1.64 points, respectively. With Qwen3-4B, its average of 40.02 exceeds GRPO, LATR, and TS-LLM by 3.20, 3.48, and 3.87 points.

LATR does not improve upon GRPO at either scale: its averages are 24.86 versus 25.42 with Qwen3-1.7B and 36.54 versus 36.82 with Qwen3-4B. The consistent APIVIS gains indicate that generating different trajectories is insufficient by itself; the rollout operator must preferentially construct decisions that form better learning targets. Appendix Figures 2 and 3 report the corresponding validation, entropy, response-length, and PPO-KL trajectories.

Search and learning ablations. Table 2 isolates the contributions of the search operator, InfoSFT, and search allocation. At the same C4/S8/K10 budget, replacing MCTS-α with the Gumbel operator increases average Avg@16 from 36.29 to 39.32, and adding InfoSFT further improves it to 40.02. Performance is non-monotonic in the number of searched chunks: $K = 5$, 10, and 20 obtain 39.20, 40.02, and 39.84, respectively. Increasing the per-decision search budget from C4/S8 to C8/S16 at $K = 10$ yields 40.06, only 0.04 points above C4/S8. The result suggests that how search constructs and transfers learning targets matters more than simply increasing search computation.

Table 2: Qwen3-4B mathematical reasoning ablations (Avg@16, %). C, S, and K denote candidates per expansion, simulations per searched decision, and the maximum number of searched chunks, respectively.

Search operator	InfoSFT	C	S	K	MATH-500	AIME-24	AIME-25	Olympiad	Average
MCTS-α	No	4	8	10	78.55	12.92	10.62	43.06	36.29
Gumbel	No	4	8	10	80.70	16.25	13.75	46.60	39.32
Gumbel	Yes	4	8	5	81.04	15.42	14.38	45.98	39.20
Gumbel	Yes	4	8	10	**81.15**	16.46	14.79	47.66	40.02
Gumbel	Yes	4	8	20	80.39	**18.75**	12.50	**47.72**	39.84
Gumbel	Yes	8	16	10	80.93	16.88	**15.00**	47.45	**40.06**

Table 3: Multi-turn agent performance without inference-time search. Entries are mean $\pm$ sample standard deviation across three runs; the base policy has one evaluation. ALFWorld reports success rates (%), and WebShop reports task score and success rate (%).

	WebShop		ALFWorld `valid_unseen`						
Method	Task Score	Success	Overall	Pick	Look	Clean	Heat	Cool	Pick2
Qwen3-4B-Instruct	40.58	15.60	27.61	37.50	66.67	16.13	13.04	14.29	29.41
GRPO	80.59±4.53	68.40±7.75	74.88±9.45	61.11±4.81	94.44±5.56	63.44±28.18	81.16±10.04	82.54±11.98	76.47±5.88
GiGPO	83.37±1.34	73.00±4.70	91.29±1.72	80.56±10.49	**100.00±0.00**	**90.32±3.23**	**94.20±2.51**	92.06±7.27	**94.12±5.88**
MCTS-α + GRPO	85.17±0.47	74.10±4.38	84.08±0.43	77.78±2.40	96.30±6.42	83.87±0.00	81.16±6.65	80.95±4.76	88.24±5.88
APIVIS + GRPO + InfoSFT	**85.43±0.28**	75.20±2.55	88.81±1.97	76.39±2.40	96.30±6.42	89.25±1.86	89.86±10.04	96.83±2.75	86.27±3.39
APIVIS + GiGPO + InfoSFT	84.69±2.08	**76.27±2.66**	**91.79±1.49**	**81.94±6.36**	**100.00±0.00**	**90.32±3.23**	91.30±4.35	**98.41±2.75**	92.16±6.80

5.3 Multi-turn Agentic Reasoning

The multi-turn study uses Qwen3-4B-Instruct for every method and evaluates the trained policies without search at inference time. Table 3 reports the unified ALFWorld `valid_unseen` evaluation over 134 tasks. Each trained method reports best-checkpoint results from three seeds; the base policy has one evaluation. APIVIS with GRPO and InfoSFT improves overall success from 74.88% to 88.81%, while APIVIS with GiGPO and InfoSFT reaches 91.79%, slightly exceeding GiGPO's 91.29%. MCTS-α with GRPO reaches 84.08%, below the value-monotone APIVIS search with the same learner.

Across ALFWorld task types, APIVIS with GiGPO and InfoSFT obtains the highest or tied highest result on Pick, Look, Clean, and Cool, whereas GiGPO remains strongest on Heat and Pick2.

The WebShop evaluation uses the training-matched 1,000-product catalog and all 500 test goals. APIVIS with GRPO and InfoSFT obtains the best aggregate task score, 85.43, while APIVIS with GiGPO and InfoSFT obtains the highest success rate, 76.27%. These results improve over GRPO's 80.59 task score and 68.40% success rate. MCTS-α with GRPO reaches 85.17 and 74.10%. Standard deviations summarize run-to-run variation. The validation and optimization trajectories for ALFWorld and WebShop are reported in Appendix Figures 4 and 5, respectively.

Training dynamics. Across ALFWorld and WebShop, APIVIS runs generally exhibit increasing entropy and response length through middle-to-late training, while validation improves and PPO KL remains controlled. The mathematical runs likewise expand response length, although their entropy is non-monotone and scale dependent. Together, Figures 2–5 are consistent with productive exploration rather than optimization collapse.

6 Conclusion

We propose APIVIS, which uses value-guided search as a policy-improvement operator within RLVR. It combines mixed direct/searched groups with InfoSFT to preserve learning when group-relative rewards are uninformative. We prove statewise and trajectory-level improvement under exact values and bound degradation under value error. Across mathematical reasoning, ALFWorld, and WebShop, the directly decoded policies outperform GRPO, GiGPO, and competitive search-based baselines. The main limitations are fixed search placement and frozen value models; stable policy–value co-evolution and learned search allocation are important future directions discussed in Appendix D.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementation and debugging, feedback on experimental protocols, mathematical exposition and proof checking, result analysis and presentation, manuscript editing, figure preparation, literature search, and reference checking. The authors determined the research questions, method, experimental protocol, and conclusions; ran the experiments; and reviewed and verified all AI-assisted code, derivations, citations, numerical results, figures, and text. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Section 4 and Algorithm 1 specify the search and learning procedure. Appendix A gives the full proof of the policy-improvement result, while Appendix B reports the models, data, optimization settings, search hyperparameters, prompts, checkpoint-selection rules, and evaluation harnesses used in each domain. Appendix C provides the corresponding training dynamics. All reported final evaluations use direct policy decoding without inference-time search.

REFERENCES

Quy-Anh Dang and Chris Ngo. Reinforcement learning for reasoning in small llms: What works and what doesn't. *arXiv preprint arXiv:2503.16219*, 2025. URL https://arxiv.org/abs/2503.16219.

Ivo Danihelka, Arthur Guez, Julian Schrittwieser, and David Silver. Policy improvement by planning with gumbel. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=bERaNdoegnO.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

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 (NeurIPS)*, 2025. URL https://arxiv.org/abs/2505.10978.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023. URL https://arxiv.org/abs/2309.17179.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*, 2025. URL https://arxiv.org/abs/2501.04519.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. URL https://arxiv.org/abs/2402.14008.

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 *NeurIPS Datasets and Benchmarks Track*, 2021. URL https://arxiv.org/abs/2103.03874.

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International Conference on Machine Learning (ICML)*, pp. 3499–3508, 2019. URL https://proceedings.mlr.press/v97/kool19a.html.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024. URL https://arxiv.org/abs/2411.15124.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2401.13178.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, 2020. doi: 10.1038/s41586-020-03051-4. URL https://www.nature.com/articles/s41586-020-03051-4.

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. URL https://arxiv.org/abs/2402.03300.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2010.03768.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404. URL https://www.science.org/doi/10.1126/science.aar6404.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. URL https://arxiv.org/abs/2408.03314.

Kaiwen Wang, Jin Peng Zhou, Jonathan Chang, Zhaolin Gao, Nathan Kallus, Kianté Brantley, and Wen Sun. Value-guided search for efficient chain-of-thought reasoning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://arxiv.org/abs/2505.17373.

Yu Xia, Jingru Fan, Weize Chen, Siyu Yan, Xin Cong, Zhong Zhang, Yaxi Lu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. AgentRM: Enhancing agent generalization with reward modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 19277–19290, 2025. doi: 10.18653/v1/2025.acl-long.945. URL https://aclanthology.org/2025.acl-long.945/.

Shangyu Xing, Siyuan Wang, Chenyuan Yang, Xinyu Dai, and Xiang Ren. Lookahead tree-based rollouts for enhanced trajectory-level exploration in reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2510.24302*, 2025. URL https://arxiv.org/abs/2510.24302.

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 (NeurIPS)*, 2022. URL https://arxiv.org/abs/2207.01206.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2305.10601.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025. URL https://arxiv.org/abs/2504.13837.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2406.03816.

Ziqi Zhao, Zhaochun Ren, Jiahong Zou, Liu Yang, Zhiwei Xu, Xuri Ge, Zhumin Chen, Xinyu Ma, Daiting Shi, Shuaiqiang Wang, Dawei Yin, and Xin Xin. Reinforced efficient reasoning via semantically diverse exploration. *arXiv preprint arXiv:2601.05053*, 2026. URL https://arxiv.org/abs/2601.05053.

A Proof of the chunk-level policy-improvement guarantee

Proof of Theorem 4.1. Condition on one fixed candidate list $\mathcal{A}_s = (a_1, \dots, a_C)$ and its values $\mathbf{q} = (q_1, \dots, q_C)$. Let $\rho_0(i \mid \mathcal{A}_s) = 1/C$ be the uniform baseline distribution over the C candidate slots, and let the random variable q take value q_i when $i \sim \rho_0$. The search distribution $\rho_h^{\mathbf{q}}$ in Eq. (4) is precisely ρ_0 reweighted by $\exp(h(q_i))$. Therefore,

$$\begin{aligned}\mathbb{E}_{\rho_h^{\mathbf{q}}}[q] &= \frac{\mathbb{E}_{\rho_0}[q\,\exp(h(q))]}{\mathbb{E}_{\rho_0}[\exp(h(q))]} \\ &= \mathbb{E}_{\rho_0}[q] + \frac{\mathrm{Cov}_{\rho_0}(q, \exp(h(q)))}{\mathbb{E}_{\rho_0}[\exp(h(q))]} \;\geq\; \mathbb{E}_{\rho_0}[q]. \end{aligned} \tag{12}$$

The final inequality holds because h is non-decreasing. Specifically, if q' is an independent copy of q under the same uniform distribution ρ_0, then

$$2\,\mathrm{Cov}(q, \exp(h(q))) = \mathbb{E}\big[(q - q')\big(\exp(h(q)) - \exp(h(q'))\big)\big] \geq 0.$$

This proves Eq. (5). Averaging its two sides over $\mathcal{A}_s \sim \pi_\theta^C$ proves Eq. (6), because a uniformly selected candidate has marginal distribution $\pi_\theta(\cdot \mid s)$. Finally, write $\widehat{q}_i = \widehat{Q}(s, a_i)$ and let $\rho_h^{\widehat{\mathbf{q}}}$ be the search distribution formed from these estimated values. The uniform error bound gives

$$\mathbb{E}_{\rho_h^{\widehat{\mathbf{q}}}}[q] \geq \mathbb{E}_{\rho_h^{\widehat{\mathbf{q}}}}[\widehat{q}] - \epsilon \geq \mathbb{E}_{\rho_0}[\widehat{q}] - \epsilon \geq \mathbb{E}_{\rho_0}[q] - 2\epsilon.$$

Taking expectation over candidate lists proves Eq. (7). □

B Full experimental configuration

Table 4: Mathematical-reasoning configuration at a glance.

Component	Setting
Policy / data	Qwen3-1.7B-Base and Qwen3-4B-Base; Open-RS, 7,000 prompts in fixed order
Training	100 steps; 64 prompts × 6 completions; seed 1
Group composition	3 direct and 3 searched completions
Value model	Frozen `VGS-AI/DeepSeek-VM-1.5B` (Wang et al., 2025); class-1 success probability $\mathrm{softmax}(\mathbf{z})_1$ from its three-class head
Training sampler	temperature 1.0; top-$p = 1.0$
Search	$C = 4$, $S = 8$; first $K = 10$ chunks; principal-path commitment
InfoSFT	weight 0.1; value gain or margin > 0.001; success-only
Optimization	actor lr 10^{-6}; one epoch; DAPO token-mean objective
Selection	validation shortlist followed by locked full evaluation
Full evaluation	Avg@16; temperature 0.7; top-$p = 0.95$; max 8,192 response tokens

Table 5: Multi-turn interaction configuration at a glance.

Component	Setting
Policy	`Qwen/Qwen3-4B-Instruct-2507`
Value model	Frozen AgentRM-3B; scalar regression output clipped to $[0, 1]$
Runtime	`verl-agent` interaction pipeline
Benchmarks	ALFWorld through AgentBoard; WebShop
Comparisons	GRPO; GiGPO (Feng et al., 2025); MCTS-α + GRPO
Search action	Reasoning and one executable environment action
Placement	First K interaction decisions
Control	Matched tasks, optimizer steps, and root trajectories
Evaluation	Direct policy on 134 ALFWorld tasks and 500 WebShop goals

Table 6: Role of APIVIS components across domains.

Domain	Search action	Placement	Value model
Math	reasoning chunk	first K chunks	VGS-1.5B; $\mathrm{softmax}(\mathbf{z})_1$ (class-1 success probability)
ALFWorld / WebShop	reasoning + environment action	first K decisions	AgentRM-3B; scalar regression output clipped to $[0, 1]$

B.1 Audited training and harness details

The following settings are taken from the launch configurations, run manifests, and evaluation artifacts used for Tables 1–3. Hardware-only microbatching, tensor parallelism, and memory fractions may vary across GPU types; they change execution granularity but not the effective optimizer batch, rollout distribution, or evaluation protocol.

Compute usage. Each reported training run uses one A100 node. Mathematical reasoning uses eight GPUs for policy optimization and validation. The ALFWorld GRPO/GiGPO runs use four learner GPUs, while search variants use eight GPUs to host the policy and colocated AgentRM replicas. WebShop uses either four A100-80GB GPUs or eight A100-40GB GPUs, depending on the execution profile; the effective optimizer batch and rollout protocol are unchanged across these profiles. Frozen value-model inference is colocated with training whenever memory permits. Full evaluations use direct policy decoding without search or value-model inference.

B.1.1 Mathematical reasoning

Table 7: Detailed mathematical-reasoning hyperparameters. Model, data, and value-model choices are summarized in Tables 4 and 6.

Hyperparameter	Value
Optimizer steps	100
Prompts / completions per prompt	64 / 6 (384 completions per step)
Training seed	1
Optimizer	AdamW; learning rate 10^{-6}; weight decay 0.01
PPO schedule	One epoch; 16-prompt minibatches; 5% linear warmup, then constant
Loss aggregation / advantages	DAPO token mean / normalized over six responses
PPO clipping	$(\epsilon_{\mathrm{low}}, \epsilon_{\mathrm{high}}) = (0.2, 0.3)$; dual clip 3.0
Entropy / explicit KL coefficients	0 / disabled
Training decoding	Temperature 1.0; top-$p = 1.0$; top-k disabled
Prompt / completion limits	512 / 3,584 tokens
Direct / searched responses	3 / 3
Search $C/S/K$	$4/8/10$
InfoSFT	Weight 0.1; gain or margin > 0.001; failed-trajectory weight 0
Validation	Every five steps; Avg@4; temperature 0.7; top-$p = 0.95$; seed 0
Full evaluation	1,234 problems; Avg@16; temperature 0.7; top-$p = 0.95$; seed 0
TS-LLM MCTS-α	$C = 4$, $S = 8$, $K = 10$; $c_{\mathrm{base}} = 19652$; $c_{\mathrm{init}} = 3$; discount 1; no root noise

Protocol details. Open-RS provides 7,000 prompts in a fixed, unshuffled order. Binary correctness is computed by `math_verify`, with the shared DAPO verifier as fallback. LATR uses its pinned key-token branching implementation at revision `2adb0edf`; TS-LLM uses the MCTS-α settings listed above.

Math policy prompt. Training, validation, and full evaluation use the same system prompt; the raw problem is the sole user message:

> A conversation between User and Assistant. The User asks a math problem, and the Assistant solves it. The Assistant should first reason through the problem step by step and then provide the final answer. Enclose the reasoning process within `<think>` and `</think>` tags, and enclose the final answer within `<answer>` and `</answer>` tags. Put the final mathematical answer inside `\boxed{}`. Respond in English only. The response format is `<think> step-by-step reasoning </think> <answer> \boxed{final answer} </answer>`.

B.1.2 ALFWORLD

Table 8: Detailed ALFWorld hyperparameters. Shared model and component choices are summarized in Tables 5 and 6.

Hyperparameter	Value
Optimizer steps	100
Task prompts / trajectories per prompt	16 / 8 (128 episodes per step)
PPO minibatch size / learning rate	64 / 10^{-6}
Validation and checkpoint interval	5 steps
Loss aggregation / explicit KL	Token mean / disabled
Advantage normalization	Group standard deviation
Invalid-action penalty	0.1
GiGPO	$\gamma = 0.95$; step weight 1.0; exact state grouping
Training decoding	Temperature 1.0; top-$p = 1.0$
Horizon / visible history	50 actions / five turns
Prompt / per-turn completion limits	2,048 / 256 or 512 tokens by campaign
Direct / searched trajectories	4 / 4
Search $C/S/K$	$4/8/10$
Search scoring	Gumbel scale 0; value coefficient 1; $c_{\text{visit}} = 50$; $c_{\text{scale}} = 1$
InfoSFT	Weight 0.1; terminal success; $\Delta V > 0$
Training validation	64 episodes; temperature 0.4; top-$p = 1.0$
Held-out evaluation	134 `valid_unseen` tasks; seed 0; temperature 0.4; top-$p = 1.0$

Protocol details. Runs use `verl-agent` revision `20bd331b` and frozen AgentRM-3B revision `8381504a`. Validation first ranks checkpoints by `val/alfworld/test_score`, then by success rate, with the earlier step breaking any remaining tie.

ALFWorld policy prompt. Training, validation, and the held-out harness use the same policy template. At each turn the policy receives the following dynamically instantiated user message, with one admissible action per line:

> `You are controlling an agent in ALFWorld.`
> `Task: {task instruction}`
> `Previous steps: [Observation i: '...', Action i: '...']`
> (omitted on the first turn)
> `Current observation: {observation}`
> `Allowed actions (choose exactly one): {admissible actions}`
> Give a reason of at most 15 words inside `<reasoning>` and `</reasoning>`, then exactly one allowed action inside `<action>` and `</action>`. Do not output any other text.

Only exact admissible actions are executed. A malformed response or an action outside the supplied list is marked invalid.

ALFWorld AgentRM prompt. AgentRM receives its native trajectory serialization: a fixed household-task instruction that enumerates the `go`, `take`, `put`, `open`, `close`, `toggle`, `clean`,

heat, and cool action forms; an assistant OK; the initial observation and task; alternating Thought:/Action: and Observation: messages for the visible five-turn history; and a final assistant message containing the candidate reason and action. The resulting scalar regression output is the search value. The fixed instruction asks the scorer to interact with a household, plan from the latest observation, and answer in exactly Thought: <thoughts> followed by Action: <next action>; it also explains that Nothing happened denotes an invalid previous action.

B.1.3 WEBSHOP

Table 9: Detailed WebShop hyperparameters. Shared model and component choices are summarized in Tables 5 and 6.

Hyperparameter	Value
Optimizer steps	100
Goals / trajectories per goal	16 / 8 (128 episodes per step)
PPO minibatch size / learning rate	64 / 10^{-6}
Validation interval / policy seeds	5 steps / 0–2
Loss aggregation / explicit KL	Token mean / disabled
Advantage normalization / invalid penalty	Group standard deviation / 0.1
Terminal success reward	10
GiGPO	$\gamma = 0.95$; step weight 1.0; trajectory normalization
Training decoding	Temperature 1.0; top-$p = 1.0$; top-k disabled
Horizon / visible history	15 actions / five turns
Prompt / per-turn completion limits	4,096 / 512 tokens
Direct / searched trajectories	4 / 4
Search $C/S/K$	$4/8/10$
Search scoring	Gumbel scale 0; value coefficient 1; $c_{\text{visit}} = 50$; $c_{\text{scale}} = 1$
InfoSFT	Weight 0.1; terminal success; minimum gain 0.001
Checkpoint selection	Top validation success rate
Full evaluation	500 goals; seed 0; temperature 0.4; top-$p = 1.0$; 128-token response limit

Protocol details. Training uses goals indexed from 500 onward in the 1,000-product catalog; validation and full evaluation use the fixed first 500 goals and matching Lucene index. The structured_history_v2 truncation rule preserves instructions, the goal, current observation, and current action list, then allocates remaining context to the newest history first. A rollout batch is replaced if this protected content alone exceeds the prompt limit.

WebShop policy prompt. Training, validation, and full evaluation use the same action-only template. The policy receives one user message with no additional system message:

> You are an expert autonomous shopping agent in WebShop.
> Shopping goal: {goal}
> {optional exploration state}
> {recent observation/action history}
> Current observation: {formatted observation}
> {already selected options}
> Actions allowed in the current state: [{available actions}]
> Choose exactly one allowed action. Copy click[...] actions exactly; at the search page use a concise query focused on the product and its discriminative attributes; inspect promising products; select all required options before buying; do not repeat an option on an unchanged page; and buy only after checking attributes and the price ceiling. Return exactly one line, either <action>search[query]</action> or <action>click[exact allowed text]</action>, with no reasoning or other text.

Observations use the SkillRise-formatted representation, and click actions are canonicalized against the currently available action list.

WebShop AgentRM prompt. The scorer begins with its fixed "You are web shopping" instruction and the allowed search[keywords] / click[value] action grammar, followed by assistant Ok., the initial WebShop instruction and observation, alternating Thought:/Action: and Observation: messages for the visible history, and the candidate reason/action as the final assistant message.

C TRAINING DYNAMICS

Figures 2–5 report the validation and optimization trajectories used in the main experiments. Each mathematical setting has one run per method: faint traces show logged values, solid lines show centered running means over a 10-step radius, and bands show the local minimum–maximum range within that window. ALFWorld has three runs per method. WebShop has two runs for GRPO, MCTS-α + GRPO, and APIVIS + GRPO + InfoSFT, and three runs for GiGPO and APIVIS + GiGPO + InfoSFT. For both multi-turn domains, lines and bands show the mean and sample standard deviation across runs at their shared steps.

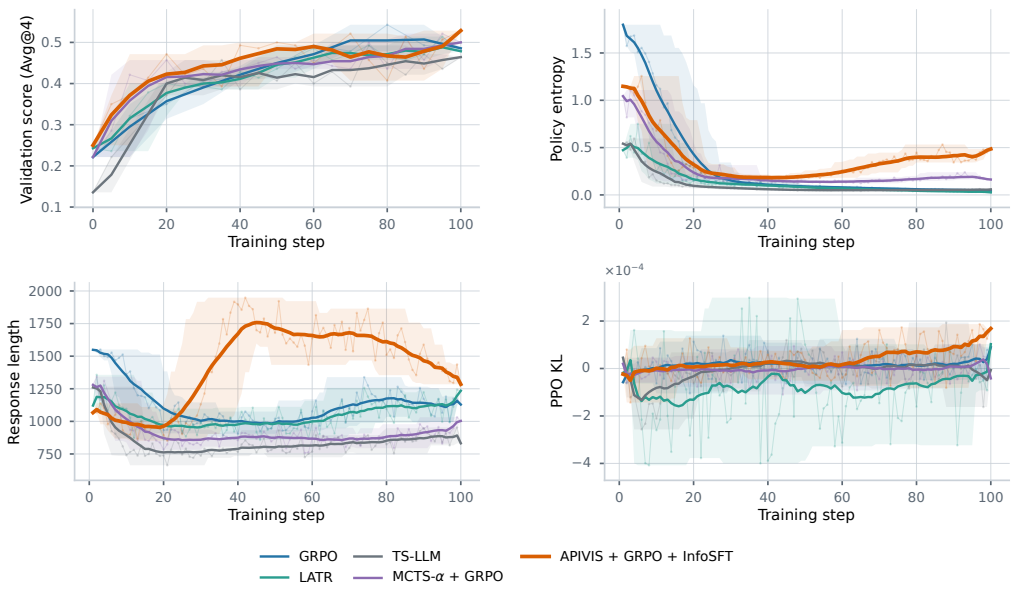

Figure 2: Qwen3-4B-Base mathematical reasoning dynamics, with one run per method. Validation reports Avg@4 on the fixed Open-RS validation set; the remaining panels report training-time entropy, response length, and PPO KL.

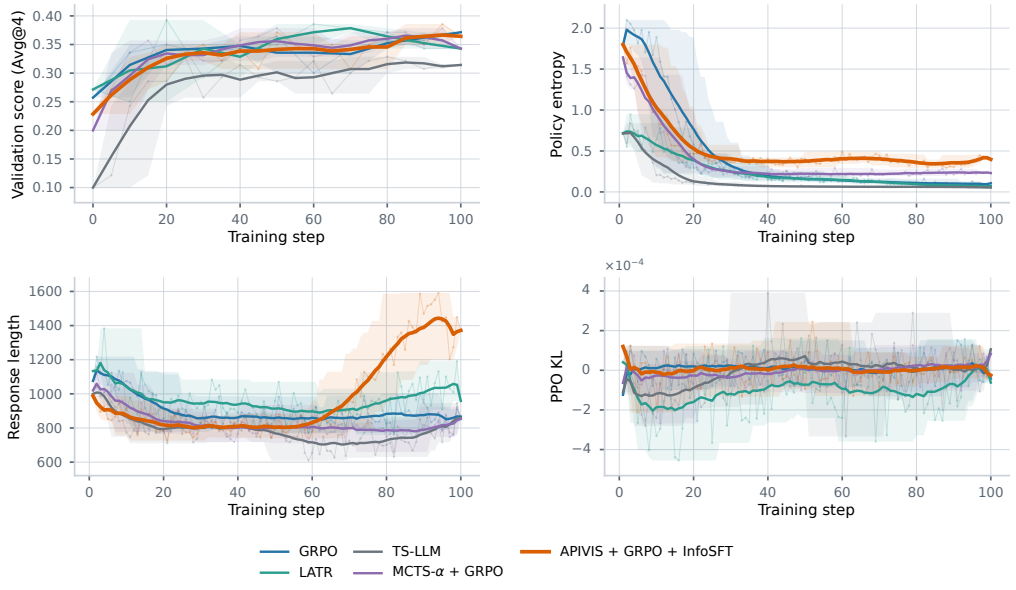

Figure 3: Qwen3-1.7B-Base mathematical reasoning dynamics, with one run per method, under the protocol of Figure 2.

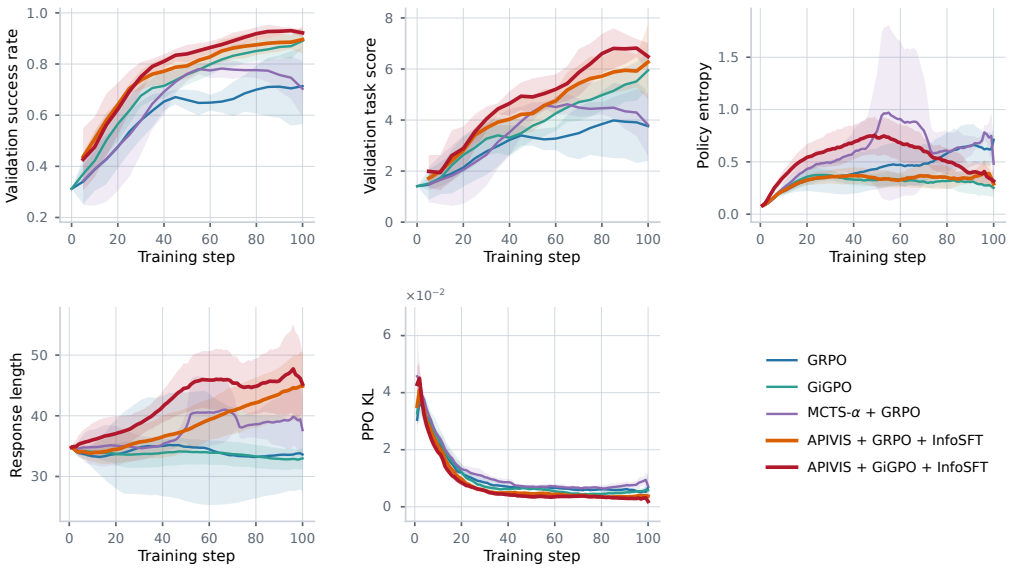

Figure 4: ALFWorld dynamics across three runs per method. Lines and bands show the cross-run mean and sample standard deviation for validation success rate, AgentBoard test score, policy entropy, response length, and PPO KL.

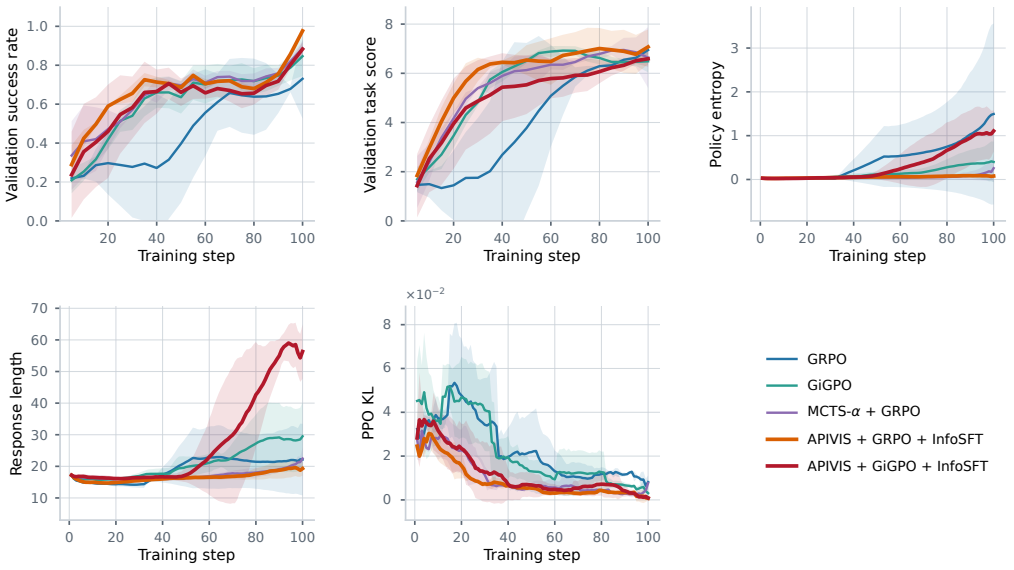

Figure 5: WebShop dynamics across two or three runs per method, as specified in the text. Lines and bands show the cross-run mean and sample standard deviation. Validation success points are matched to validation task-score events by step and timestamp, excluding training-rollout success measurements.

D Discussion and Limitations

Shared structure across domains. The claim that APIVIS applies across domains rests on a common procedure for search and learning. In every domain, the method samples candidate decisions from the current policy, ranks them by their estimated continuation values, backs up those values, retains the selected branch, mixes trajectories generated with and without search during GRPO training, and optionally distills the selected decisions. The action boundaries, state transitions, verifiers, and value models vary by domain. Mathematical reasoning operates on text chunks and uses an answer verifier. ALFWorld and WebShop execute actions in interactive environments.

Search placement is fixed. The current method applies search to the first K decisions of each searched trajectory. This rule is shared across domains but may spend computation on decisions for which direct sampling is already sufficient. Learning to allocate search calls dynamically across reasoning chunks and environment actions is an important direction for future work.

Joint value learning remains open. We do not require the frozen value model to remain globally calibrated under the evolving policy distribution. Search only relies on its local ranking ability among candidate continuations generated from the current policy. Freezing the scorer is therefore a controlled way to isolate the rollout operator, provided that this local ordering remains informative.

Jointly updating the policy and value model could adapt the scorer to newly visited states, but it is not a minor extension of the present study. Reliable value learning requires a dedicated training pipeline and substantial trajectory data, as illustrated by VGS and AgentRM (Wang et al., 2025; Xia et al., 2025). Joint training would additionally introduce a potentially unstable feedback loop: policy updates change the value model's input distribution, value errors alter the trajectories selected by search, and distillation can reinforce those errors in the policy. We therefore leave value training outside the scope of this work. A central future direction is a stable iterative procedure in which policy and value models co-evolve while controlling critic drift and self-reinforcing search errors.