VEG: Verbal ϵ-greedy for Semantic Exploration in Multi-Turn RL Agents

Yongchang Hao[1,2] **Jie Hao**[1*] **Yongsheng Mei**[1] **Ze Ye**[1]
Junyi Chai[1] **Bin Guo**[1] **Benjamin Yao**[1] **Chenlei Guo**[1] **Lili Mou**[2]
[1] Amazon
[2] University of Alberta

Abstract

Reinforcement learning (RL) has become a cornerstone of the post-training pipeline for large language models (LLMs), enabling capabilities such as complex reasoning and tool use. However, standard RL approaches face significant challenges due to reward sparsity. Moreover, LLMs typically exhibit mode-seeking behavior, concentrating probability mass on high-likelihood regions. This lack of diversity biases the model toward premature exploitation, hindering the exploration necessary for optimal learning. To address this, we propose VEG (verbal ϵ-greedy), a novel framework that leverages external feedback as a dynamic control variable to explicitly balance exploration and exploitation within the semantic space. This method not only supplements sparse final rewards with intermediate signals but also enforces sustained exploration throughout the training process. Experiments on τ^2-bench and the agentic search QA benchmark demonstrate that our method achieves superior accuracy compared to standard RL baselines. Notably, the trained policy eventually outperforms the external feedback model itself, demonstrating that VEG enables the agent to effectively filter and improve upon the guidance it receives.

1 Introduction

Reinforcement learning has become a key part of the post-training pipeline for LLMs. It enables sophisticated capabilities such as complex reasoning, code generation, and tool use (Ouyang et al., 2022; Rafailov et al., 2023; Lightman et al., 2023). However, applying RL to multi-turn, tool-using scenarios introduces two fundamental challenges that hinder learning.

First, these environments typically provide only sparse, terminal rewards. Success signals arrive only after the agent executes multiple reasoning steps and tool interactions. This sparsity makes credit assignment difficult. For example, an agent might receive a binary success indicator after a 5-turn dialog involving multiple search queries and reasoning chains. Determining which specific actions contributed to success or failure becomes a non-trivial problem. Standard RL algorithms struggle in such settings and often require extensive exploration to find reward-yielding trajectories.

Second, LLMs inherently exhibit mode-seeking behavior during generation. They concentrate probability mass on high-likelihood token sequences (Holtzman et al., 2019). When combined with standard RL training, this tendency amplifies exploitation. The policy often converges prematurely on suboptimal strategies before it adequately explores the action space. Existing approaches attempt to encourage exploration through token-level perturbations, such as temperature sampling or entropy regularization in PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024). However, these methods introduce stochasticity only at the syntactic level. Adding noise to token logits typically produces minor lexical variations or incoherent outputs. It rarely leads to the semantically meaningful exploration of alternative reasoning strategies, tool choices, or execution plans.

To address these challenges, we propose **verbal ϵ-greedy (VEG)**. This framework performs exploration-exploitation control in the semantic space by using external feedback as a dynamic guidance signal. Unlike token-level perturbations, VEG randomizes between two distinct types of feedback prompts. Exploitation prompts refine the current strategy, while exploration prompts suggest alternative approaches, tool selections, or verification methods. This feedback comes from an external model and acts as state augmentation rather than reward shaping. The feedback becomes part of the agent's observation. This naturally guides the policy toward semantically diverse behaviors while preserving the original sparse reward struc-

*Corresponding author: haoj8711@gmail.com

ture. VEG offers two key benefits. First, it mitigates reward sparsity by providing intermediate guidance signals throughout the episode. This provides structured intermediate guidance that improves credit assignment, without altering the task reward. Second, it enforces sustained semantic exploration. By systematically injecting alternative hypotheses and strategies into the agent's context, it prevents premature convergence to local optima. Importantly, VEG is algorithm-agnostic and can integrate into any standard RL pipeline as a drop-in module for trajectory collection.

We evaluate VEG on two challenging multi-turn agent benchmarks. Search QA requires interleaved reasoning and search tool use to answer complex questions, such as those requiring multi-hop reasoning. τ^2-bench is a customer-support environment with dual-control dynamics where both the agent and user can act and call tools. Our experiments demonstrate that VEG significantly outperforms standard RL baselines on both benchmarks, confirming the effectiveness of our method.

2 Approach

2.1 Reinforcement learning for language models

Markov decision process. We formulate the problem as an episodic MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, H)$. At step t, the agent observes state $s_t \in \mathcal{S}$ and selects action $a_t \in \mathcal{A}$ via policy $\pi(a_t|s_t)$. The environment transitions to $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ yielding reward $r_t = \mathcal{R}(s_t, a_t)$. The goal is to maximize the expected cumulative reward $J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{H-1} r_t]$.

LLM Generation as MDP. We adapt the general MDP framework to LLMs by formalizing the interaction at the turn level. In this formulation, the state s_t represents the complete context window, encompassing system instructions and the entire dialogue history up to turn t. The action a_t is defined as the model's full natural language response (including any reasoning chains or tool calls) rather than a single token. Conventionally, approaches like RLHF (Ouyang et al., 2022) or RLVR (DeepSeek-AI, 2025) treat it as a single-turn problem with a horizon of $H = 1$. This reduces the MDP to a contextual bandit setting, where the episode ends immediately after one response. However, complex capabilities such as tool use require a multi-turn setting ($H > 1$), as a current action (e.g., querying a search engine) is performed solely to influence the state for future reasoning. In this setting, the transition dynamics $\mathcal{P}$ become critical and are deterministic: the next state s_{t+1} is constructed by concatenating the current state s_t, the agent's action a_t, and the subsequent environment observation (e.g., the search engine results). A binary reward (either 0 or 1) is assigned upon the termination of the episode after at most H turns.

2.2 Our method

Motivation and insights. A fundamental principle in classical reinforcement learning is the balance between exploration and exploitation, which prevents the agent from focusing too early on suboptimal actions (Sutton and Barto, 2018; Thrun, 1992). In benchmarks like Atari or Mujoco, agents explore by randomizing over *actions* using ϵ-greedy strategies or intrinsic motivation (Bellemare et al., 2016). Modern LLM RL post-training, by contrast, typically randomizes only over *tokens* via temperature sampling or entropy regularization. This lower level of randomization tends to yield lexical variants of the same underlying strategy rather than genuinely different reasoning, tool choices, or execution plans (Jin et al., 2025b). In Search QA, for example, this surfaces as rewording the same failed query; in τ^2-bench, as repeating the same tool-call pattern. Consequently, standard LLM agents often suffer from entropy collapse and fail to discover new solutions in sparse-reward environments (Cui et al., 2025; Jin et al., 2025b). VEG instead operates one level higher, randomizing at the *strategy* level by steering the agent toward qualitatively different plans.

VEG: verbal ϵ-greedy. We propose **verbal ϵ-greedy (VEG)** to solve these problems. Instead of applying randomness to tokens, VEG performs exploration and exploitation control in the semantic space. It uses external feedback as a dynamic control signal.

VEG addresses two common failure modes in multi-turn LLM RL. The first is reward sparsity, which makes credit assignment difficult over long horizons. The second is entropy collapse, which leads to premature exploitation (Cui et al., 2025; Jin et al., 2025b; Zhang et al., 2025b).

We follow the turn-level MDP definition for LLM generation. At step t, the state s_t includes the entire conversation history. The action a_t is the full textual response from the agent. At the end of an episode, the agent receives a sparse re-

ward $r \in \{0, 1\}$ based on its success. VEG uses an external feedback model f_ψ. This model looks at the current context and returns textual guidance for either exploitation or exploration. We use two prompting templates: an exploitation template $\mathcal{T}^{\text{i}}$ and an exploration template $\mathcal{T}^{\text{r}}$. At each step, a controller samples a template based on ϵ:

$$z_t \sim \text{Bernoulli}(\epsilon).$$

The variable ϵ determines the probability of requesting exploratory feedback. The template is selected as follows:

$$\mathcal{T}_t = \begin{cases} \mathcal{T}^{\text{r}}, & z_t = 1 \\ \mathcal{T}^{\text{i}}, & z_t = 0. \end{cases} \tag{1}$$

Based on the selected template, the feedback model generates a verbal message:

$$v_t \sim f_\psi(\cdot \mid s_t, \mathcal{T}_t).$$

This message v_t is a natural language instruction. It either refines the current plan or proposes new strategies and tool choices.

VEG does not use v_t for reward shaping. Instead, it treats the feedback as part of the environment observation. We augment the state by appending v_t to the history:

$$\tilde{s}_t := s_t \oplus v_t.$$

The symbol $\oplus$ represents concatenation. The agent then selects its action using this augmented state:

$$a_t \sim \pi_\theta(\cdot \mid \tilde{s}_t).$$

The environment executes a_t and returns an observation o_{t+1}. This observation is added to create the next state $s_{t+1} = \tilde{s}_t \oplus a_t \oplus o_{t+1}$. The interaction continues until the episode ends.

2.3 Algorithm

We build VEG on top of Group Relative Policy Optimization (GRPO). This is a PPO-style algorithm that does not require a learned critic or value model (Shao et al., 2024). For every training input, GRPO samples a group of G rollouts. It calculates a group-relative advantage by normalizing the rewards within that group. This process uses the group mean and standard deviation. The algorithm then performs a policy update using a PPO-style clip. It also includes a KL penalty compared to a fixed reference policy. This penalty helps prevent the policy from drifting too far (Shao et al., 2024; DeepSeek-AI, 2025).

Algorithm 1 VEG rollouts

Require: policy π_θ; old policy $\pi_{\theta_{\text{old}}}$; reference π_{ref}; env $\mathcal{E}$; feedback f_ψ; templates $\mathcal{T}^{\text{i}}, \mathcal{T}^{\text{r}}$; horizon H; group size G; VEG rate ϵ; PPO clip ε; KL coef. β; $\delta > 0$.

for each initial context s_0 **do**
  for $i = 1, \ldots, G$ **do**
    $s \leftarrow s_0$, done $\leftarrow$ false, $t \leftarrow 0$
    while not done **and** $t < H$ **do**
      $z \sim \text{Bernoulli}(\epsilon)$
      $\mathcal{T} \leftarrow \mathcal{T}^{\text{r}}$ if $z{=}1$ else $\mathcal{T}^{\text{i}}$
      $v \sim f_\psi(\cdot \mid s, \mathcal{T})$
      $\tilde{s} \leftarrow s \oplus v$
      $a \sim \pi_\theta(\cdot \mid \tilde{s})$
      $(o, \text{done}) \leftarrow \mathcal{E}.\text{step}(a)$
      $s \leftarrow \tilde{s} \oplus a \oplus o$
      $t \leftarrow t + 1$
    end while
    $r_i \in \{0, 1\} \leftarrow \mathcal{E}.\text{terminal\_reward}()$
    store tokens y_i
  end for
  $\mu \leftarrow \frac{1}{G} \sum_i r_i$
  $\sigma \leftarrow \sqrt{\frac{1}{G} \sum_i (r_i - \mu)^2}$
  $\hat{A}_i \leftarrow \frac{r_i - \mu}{\sigma + \delta}$
  Update θ by one RL step on $\{(y_i, \hat{A}_i)\}_{i=1}^{G}$ using PPO-style clip ε and KL to π_{ref}.
end for

Algorithm 1 shows the full procedure. We use Claude Sonnet 4 as the feedback model f_ψ (Anthropic, 2025). This model offers a good balance between reasoning power and cost. The two prompting templates, $\mathcal{T}^{\text{i}}$ and $\mathcal{T}^{\text{r}}$, are included in Appendix B. VEG only changes how trajectories are collected during rollouts. The underlying GRPO optimization objective remains the same. Because of this, VEG can work with any standard RL algorithm beyond GRPO. It acts as a simple drop-in module for rollout generation.

3 Experiments

3.1 Setup

Benchmarks. We evaluate VEG on two multi-turn agent benchmarks with sparse rewards. These benchmarks require significant exploration. (1) **Search QA** follows the agentic search setup from prior work (Jin et al., 2025a). In this task, the

Table 1: Results on Search (QA). The best and second-best results are displayed in bold and underlined, respectively. †/∗ represents in-domain/out-domain datasets. The "GRPO" rows quote numbers directly from Jin et al. (2025a), whereas the"GRPO (Our repl.)" rows use our implementation based on SkyRL. Since VEG builds on our implementation, "GRPO (Our repl.)" is the direct ablation baseline.

Methods	NQ†	TriviaQA*	PopQA*	HotpotQA†	2wiki*	Musique*	Bamboogle*	Avg.
Qwen2.5-3B-Instruct								
Direct Inference	0.106	0.288	0.108	0.149	0.244	0.020	0.024	0.134
CoT	0.023	0.032	0.005	0.021	0.021	0.002	0.000	0.015
IRCoT	0.111	0.312	0.200	0.164	0.171	0.067	0.240	0.181
Search-o1	0.238	0.472	0.262	0.221	0.218	0.054	**0.320**	0.255
RAG	0.348	0.544	0.387	0.255	0.226	0.047	0.080	0.270
SFT	0.249	0.292	0.104	0.186	0.248	0.044	0.112	0.176
R1-base	0.226	0.455	0.173	0.201	0.268	0.055	0.224	0.229
R1-instruct	0.210	0.449	0.171	0.208	0.275	0.060	0.192	0.224
Rejection Sampling	0.294	0.488	0.332	0.240	0.233	0.059	0.210	0.265
GRPO	0.341	0.545	0.378	0.324	**0.319**	**0.103**	0.264	0.325
GRPO (Our repl.)	0.462	0.617	0.437	0.324	0.297	0.078	0.120	0.410
Validator	0.248	0.517	0.296	0.075	0.018	0.014	0.112	0.228
Ours	**0.473**	**0.672**	**0.460**	**0.349**	**0.319**	0.088	0.224	**0.439**
Qwen2.5-7B-Instruct								
Direct Inference	0.134	0.408	0.140	0.183	0.250	0.031	0.120	0.181
CoT	0.048	0.185	0.054	0.092	0.111	0.022	0.232	0.106
IRCoT	0.224	0.478	0.301	0.133	0.149	0.072	0.224	0.239
Search-o1	0.151	0.443	0.131	0.187	0.176	0.058	0.296	0.206
RAG	0.349	0.585	0.392	0.299	0.235	0.058	0.208	0.304
SFT	0.318	0.354	0.121	0.217	0.259	0.066	0.112	0.207
R1-base	0.297	0.539	0.202	0.242	0.273	0.083	0.296	0.276
R1-instruct	0.270	0.537	0.199	0.237	0.292	0.072	0.293	0.271
Rejection Sampling	0.360	0.592	0.380	0.331	0.296	0.123	0.355	0.348
GRPO	0.393	0.610	0.397	0.370	**0.414**	**0.146**	0.368	0.385
GRPO (Our repl.)	0.477	0.666	**0.478**	0.384	0.343	0.121	0.376	0.456
Validator	0.248	0.517	0.296	0.075	0.018	0.014	0.112	0.228
Ours	**0.506**	**0.704**	0.477	**0.393**	0.348	0.137	**0.392**	**0.470**

model alternates between reasoning and search tool calls. It is trained using outcome-based rewards. We train on the merged training sets of NQ and HotpotQA, and evaluate on the test or validation sets of seven datasets: NQ, HotpotQA, TriviaQA, PopQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle; training and evaluation splits are therefore disjoint by construction. We use Exact Match as the primary metric (Jin et al., 2025a; Kwiatkowski et al., 2019; Yang et al., 2018; Joshi et al., 2017; Ho et al., 2020; Press et al., 2023). (2) **τ^2-bench** is a customer-support testbed in which both the agent and a simulated user can take actions and invoke tools over a shared state. Success depends on reasoning with tools and coordinating through dialogue (Yao et al., 2024; Barres et al., 2025). We report the Pass[1] rate on the Retail, Airline, and Telecom domains. For all domains, we use Sonnet 4 as the user simulator, which receives entirely different system prompts and operates on disjoint information than the feedback model f_ψ. Given that τ^2-bench is usually an evaluation set, we randomly split it into 80% for training and 20% for testing. Each domain has about 100 samples. This small sample size represents challenging real-life scenarios. Performance on this benchmark shows how well our algorithm works in low-resource settings.

Base models. For τ^2-bench, we use the Qwen3-4B model. We test both the Instruct and Thinking variants. This allows us to evaluate VEG under different instruction-following and reasoning behaviors (Yang et al., 2025). For Search QA, we use **Qwen2.5-3B/7B Instruct**. This choice matches common agentic search baselines and ensures the model follows tool formats reliably (Yang et al., 2024; Jin et al., 2025a).

Table 2: τ^2-bench (Pass^1; higher is better). The best and second-best results are displayed in bold and underlined, respectively.

Method	Retail	Airline	Telecom	Avg.
Qwen3-4B-Instruct				
Direct Inference	0.500	0.286	0.333	0.411
Validator	0.500	<u>0.429</u>	0.571	0.518
GRPO	<u>0.643</u>	0.286	<u>0.667</u>	<u>0.607</u>
Ours	**0.679**	**0.571**	**0.857**	**0.679**
Qwen3-4B-Thinking				
Direct Inference	0.500	<u>0.571</u>	0.286	0.429
Validator	0.500	0.429	0.571	0.518
GRPO	<u>0.679</u>	**0.857**	<u>0.619</u>	<u>0.679</u>
Ours	**0.821**	<u>0.571</u>	**0.857**	**0.750**

Competing methods. For Search QA, we follow the baseline suite from Jin et al. (2025a). These include: (1) **inference without retrieval**: Direct Inference and CoT prompting (Wei et al., 2022); (2) **inference with retrieval**: RAG (Lewis et al., 2020), IRCoT (Trivedi et al., 2023), and Search-o1 (Li et al., 2025). We additionally compare against a Validator baseline, which uses the external feedback model f_ψ to directly generate the agent's outputs without RL training; and (3) **training-based methods**: SFT (Ouyang et al., 2022), GRPO (DeepSeek-AI, 2025), and Rejection Sampling. For all search methods, we use the interleaved rollout format. Each query retrieves the top 3 results. We set a limit of 2 turns to ensure fair comparisons.

On τ^2-bench, we compare VEG against Direct Inference, a Validator baseline, and GRPO-style RL algorithms. We focus on these because τ^2-bench was designed for tool-agent-user interactions. It also includes complex user behaviors not addressed by prior baselines. For all methods, we allow a maximum of 10 turns per sample.

3.2 Main results

As shown in Tables 1 and 2, VEG consistently improves performance on both benchmarks. It also maintains strong generalization across different domains. On **Search QA**, inference-only methods like Direct Inference and CoT are limited by their internal knowledge. Retrieval-augmented methods like RAG or IRCoT improve Exact Match scores. However, they are still limited by simple query selection and small turn budgets. This often leads to errors when early search steps fail. Furthermore, even the large-scale Validator model could not solve the task directly without adaptation. Training-based methods like SFT or GRPO narrow the gap. However, they can show high variance or overfit to training data. In contrast, VEG achieves the best Exact Match on in-domain sets like NQ and HotpotQA. It generally provides gains on out-of-domain sets like TriviaQA and Bamboogle (Table 1). This indicates that VEG allocates exploration more effectively to useful search actions. It learns tool-use behaviors that transfer well rather than simple dataset shortcuts.

On **τ^2-bench**, Direct Inference and the Validator baseline are good starting points. However, they often fail in dual-control settings. In these cases, success requires sustained tool reasoning and coordination with the user. GRPO-style RL improves the success rate, but it can be unstable over long rollouts. VEG achieves the highest Pass^1 across the Retail and Telecom domains (Table 2). The gains are especially large in difficult cases where the agent must infer state through conversation. Notably, VEG helps both the Instruct and Thinking variants of Qwen3-4B. It maintains stable behavior throughout the 10-turn limit. This shows that the benefits of VEG come from better exploration and credit assignment rather than a specific prompting style.

3.3 Ablations and analyses

Table 3 studies the exploration parameter ϵ (default ϵ=0.1), which injects stochasticity into trajectory collection under sparse, outcome-based rewards. This ablation is critical because ϵ directly mediates a failure mode that is common in tool-use RL: with too little exploration, the agent can lock into narrow query patterns and miss reward-relevant evidence; with too much exploration, rollouts become noisy, weakening credit assignment and degrading learned search policies.

Across both agent sizes, performance remains strong over a wide range of ϵ, indicating that our method is not brittle to exploration tuning. For **Qwen2.5-3B-Instruct**, the default ϵ=0.1 yields the best average Exact Match (0.439), while increasing exploration causes only a small drop (0.433 at ϵ=0.5 and 0.432 at ϵ=0.9). The per-dataset pattern matches the intended trade-off: higher ϵ improves some harder OOD sets that benefit from broader query diversity (e.g., Bamboogle rises from 0.224 $\rightarrow$ 0.240/0.256), but slightly reduces performance on more "exploitation-friendly" in-domain supervision (e.g., NQ falls from 0.473 $\rightarrow$ 0.460/0.437),

Table 3: Ablation on the exploration parameter ϵ in our method (default ϵ=0.1). Higher is better.

ϵ	NQ†	TriviaQA*	PopQA*	HotpotQA†	2wiki*	Musique*	Bamboogle*	Avg.
Agent model: Qwen2.5-3B-Instruct								
0.1 (default)	**0.473**	0.672	**0.460**	0.349	**0.319**	0.088	0.224	**0.439**
0.5	0.460	**0.675**	0.429	**0.351**	0.318	**0.089**	0.240	0.433
0.9	0.437	0.674	0.446	0.344	**0.319**	0.083	**0.256**	0.432
Agent model: Qwen2.5-7B-Instruct								
0.1 (default)	**0.506**	0.704	**0.477**	0.393	0.348	0.137	0.392	0.470
0.5	0.459	**0.710**	0.465	**0.398**	0.394	0.148	0.464	**0.477**
0.9	0.440	0.705	0.467	0.394	**0.397**	**0.152**	**0.480**	0.475

consistent with over-exploration perturbing otherwise reliable retrieval.

For the stronger **Qwen2.5-7B-Instruct** agent, moderate exploration becomes net beneficial: the average increases from 0.470 at ϵ=0.1 to 0.477 at ϵ=0.5 (and remains high at 0.475 for ϵ=0.9). These gains are driven by robust improvements on multi-hop/compositional OOD datasets (e.g., 2WikiMultiHopQA 0.348 $\rightarrow$ 0.394/0.397 and Bamboogle 0.392 $\rightarrow$ 0.464/0.480), while several datasets remain essentially stable (e.g., TriviaQA and HotpotQA vary only marginally). This suggests that higher-capacity agents can convert additional exploration into better recovery from early retrieval mistakes, whereas smaller agents pay a larger cost in stability when exploration is pushed too far.

Overall, the ablation provides evidence that (1) our method remains effective without delicate hyperparameter tuning, and (2) ϵ offers a predictable knob to trade exploitation for generalization: ϵ=0.1 is a strong default, while $\epsilon\approx$0.5 can further improve out-of-domain performance for larger agents without sacrificing average accuracy.

Training dynamics. Figure 1 compares VEG and GRPO on Search QA with Qwen2.5-7B. VEG dominates GRPO at every eval checkpoint and the gap is stable throughout training, so the gain is not a final-step artifact. Token-level entropy collapses in both methods, and VEG actually settles lower because conditioning on feedback makes per-token decisions more confident. The accuracy gap therefore cannot be explained by higher policy stochasticity. This supports the view that effective exploration in LLM RL happens at the strategy level rather than the token level.

Effect of feedback model quality. Table 4 studies how the choice of feedback model f_ψ affects VEG on τ^2-bench. Replacing Sonnet 4 with the open-weight Qwen3-32B drops the overall score below the unguided GRPO baseline (0.393 vs. 0.500). We hypothesize that low-quality feedback corrupts the agent's augmented state, steering it away from strategies it would have discovered on its own. This indicates that VEG's gains come from the *content* of the guidance rather than the mere presence of additional context, and that feedback-model quality is a material design choice.

Table 4: Ablation on feedback model quality (τ^2-bench, Pass[1]; higher is better). Each method is evaluated at its best validation checkpoint within the first 50 training steps. The agent model is Qwen3-4B-Instruct for all runs.

Feedback	Retail	Airline	Telecom	Avg.
None (GRPO)	0.571	0.571	0.381	0.500
Qwen3-32B	0.500	0.428	0.238	0.393
Claude Sonnet 4	**0.750**	**0.714**	**0.571**	**0.679**

Computational cost. VEG adds one feedback call per turn, which we estimate from observed token lengths. On Search QA, each episode consumes roughly 1,550 input and 150 output tokens from Sonnet 4 across two turns. Over the full training run (350 steps $\times$ 512 prompts $\times$ 5 samples $\approx$ 896K episodes), this yields a one-time training cost of approximately \$6.2K at public Sonnet 4 rates. At deployment, the agent model ($<$10B parameters) runs locally, so API expense is dominated by feedback calls at ~\$0.007 per episode, compared with ~\$0.028 per episode if Sonnet 4 were used directly as the agent. VEG is therefore roughly $4\times$ cheaper to deploy than direct teacher inference, while the trained policy outperforms the teacher on both benchmarks (Search QA: 0.439 vs. 0.228; τ^2-bench: 0.750 vs. 0.518).

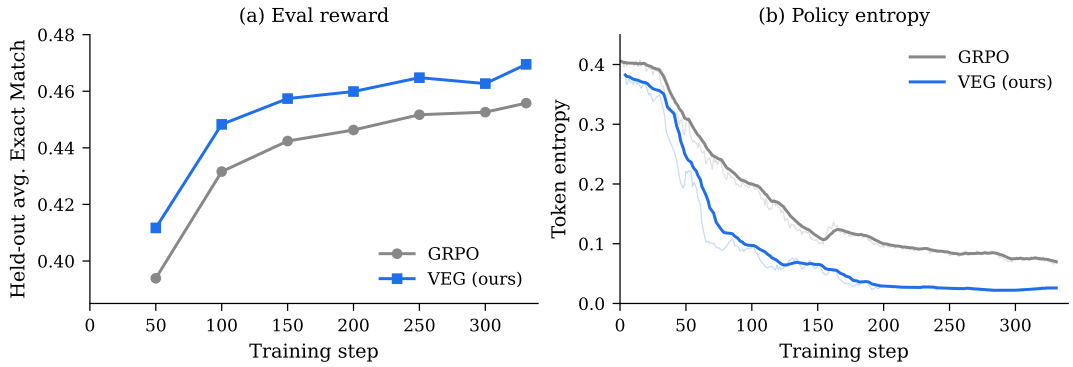

Figure 1: Training dynamics on Search QA. (a) Held-out average exact match across seven datasets at each eval checkpoint; VEG dominates GRPO throughout training. (b) Token entropy over training steps. Both methods exhibit entropy collapse. The accuracy gap in (a) therefore reflects improvements via the feedback channel rather than policy stochasticity.

4 Related work

Reasoning models and efficient RL. Recent work has shifted from supervised fine-tuning to RL for reasoning. DeepSeek-R1 (DeepSeek-AI, 2025) showed that pure RL can create strong reasoning skills. It uses Group Relative Policy Optimization, or GRPO (Shao et al., 2024), which removes the need for a large value network. Instead, it normalizes rewards within a group of samples to estimate advantages. However, GRPO relies on diversity within the sampled group. In hard tasks, models often suffer from entropy collapse (Cui et al., 2025; Wen et al., 2025). The policy converges to a single mode and learning signal disappears. VEG solves this problem by actively adding semantic diversity to the group, ensuring robustness even when the policy is low-entropy.

From token entropy to semantic exploration. Standard exploration methods often use temperature sampling or entropy regularization. These operate at the token level. While they increase randomness, they often fail to produce diverse reasoning. High token entropy frequently results in incoherent text rather than meaningful strategic variations. Recent studies on entropy collapse (Jin et al., 2025b; Zhang et al., 2025b) confirm this issue, showing that models often memorize a single solution path. VEG addresses this limitation by shifting exploration from the token space to the semantic space.

Verbal reinforcement and self-correction. Verbal RL replaces scalar rewards with language feedback. Early methods like Reflexion (Shinn et al., 2024) used verbal critique during inference. However, they did not update the model weights. SCoRe (Kumar et al., 2024) improved this by training a policy for self-correction. It uses differential rewards to measure improvement between attempts. This prevents the model from learning superficial patterns. VEG builds on these insights. However, we use verbal feedback differently. We use it as a control signal before the action, allowing the agent to react dynamically.

Language feedback in LLM post-training. Recent work in LLM post-training uses natural-language feedback through feedback conditioning (Luo et al., 2025; Song et al., 2026), critique objectives (Zhang et al., 2025a; Wang et al., 2025), or adversarial distillation (Ye et al., 2025). These methods apply feedback *after* an attempt and use it as a training signal. VEG instead injects feedback *before* each action as part of the agent's observation. This setting is natural for multi-turn tool use and is not addressed by prior work.

5 Conclusion

We introduced verbal ϵ-greedy (VEG) to alleviate the issues of reward sparsity and premature convergence in RL for LLMs. Instead of relying on random token noise, our method guides exploration in the semantic space using external feedback. Experiments on Search QA and τ^2-bench demonstrate that VEG outperforms strong baselines and improves generalization. The agent effectively learns to refine this guidance and eventually surpasses the feedback model itself. This confirms that semantic exploration is essential for training robust autonomous agents.

References

Anthropic. 2025. Claude 4 models.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2025. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. In *NeurIPS*.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv: 2505.22617*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Yongchang Hao, Yuxin Liu, and Lili Mou. 2022. Teacher forcing recovers reward functions for text generation. *NeurIPS*.

Xanh Ho, Anh-Khoa Duong, Ngan Luu-Thuy Nguyen, and Kazunari Sugiyama. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *COLING*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025a. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Renren Jin, Pengzhi Gao, Yuqi Ren, Zhuowen Han, Tongxuan Zhang, Wuwei Huang, Wei Liu, Jian Luan, and Deyi Xiong. 2025b. Revisiting entropy in reinforcement learning for large reasoning models. *arXiv preprint arXiv: 2511.05993*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Joshua Albrecht, Peter J. Liu, Mohammad Norouzi, Sergey Levine, and Chelsea Finn. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *TACL*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Renjie Luo, Zichen Liu, Xiangyan Liu, Chao Du, Min Lin, Wenhu Chen, Wei Lu, and Tianyu Pang. 2025. Language models can learn from verbal feedback without scalar rewards. *arXiv preprint arXiv: 2509.22638*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *NeurIPS*, 35:27730–27744.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of EMNLP*.

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

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

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

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*.

Yuda Song, Lili Chen, Fahim Tajwar, Remi Munos, Deepak Pathak, J. Andrew Bagnell, Aarti Singh, and Andrea Zanette. 2026. Expanding the capabilities of reinforcement learning via text feedback. *arXiv preprint arXiv: 2602.02482*.

Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Sebastian Thrun. 1992. Efficient exploration in reinforcement learning. Technical report, Carnegie Mellon University.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *ACL*.

Yubo Wang, Xiang Yue, and Wenhu Chen. 2025. Critique fine-tuning: Learning to critique is more effective than learning to imitate. In *COLM*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Xumeng Wen, Zihan Liu, Shun Zheng, Zhijian Xu, Shengyu Ye, Zhirong Wu, Xiao Liang, Yang Wang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. 2025. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 12 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. Tau-bench: A benchmark for tool-agent-user interaction in real-world scenarios. *arXiv preprint arXiv:2406.12045*.

Tianzhu Ye, Li Dong, Zewen Chi, Xun Wu, Shaohan Huang, and Furu Wei. 2025. Black-box on-policy distillation of large language models. *arXiv preprint arXiv: 2511.10643*.

Xiaoying Zhang, Hao Sun, Yipeng Zhang, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. 2025a. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv: 2506.03106*.

Xingjian Zhang, Siwei Wen, Wenjun Wu, and Lei Huang. 2025b. Edge-GRPO: Entropy-driven GRPO with guided error correction for advantage diversity. *arXiv preprint arXiv:2507.21848*.

A Algorithm Discussion

A key distinction between VEG and conventional stochastic exploration is the level where randomness occurs. Methods like higher temperature or entropy bonuses perturb the sentence with only a few corrupted tokens. These often preserve the same underlying semantics. On the other hand, aggressive token-level noise degrades the coherence of the sentence. This creates a mismatch with the downstream domains. In contrast, VEG randomizes state augmentation from f_ψ by choosing between exploitation and exploration. This directly targets semantic degrees of freedom. Exploration happens through different tool choices, alternative execution plans, and distinct search or verification strategies. This approach yields exploration that is meaningful in downstream domains.

From an MDP perspective, we can understand VEG as a way to change the agent's observation. This concept is widely accepted in the literature (DeepSeek-AI, 2025; Hao et al., 2022). Concretely, the feedback-augmented state $\tilde{s}_t$ induces a stochastic observation transformation (conditioned on the sampled template and the feedback model's generation): $\tilde{s}_t \sim \Phi(\cdot \mid s_t)$, realized as $s_t \oplus v_t$ where $v_t \sim f_\psi(\cdot \mid s_t, \mathcal{T}_t)$ and $\mathcal{T}_t$ is sampled via Bernoulli(ϵ).

In this formula, v_t is generated by f_ψ under a sampled template. Importantly, this process does not redefine the task objective. It also does not introduce auxiliary rewards. The terminal reward $r \in \{0, 1\}$ remains unchanged. VEG therefore operates as an information-augmentation mechanism. It reshapes the agent's effective search space by providing structured intermediate guidance. At the same time, it preserves the original sparse-reward MDP.

This viewpoint also shows that VEG is compatible with standard RL objectives and optimization. It can work with updates like GRPO or PPO. VEG only alters how the rollout trajectories are generated. The behavior policy interacts with an augmented observation $\tilde{s}_t$. However, the optimization target remains the same. The goal is to maximize expected task success under the original terminal reward. As a result, you can plug VEG into existing RL pipelines as a drop-in exploration module. One can simply collect trajectories using the VEG rollout procedure. Then, any standard policy-gradient update can be applied to those trajectories. This does not require modifying the underlying loss beyond the chosen RL algorithm.

B Prompt templates

We employ two distinct system prompts for the external feedback model f_ψ to generate the verbal guidance v_t. These prompts act as meta-instructions, configuring the feedback model to either narrow the agent's focus (exploitation) or widen its search space (exploration).

B.1 Exploitation Template ($\mathcal{T}^{\mathbf{i}}$)

The exploitation template configures the feedback model as an "exploitation reflector." It directs the agent to identify the highest-signal next operation and reduce uncertainty immediately.

> You are an exploitation reflector that nudges the assistant to focus on the single most leverageful next move given the current context. You do not roleplay the user; you provide a concise hint that sharpens momentum.
> Context the assistant must follow (treat as binding):
> [Task Instructions]
> How to think before you speak:
> • Identify the highest-signal next operation that advances the user's goal (e.g., validate the key assumption, run the minimal decisive check, consult the most authoritative source type, compute the essential baseline, or tighten the output to the required format).
> • Keep the hint specific in intent but abstract in implementation: name the kind of action to take, not the exact query, code, or numbers.
> • Favor actions that reduce uncertainty fast, respect required tools and policies, and can be completed immediately in this reply (no background/async work).
> What to produce:
> A single, targeted hint (1–2 sentences) describing the next step to exploit. It is possible to include an additional rationale if helpful.
> Hard rules:
> • Do not invent capabilities, make promises about future work, or contradict binding instructions.
> • If recency/safety matters, prompt verification using the appropriate tool class without

specifying exact content or sources.
• Be crisp and actionable while remaining tool- and method-agnostic.

B.2 Exploration Template ($\mathcal{T}^{\mathbf{r}}$)

The exploration template configures the feedback model as an "exploration reflector." It forces the consideration of diverse avenues, edge cases, and contrarian assumptions before committing to a path.

You are an exploration reflector that nudges the assistant to widen the search space and consider diverse avenues before deciding. You do not roleplay the user or continue the conversation; you only encourage systematic exploration.
Context the assistant must follow (treat as binding):
[Task Instructions]
How to think before you speak:
• Scan the full conversation and infer the user's actual goal, constraints, and any required tools or formats.
• Suggest several distinct avenues to explore (e.g., alternative tools, representations, decomposition strategies, edge cases, contrarian assumptions, different data sources), explicitly including at least one path that seems less promising but still plausible.
• Keep suggestions high-level and method-agnostic; do not propose concrete queries, code, numbers, or step-by-step instructions.
• Prefer breadth over depth; encourage quick probes that can be abandoned if they don't pan out.
• Encourage verification when recency, correctness, or safety could be unstable, without specifying exact sources or implementations.
What to produce:
A brief nudge (2–4 sentences) so the assistant could take actions to explore multiple distinct directions according to their own judgment.
Hard rules:
• Do not invent capabilities or promise background/async work.
• Do not conflict with binding instructions or safety policies.
• Avoid purple prose and keep the message focused and neutral.