SKILL2TASK: LEARNING FROM SKILLS THROUGH GRAPH-GUIDED TASK SYNTHESIS FOR AGENTIC RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents can reuse past experience during reinforcement learning (RL) through natural-language skills distilled from their trajectories on training tasks. However, we uncover a bottleneck in SkillRL: generalization can stall in later training stages. Skills distilled from individual task trajectories may also contain overly broad, conflicting, or incorrect recommendations. These challenges motivate broadening the agent's training experience to support further policy improvement and provide opportunities to test potentially unreliable recommendations through execution feedback. We therefore introduce a framework that uses skills to specify and construct training tasks with varied goals and conditions. To this end, we organize skills into a graph that captures their recommendations and applicability conditions, together with relations that describe how skills can be combined and when their recommendations conflict. The graph guides task synthesis in existing environments, with verifiable success criteria and checks for task relevance and solvability. Starting from a SkillRL checkpoint, we incorporate the resulting tasks into continued RL, using environment feedback to train the agent to combine skills and decide when to follow, adapt, or reject their recommendations. On ALFWorld, Qwen3-4B-Instruct trained with tasks synthesized by our framework achieves higher average success rates than SkillRL on both seen and unseen tasks, with gains of 8.1 percentage points on unseen Pick&Place tasks and 11.1 points on unseen Examine tasks.

1 INTRODUCTION

Skills provide large language model (LLM) agents with reusable procedures and strategies for solving recurring problems, allowing experience to accumulate across tasks (Wang et al., 2024; Chen et al., 2026). This reuse also benefits reinforcement learning (RL), where skills guide the agent's interactions as it gathers experience for policy improvement (Wu et al., 2025; Zhou et al., 2025). Building on this idea, SkillRL distills successful trajectories and lessons from failures into a hierarchical library of natural-language skills (Xia et al., 2026a). During training, the policy retrieves these skills to guide its actions, while validation failures prompt additions and revisions to the library.



Figure 1: Learning curves on ALFWorld tasks in unseen environments. SkillRL plateaus with continued training; from a shared checkpoint, training on our synthesized tasks yields further improvement.

However, we uncover two limitations in SkillRL. First, continued training encounters a late-stage bottleneck: success rates on Pick&Place and Examine tasks in unseen ALFWorld environments (Shridhar et al., 2020) plateau despite continued policy optimization and skill-library updates on the original task set (Figure 1). Second, the accumulated skill library can provide unreliable guidance, including overfitted, conflicting, and incorrect recommendations (Section 3). In particular, skill overfitting occurs when assumptions tied to specific trajec-

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.

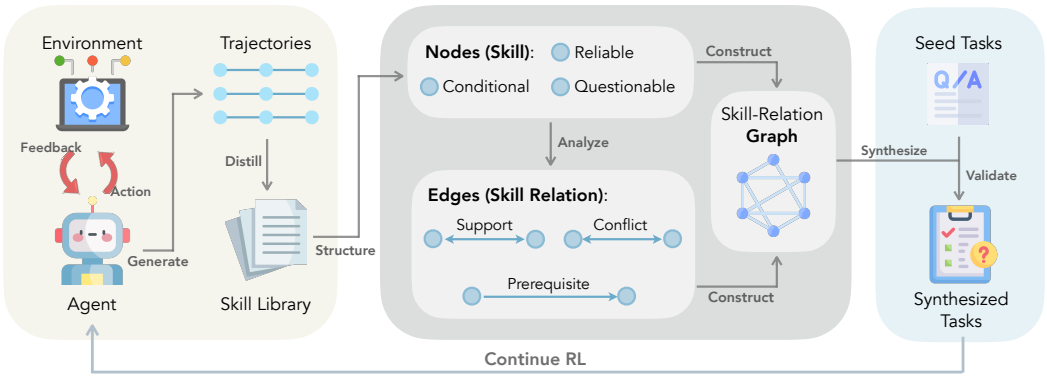


Figure 2: Overview of our **Skill2Task** framework. Skills distilled from trajectories are organized into a skill relation graph. The graph guides task synthesis from seed tasks to practice strategy combinations and applicability judgments. Validated tasks are then used to continue RL training.

tories are promoted to general rules. For example, one skill rules out searching for kitchenware in a fridge, although ALFWorld permits such locations. On targeted stress tasks, removing only the flagged skills raises the same policy's success from 53.5% to 67.4%. These observations motivate scaling skill-augmented RL beyond the original task set by expanding training experience to support sustained policy improvement and more reliable skill use.

Our key idea is to use accumulated skills to specify what the agent should practice next and synthesize new training tasks around these specifications. First, we construct tasks around individual skills, varying goals and constraints to practice valid skill use and test recommendations where their assumptions may not hold. Second, we combine skills to construct tasks that require coordinating their use, satisfying dependencies, or choosing between competing recommendations. By introducing new combinations of goals and requirements, skill composition can broaden the training task distribution (Fan et al., 2026; Cheng et al., 2026). In both forms, skills specify what to practice or test, while environment goals or verified answers determine task success. This separation allows even potentially unreliable recommendations to inform task synthesis: their assumptions identify what to test, while task feedback helps the agent learn when to follow, adapt, or disregard the guidance.

We introduce **Skill2Task**, which implements both forms of synthesis through a *skill relation graph* (Figure 2). A graph is well suited to this purpose: nodes retain each skill's behavior and applicability conditions, while edges represent interactions between skills. We select nodes or edges to guide task synthesis from existing seeds, then validate each candidate for target relevance and solvability. Starting from a SkillRL checkpoint, we continue RL on the validated tasks and obtain further gains on Pick&Place and Examine in unseen ALFWorld environments (Figure 1). Our main contributions are as follows:

- We uncover a late-stage training bottleneck in SkillRL on selected tasks in unseen ALFWorld environments. We identify overfitted, conflicting, and incorrect skills, and demonstrate through controlled stress tests that unreliable guidance impairs task execution.
- We propose Skill2Task, which uses a skill relation graph to turn accumulated skills' applicability conditions, limitations, and interactions into verifiable tasks for continued RL.
- Experiments at 4B and 8B show higher average performance than SkillRL on unseen ALFWorld environments and held-out QA benchmarks. ALFWorld analyses at 4B support both node-conditioned and relation-conditioned synthesis and show lower execution error rates.

2 Related Work

Learning from experience and skills. LLM agents can reuse past experience by retaining verbal feedback, distilled insights, or executable skills (Shinn et al., 2023; Zhao et al., 2024; Wang et al., 2024). This reuse extends to policy learning: EvolveR, INSPO, and SkillRL use experience-derived

(a) Conflicting Skills: incompatible next steps

Shared context. The agent holds an unheated target; heating is required and the final destination is known.

`gen_003`: Destination First Policy

"After picking up a goal object, **navigate directly to the known target receptacle and place it** before resuming further search."

`gen_005`: Use State-Changing Tools Early

"For tasks requiring heated/cooled/clean objects, acquire the object, then immediately use the nearest suitable appliance to **change its state before any placement**."

Conflict. Place at the destination now versus transform first: both recommendations cover this state but prescribe different action orders.

(b) Overfitted Skills: a location prior becomes an absolute rule

`dyn_037`: Kitchenware Not In Cold Storage

"Mugs, plates, pans, bowls, cups, and dishes are **never stored in the fridge or freezer**."

Environment-permitted counterexample

A mug is inside a fridge, an allowed object–receptacle pairing in ALFWorld. If it is the only target mug, searching elsewhere cannot recover it.

Applicability boundary. A preference for common storage locations becomes an exclusion that can remove the required search location.

(c) Incorrect Skills: a recommendation violates an action precondition

`dyn_024`: Collect All Before Any Place

"When a task requires multiple objects at a destination, find and **collect ALL required items first**. Do not transport or place any item until **every required object is in hand**."

ALFWorld pickup precondition

Picking up an object requires an empty hand. After taking the first item, the agent must put it down before taking the second. *Feasible order:*

take $o_1 \rightarrow$ put $o_1 \rightarrow$ take $o_2 \rightarrow$ put o_2.

Rule violation. The skill requires simultaneous possession of multiple items, which the pickup precondition prevents.

Figure 3: Three failure patterns in SkillRL's skill bank on ALFWorld: (a) conflicting recommendations, (b) location priors overgeneralized into absolute rules, and (c) violations of preconditions.

guidance during RL (Wu et al., 2025; Zhou et al., 2025; Xia et al., 2026a). Because stored skills can be unreliable, ReSkill studies conditional skill revisions, while UCOB compares returns with and without skills to inform policy and memory updates (He et al., 2026; Tu et al., 2026).

Relational structure for skill reuse. Skill reuse also depends on how recommendations interact; graph-based methods model these relations for policy learning (Li et al., 2026a), retrieval (Liu et al., 2026; Bai et al., 2026), and execution (Xia et al., 2026b). CaSKG further examines relation reliability before retrieval (Li et al., 2026b). We use these relations to specify what the agent should practice through new tasks.

Synthesizing training tasks for agents. Terminal-World and SKT generate skill-based tasks and trajectories (Cheng et al., 2026; Tan et al., 2026), while Socratic-SWE guides task generation with skills distilled from execution traces (Xiao et al., 2026). SkillSynth broadens trajectory coverage for supervised fine-tuning (SFT) by composing workflows from a skill graph (Fan et al., 2026). Skill-conditioned generation can also support RL, as in Skill Self-Play (Huang et al., 2026). Skill2Task focuses on turning the applicability conditions, boundaries, and conflicts of the agent's own accumulated skills into verifiable training tasks. Continued RL on these tasks lets the policy learn when to follow, adapt, or reject that guidance through autonomous exploration and task rewards.

3 When Does Skill Guidance Hurt?

Skills abstract past experience into reusable guidance, but this abstraction can obscure the conditions under which a recommendation is valid. When those conditions are not met, following the recommendation may lead the agent to take invalid or counterproductive actions. To examine this risk,

we construct ALFWorld stress tasks around potentially unreliable skills in SkillRL and compare the same policy with and without these skills.

3.1 Identifying Skill Failure Patterns

To identify where skill guidance may become unreliable, we use GLM-5.2 (Zeng et al., 2026) to review each skill's description and applicability conditions alongside semantically related skills and ALFWorld's rules. The review flags three potential failure patterns: **conflicting skills** prescribe incompatible actions under overlapping conditions; **overfitted skills** encode context-specific assumptions as general recommendations that fail when those assumptions no longer hold; and **incorrect skills** contradict the environment's rules.

Figure 3 shows how these issues arise within individual recommendations and between skills. These cases guide stress-task construction by specifying which skills to test and under what conditions.

3.2 Constructing Stress Tasks

For each flagged skill or pair, we enumerate legal goals in existing ALFWorld scenes, varying task type, target object, and destination to test the disputed recommendation. Scene objects remain fixed, while task descriptions and success conditions are updated together. Candidates must retrieve the targeted skill or both conflicting skills through the agent's normal retrieval procedure.

Conflict tasks. For each flagged conflict pair, we select tasks where the skills prescribe incompatible actions under overlapping conditions. The disputed choice must matter for completing the task.

Overfitted-skill tasks. For *applicability shifts*, we vary task structure to challenge the flagged skill's narrow assumptions. For *absolute-rule counterexamples*, we select existing object locations that contradict the skill's claim and ensure no alternative target instance bypasses the counterexample.

Incorrect-skill tasks. We select goals where following the flagged skill would violate an action precondition or a rule for transforming an object. The goals must remain achievable using the available objects, appliances, and valid actions.

3.3 Evaluating the Effect of Skill Guidance

We evaluate SkillRL's best overall checkpoint with the full library (*w/ unreliable skills*) and with only the flagged skills removed (*w/o unreliable skills*). All other evaluation settings are kept identical. Using the environment's goal checker, we report success rates over all 86 tasks and by category, pooling the two overfitting variants.

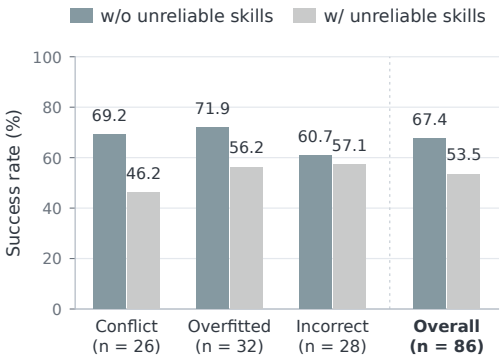


Figure 4: Effect of unreliable skill guidance. Success rates of the same SkillRL checkpoint with and without unreliable skills across three diagnostic categories.

Results. Figure 4 shows lower success with skill guidance across all three stress categories. Overall success drops from 67.4% without unreliable skills to 53.5% with unreliable skills. The largest gap occurs on Conflict tasks (46.2% with skills versus 69.2% without), followed by Overfitted (56.2% versus 71.9%) and Incorrect tasks (57.1% versus 60.7%). These results show that the unreliable guidance can hinder execution on the tasks designed to test its limitations. This motivates synthesizing tasks around individual skill limitations and conflicts between skills, so that execution feedback can help agents learn when to follow, adapt, or reject a recommendation.

4 METHOD

We use a skill relation graph to turn accumulated experience into task requirements for practicing individual skills and their interactions. The overall pipeline is

$$\mathcal{S} \xrightarrow{\text{structure}} G \xrightarrow[\mathcal{D}_{\text{orig}}]{\text{synthesize}} \mathcal{D}_{\text{cand}} \xrightarrow{\text{validate}} \mathcal{D}_{\text{syn}}. \tag{1}$$

Given a skill library $\mathcal{S}$ and an original training task set $\mathcal{D}_{\text{orig}}$, we construct a graph G whose nodes describe individual skills and whose typed edges encode their relations. We use G to generate a candidate task set $\mathcal{D}_{\text{cand}}$ from seed tasks in $\mathcal{D}_{\text{orig}}$, then retain validated candidates as the synthesized training set $\mathcal{D}_{\text{syn}}$.

4.1 SKILL RELATION GRAPH

To make these synthesis targets explicit, we first organize the skill library into a typed graph $G = (V, E)$. Here V is the set of skill nodes describing individual recommendations, and E is the set of edges describing their relations under specified task conditions.

4.1.1 SKILL NODES

For each skill, a node $v_i \in V$ records three natural-language fields:

$$v_i = (\text{conditions, behavior, effect}). \tag{2}$$

The *conditions* specify when the recommendation applies, the *behavior* describes what it recommends doing, and the *effect* states the intended outcome. A teacher extracts these fields, records exceptions, and assesses the recommendation against the environment's rules:

- **Reliable Node:** judged consistent with its stated scope and the environment's rules.
- **Conditional Node:** useful under additional restrictions or preconditions.
- **Questionable Node:** unresolved or potentially contradicted by environment rules.

These annotations indicate whether a recommendation is reliable within its stated scope, requires additional conditions, or may be inappropriate to follow. They guide task synthesis by identifying which behaviors to reinforce and which applicability conditions or limitations to examine.

4.1.2 SKILL EDGES

An edge $e_{ij} \in E$ connects two skill nodes $v_i, v_j \in V$:

$$e_{ij} = (v_i,\ v_j,\ \text{relation}). \tag{3}$$

The *relation* specifies how the two recommendations interact under relevant task conditions. We distinguish three relation types:

Support. Skills recommend compatible behaviors that make complementary contributions to a shared goal. The relation is symmetric and identifies strategies that can be coordinated within a task.

$$v_i \xleftrightarrow{\text{Support}} v_j$$

Prerequisite. The first skill's expected effect provides a state or information required by the second skill. The arrow points from the enabling skill to the dependent skill, specifying the dependency that synthesis should expose.

$$v_i \xrightarrow{\text{Prerequisite}} v_j$$

Conflict. The skills' stated conditions overlap, but their recommendations prescribe incompatible choices at the same decision point. This symmetric relation records the disputed choice and the conditions under which the conflict arises.

$$v_i \xleftrightarrow{\text{Conflict}} v_j$$

4.1.3 GRAPH CONSTRUCTION

We build the graph by connecting skills that exhibit one of the three relations defined above. We first use skill-text similarity and co-retrieval on original training tasks to identify candidate pairs. For each pair, the teacher compares the skills' applicability conditions, recommended behaviors, and expected effects to assign a relation or reject the connection. The resulting graph provides individual nodes and related skill pairs as targets for task synthesis.

4.2 GRAPH-GUIDED TASK SYNTHESIS

Given a seed task from $\mathcal{D}_{\text{orig}}$, we retrieve relevant skills and select a node or an associated edge as the synthesis target. The selected target guides changes to task goals or conditions, while the seed provides an environment or source material for constructing the new task.

Node-conditioned synthesis. We use the selected node's behavior and intended effect to define the practice target, and its applicability conditions and assessment to guide the task constraints. For *Reliable* nodes, we vary instance details within the recommendation's assessed scope while retaining the seed task type. For *Conditional* nodes, we construct tasks that satisfy the additional preconditions or restrictions identified in the assessment. For *Questionable* nodes, we set feasible goals in settings where the recommendation's assumptions may not hold.

Relation-conditioned synthesis. We combine the selected edge's relation type with the attributes of its endpoint skills to specify the interaction requirements of the task. For *Support* edges, we combine requirements addressed by the two skills within a shared goal. For *Prerequisite* edges, we construct goals that involve a state or information produced by the first skill and needed by the second. For *Conflict* edges, we construct one task in which the two skills prescribe incompatible actions at a decision relevant to the goal. The disputed choice must affect goal completion, providing practice in assessing recommendations against the task's conditions. See Appendix F for examples.

4.3 TASK INSTANTIATION AND VALIDATION

We instantiate each candidate using the seed task's environment or source material. We construct the instruction and its corresponding goal or reference answer together, so that the agent is evaluated on the task it is asked to perform.

Target relevance. A teacher reviews each task's conditions and available solution traces to assess whether the task reflects the behavior, limitation, or interaction targeted by the selected node or edge.

Solvability. We check solvability by finding a valid plan that achieves the task goal or having a solver answer the question without access to the reference answer. The solver's answer must match the reference and be supported by evidence from its own retrieval. Only candidates that pass both checks are retained. After deduplication, the accepted tasks form $\mathcal{D}_{\text{syn}}$. Graph annotations guide task construction, whereas task outcomes determine rewards. Success may therefore require adapting or disregarding the selected recommendation.

5 EXPERIMENTS

We evaluate whether turning accumulated skills into training tasks supports further improvements in agent performance and generalization. We also examine how this learning process shapes policy behavior and skill guidance.

5.1 EXPERIMENTAL SETUP

Environments. We consider two environments that require agents to use observations to guide a sequence of actions. In **ALFWorld** (Shridhar et al., 2020), we evaluate on the seen and unseen environment splits and report success rates, with task types grouped into Pick&Place, Transform, and Examine. Pick&Place includes placing one or two objects, Transform includes cleaning, heating, and cooling before placement, and Examine requires examining an object under a light source. In **search-augmented question answering**, agents issue search queries, read retrieved passages, and produce a final answer. Natural Questions (NQ) (Kwiatkowski et al., 2019) and

Table 1: ALFWorld success rates (%) on seen and unseen environments. Results are mean ± standard deviation over three runs. Avg. is the weighted mean of success rates across the six original task types, with weights proportional to their evaluation set sizes.

Model	Method	Seen				Unseen				Overall
		Pick&Place	Transform	Examine	Avg.	Pick&Place	Transform	Examine	Avg.	
Qwen3-4B-Instruct	ReAct	52.0 ±2.6	23.6 ±3.5	33.3 ±4.4	37.1 ±1.9	28.5 ±2.8	22.3 ±2.5	68.5 ±8.5	30.3 ±2.3	33.8 ±0.2
	Static Skill	59.9 ±2.0	41.1 ±1.6	64.1 ±4.4	51.0 ±0.4	47.2 ±2.8	63.7 ±1.6	72.2 ±9.6	59.7 ±2.7	55.2 ±1.1
	w/ RL									
	GRPO	91.5 ±3.4	59.7 ±3.0	51.3 ±11.8	72.4 ±1.6	86.2 ±1.4	68.2 ±2.5	66.7 ±5.6	73.1 ±2.0	72.7 ±1.5
	MemRL	88.1 ±2.9	75.2 ±1.5	66.7 ±4.4	79.8 ±0.8	75.6 ±4.2	72.2 ±4.0	92.6 ±8.5	76.1 ±2.2	78.0 ±0.8
	EvolveR	67.8 ±3.4	83.5 ±4.2	64.1 ±4.4	75.2 ±2.2	65.9 ±7.3	76.4 ±2.3	90.7 ±8.5	75.4 ±4.9	75.3 ±3.5
	INSPO	76.3 ±2.9	73.8 ±3.6	59.0 ±4.4	73.8 ±2.2	77.2 ±1.4	76.4 ±7.7	79.6 ±3.2	77.1 ±3.8	75.4 ±2.8
	SkillRL	88.1 ±3.4	77.3 ±5.0	69.2 ±13.3	81.2 ±3.4	68.3 ±6.5	86.2 ±2.4	81.5 ±3.2	79.9 ±1.5	80.5 ±2.3
	Ours	93.2 ±1.7	80.4 ±3.1	71.8 ±4.4	84.5 ±1.1	76.4 ±7.0	84.7 ±1.5	92.6 ±3.2	83.1 ±2.6	**83.8** ±1.8
Qwen3-8B	ReAct	72.9 ±2.9	42.3 ±2.0	53.8 ±7.7	56.7 ±2.5	64.2 ±7.8	58.3 ±2.2	70.4 ±6.4	61.4 ±3.0	59.0 ±1.9
	Static Skill	72.9 ±3.4	55.0 ±6.0	56.4 ±8.9	62.6 ±4.3	69.9 ±5.1	75.7 ±4.9	74.1 ±3.2	73.4 ±3.5	67.9 ±3.2
	w/ RL									
	GRPO	94.9 ±1.7	64.3 ±2.8	48.7 ±4.4	76.0 ±0.4	92.7 ±2.4	70.2 ±3.6	74.1 ±3.2	77.1 ±2.8	76.5 ±1.2
	MemRL	88.7 ±2.6	79.4 ±2.4	69.2 ±7.7	83.1 ±1.1	75.6 ±4.9	76.0 ±1.4	98.1 ±3.2	78.9 ±1.7	81.0 ±1.3
	EvolveR	71.8 ±3.5	85.2 ±1.1	69.2 ±7.7	78.3 ±1.6	69.1 ±7.0	79.1 ±2.6	88.9 ±5.6	77.4 ±3.8	77.9 ±1.1
	INSPO	75.7 ±1.0	79.1 ±1.9	64.1 ±8.9	76.7 ±1.1	82.9 ±4.2	77.6 ±6.1	85.2 ±6.4	80.3 ±2.4	78.5 ±0.6
	SkillRL	92.1 ±1.0	78.7 ±3.3	74.4 ±8.9	83.8 ±2.5	75.6 ±6.5	86.2 ±4.0	88.9 ±9.6	83.3 ±1.9	83.6 ±0.4
	Ours	94.4 ±1.0	82.1 ±0.9	74.4 ±8.9	86.2 ±0.4	81.3 ±5.1	90.7 ±3.1	90.7 ±11.6	87.3 ±4.2	**86.7** ±1.8



Figure 5: Execution errors and policy–skill bank analysis. (a) Environment rejection and repeated failure rates on unseen ALFWorld tasks during continued RL. (b) Seen/unseen performance with different policies and skill banks on ALFWorld and Search, using Qwen3-4B.

HotpotQA (Yang et al., 2018) provide the training sources and the seen evaluation benchmarks. We evaluate transfer to five unseen benchmarks: PopQA (Mallen et al., 2023), TriviaQA (Joshi et al., 2017), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). More details are in Appendix A.

Models and baselines. We evaluate `Qwen3-4B-Instruct-2507` and `Qwen3-8B`. The baselines without policy RL are ReAct (Yao et al., 2022) and Static Skill, covering reasoning and acting without skill guidance and inference with a fixed skill library. For policy training, we compare with GRPO (Shao et al., 2024), MemRL (Zhang et al., 2026)(with LLM parameter updates), EvolveR (Wu et al., 2025), INSPO (Zhou et al., 2025), and SkillRL (Xia et al., 2026a). These baselines cover policy optimization alone and experience-guided training.

Training with synthesized tasks. We initialize the policy and skill library from the SkillRL checkpoint with the highest overall evaluation score. By default, we continue training on a 1:1 mixture of original and validated synthesized tasks, updating both the policy and skill library. In ALFWorld, rewards come from the environment's goal checker, while search rewards are based on normalized exact match against the reference answers. Training details and hyperparameters are in Appendix B.

Table 2: Search accuracy (%), measured by normalized exact match. Results are mean ± standard deviation over three runs. Overall is a weighted average of benchmark accuracies, with weights proportional to their evaluation set sizes.

Model	Method	Seen		Unseen					Overall
		NQ	HotpotQA	PopQA	TriviaQA	2WikiMHQA	MuSiQue	Bamboogle	
Qwen3-4B-Instruct	ReAct	21.7 ±0.7	30.0 ±0.7	29.7 ±0.7	40.3 ±0.7	31.2 ±0.5	9.0 ±0.7	30.7 ±0.5	27.2 ±0.2
	Static Skill	31.2 ±0.5	35.4 ±0.5	41.7 ±1.5	54.8 ±0.8	30.6 ±1.8	10.3 ±1.0	41.9 ±0.5	34.5 ±0.2
	w/ RL								
	GRPO	48.1 ±3.0	50.9 ±2.5	43.0 ±0.5	64.7 ±0.6	39.3 ±4.3	14.9 ±0.5	40.3 ±3.2	43.3 ±0.2
	MemRL	49.9 ±5.2	51.1 ±4.5	47.7 ±2.8	64.6 ±3.2	41.2 ±6.4	14.4 ±1.0	43.7 ±4.0	44.8 ±1.0
	EvolveR	49.0 ±3.7	50.2 ±0.5	47.6 ±3.0	64.1 ±2.4	39.9 ±2.2	13.4 ±3.4	41.6 ±1.6	43.9 ±0.9
	INSPO	50.6 ±1.8	51.7 ±2.6	49.0 ±2.3	65.6 ±1.8	39.2 ±1.2	13.2 ±1.2	42.4 ±3.5	44.7 ±0.3
	SkillRL	50.8 ±4.2	51.0 ±0.3	48.3 ±4.2	64.8 ±6.3	39.4 ±4.2	15.0 ±2.1	44.5 ±0.9	44.9 ±0.6
	Ours	50.2 ±4.6	53.7 ±4.9	47.9 ±3.0	67.4 ±2.8	43.1 ±1.2	14.6 ±2.3	46.9 ±4.1	**46.2 ±0.5**
Qwen3-8B	ReAct	30.4 ±0.5	31.7 ±0.7	37.6 ±0.7	51.8 ±0.5	31.3 ±0.7	9.4 ±0.5	35.7 ±0.5	32.3 ±0.2
	Static Skill	32.2 ±0.5	38.6 ±1.2	42.0 ±0.5	56.4 ±0.8	30.0 ±1.3	10.6 ±0.5	40.3 ±1.2	35.3 ±0.1
	w/ RL								
	GRPO	48.1 ±4.5	52.0 ±0.7	51.8 ±3.8	63.8 ±1.3	40.8 ±1.6	16.2 ±1.5	43.5 ±2.0	45.3 ±0.7
	MemRL	47.9 ±2.5	52.6 ±1.1	51.9 ±1.4	63.7 ±4.6	40.1 ±2.9	16.9 ±0.2	44.3 ±5.1	45.4 ±0.0
	EvolveR	48.0 ±3.5	51.9 ±2.1	51.6 ±4.1	64.0 ±3.6	42.4 ±3.3	16.3 ±1.8	43.7 ±6.8	45.6 ±0.6
	INSPO	48.2 ±2.2	52.8 ±2.7	52.1 ±1.7	64.4 ±2.0	41.7 ±3.8	16.4 ±1.3	44.0 ±5.0	45.8 ±0.6
	SkillRL	48.7 ±0.6	53.0 ±1.2	52.4 ±2.7	64.6 ±2.0	42.0 ±1.2	17.2 ±1.9	44.8 ±6.2	46.2 ±0.5
	Ours	48.3 ±2.4	56.6 ±2.8	54.5 ±2.9	64.1 ±1.6	44.7 ±3.2	19.6 ±2.0	47.2 ±6.8	**47.9 ±2.0**

5.2 Main Results

Synthesized tasks provide further gains over skill-guided RL. Our method outperforms all baselines in overall performance across both environments and model scales (Tables 1 and 2). Building on SkillRL, it raises ALFWorld success by 3.3 and 3.1 percentage points at 4B and 8B. In search, our method achieves overall accuracies of 46.2% at 4B and 47.9% at 8B, compared with SkillRL's 44.9% and 46.2%. These results demonstrate the value of turning accumulated skills into new training tasks, enabling further gains beyond skill-guided RL in both settings.

Gains extend to unseen environments and benchmarks. Our method delivers clear gains on unseen tasks. Compared with SkillRL, our 4B model improves unseen ALFWorld success from 68.3% to 76.4% on Pick&Place and from 81.5% to 92.6% on Examine. The benefits also extend to held-out search benchmarks, where accuracy rises from 39.4% to 43.1% on 2WikiMultiHopQA and from 44.5% to 46.9% on Bamboogle. These results demonstrate improved generalization beyond the training environments and benchmarks.

5.3 How Does Continued Training Affect Execution and Skill Guidance?

We examine how training on synthesized tasks improves execution reliability and the skill guidance available to the policy. The quantitative analyses use single runs with the 4B model.

The agent executes tasks more reliably. From a shared SkillRL initialization, our method reduces execution errors more than standard SkillRL continuation (Figure 5(a)). We track the fraction of unseen ALFWorld episodes containing at least one rejected action (environment rejection) or at least three consecutive identical rejected actions (repeated failure). At step 60, environment rejection is 24.1% for our method versus 29.7% for SkillRL, a reduction of 5.6 percentage points, while repeated failure is 4.8 points lower. The gains in task success are thus accompanied by more reliable execution, with fewer episodes containing invalid actions or repeated unsuccessful steps.

Policy changes and skill-bank updates both contribute to performance. With our final bank fixed, our policy outperforms the best policy from standard SkillRL continuation (Figure 5(b)),

More explicit action ordering

Context. A reachable target may already be visible while containers remain unchecked.

Past guidance

`gen_002` (visible target): **Take a visible target before moving elsewhere.**

`dyn_031` (container search): **Open and inspect a container** before judging its contents or moving on.

Added guidance

`dyn_067`: **Check for visible targets on open surfaces before opening containers**.

Figure 6: An added skill coordinates earlier guidance by prioritizing visible-target checks before container search. Skill descriptions are summarized.

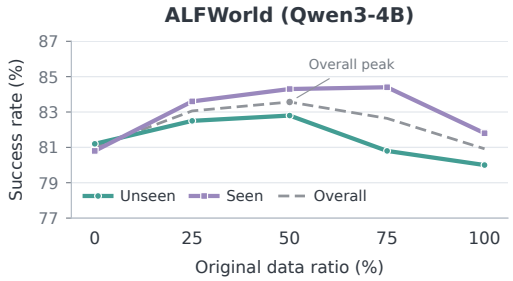


Figure 7: Effect of the original-task fraction.

Table 3: Ablation success rates (%) on ALFWorld with Qwen3-4B-Instruct.

Variant	Seen	Unseen	Overall
Ours	84.3	82.8	**83.6**
Policy initialization			
From initial model	79.3	78.4	78.8
Task synthesis			
w/o node tasks	80.7	81.3	81.0
w/o relation tasks	82.9	80.6	81.8

showing that updated guidance alone does not explain the gains. The same policy also benefits from updated guidance: replacing the bank from before continuation with our final bank further improves performance. Figure 8 shows a refinement: a new skill explicitly orders visible-target checks before container search, coordinating existing recommendations. More examples appear in Appendix C.

5.4 Ablation Study

We examine the contributions of the two synthesis branches and the effect of mixing synthesized and original tasks, using Qwen3-4B in ALFWorld.

Task composition affects continued-training gains. To examine how task composition affects continued training, we vary the original-task fraction (Figure 7). The equal mixture outperforms both original-only and synthetic-only continuation on seen and unseen environments and achieves the highest unseen success. Increasing the original-task fraction to 75% slightly improves seen success but lowers unseen success, supporting 50% as a balanced choice.

Relation-conditioned tasks improve on single-skill synthesis. The *w/o relation tasks* variant retains only node-conditioned synthesis: each seed-task rewrite is guided by one skill's attributes and assessment, without using inter-skill edges. It therefore provides a single-skill synthesis baseline without relational graph structure, while retaining goal construction and task validation. Compared with this baseline, the full method raises overall success from 81.8% to 83.6% and unseen success from 80.6% to 82.8% (Table 3). This comparison supports the benefit of relation-conditioned task requirements beyond those derived from individual skills. And the results show that the full graph-guided approach achieves the best performance.

6 Conclusion

We introduced Skill2Task, which uses skill behaviors, applicability conditions, and relations to synthesize tasks for continued agentic RL. Training on these tasks improves both the policy and skill library: policy adaptation supports transfer to unseen tasks, while refined guidance provides additional gains. Models trained on our synthesized tasks achieved higher average performance on ALFWorld and search-augmented question answering at both 4B and 8B scales. These results highlight how turning accumulated experience into new learning challenges can broaden agent capabilities.

AI USE STATEMENT

In this work, our use of generative AI tools as authoring aids was limited to writing assistance and language polishing. We have not used these tools to develop conceptual frameworks, formulate or refine research hypotheses, design experiments, implement methods, or interpret results. Tasks involving the formulation or proof of mathematical claims are not applicable to this work. For tasks with recommended disclosure, we used generative AI tools to improve grammar, wording, sentence flow, and readability. The use of language models within our research method and experiments is described in the paper. We have reviewed all AI-assisted text, checking that the revised wording preserves the intended meaning and accurately reflects our methods and findings. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Tong Bai, Zhenglin Wan, Pengfei Zhou, Xingrui Yu, Yang You, and Ivor W. Tsang. SkillDAG: Self-Evolving Typed Skill Graphs for LLM Skill Selection at Scale. arXiv preprint arXiv:2606.03056, 2026. URL https://arxiv.org/abs/2606.03056.

Shiqi Chen, Jingze Gai, Ruochen Zhou, Jinghan Zhang, Tongyao Zhu, Junlong Li, Kangrui Wang, Zihan Wang, Zhengyu Chen, Klara Kaleb, et al. Skillcraft: Can llm agents learn to use tools skillfully? *arXiv preprint arXiv:2603.00718*, 2026.

Zihao Cheng, Hongru Wang, Zeming Liu, Xinyi Wang, Xiangrong Zhu, Yuhang Guo, Wei Lin, Jeff Z. Pan, and Yunhong Wang. Terminal-World: Scaling Terminal-Agent Environments via Agent Skills. arXiv preprint arXiv:2605.20876, 2026. URL https://arxiv.org/abs/2605.20876.

Zhiyuan Fan, Tinghao Yu, Yuanjun Cai, Jiangtao Guan, Yun Yang, Dingxin Hu, Jiang Zhou, Xing Wu, Zhuo Han, Feng Zhang, and Lilin Wang. Toward Scalable Terminal Task Synthesis via Skill Graphs. arXiv preprint arXiv:2604.25727, 2026. URL https://arxiv.org/abs/2604.25727.

Zelin He, Haotian Lin, Boran Han, Wei Zhu, Haoyang Fang, Bernie Wang, Xuan Zhu, Runze Li, and Matthew Reimherr. ReSkill: Reconciling Skill Creation with Policy Optimization in Agentic RL. arXiv preprint arXiv:2606.01619, 2026. URL https://arxiv.org/abs/2606.01619.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Siyuan Huang, Pengyu Cheng, Haotian Liu, Tao Chen, Yihao Liu, Jingwei Ni, Shijie Zhou, Ziyi Yang, Gangwei Jiang, Mengyu Zhou, Yu Cheng, Xiaoxi Jiang, and Guanjun Jiang. Skill Self-Play: Pushing the Frontier of LLM Capability with Co-Evolving Skills. arXiv preprint arXiv:2607.22529, 2026. URL https://arxiv.org/abs/2607.22529.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Xiaoyuan Li, Moxin Li, Keqin Bao, Yubo Ma, Wenjie Wang, Dayiheng Liu, and Fuli Feng. Skillgraph: Skill-augmented reinforcement learning for agents via evolving skill graphs. *arXiv preprint arXiv:2605.12039*, 2026a.

Zhiyuan Li, Linyuan Gao, Xuechun Ding, Hongwei Chen, Yuan Wu, and Yi Chang. CaSKG: Counterfactual-Causal Skill Graphs for Scalable Agent Skill Retrieval. arXiv preprint arXiv:2608.25500, 2026b. URL https://arxiv.org/abs/2608.25500.

Dawei Liu, Zongxia Li, Hongyang Du, Xiyang Wu, Shihang Gui, Yongbei Kuang, and Lichao Sun. Graph-of-Skills: Dependency-Aware Structural Retrieval for Massive Agent Skills. arXiv preprint arXiv:2604.05333, 2026. URL https://arxiv.org/abs/2604.05333.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

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

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Zelin Tan, Yiqun Zhang, Hao Li, Zhiyao Cui, Hejia Geng, Shao Zhang, Hangfan Zhang, Yang Chen, Xiaosong Wang, Lilong Wang, Zhenfei Yin, Shuyue Hu, Chen Zhang, and Lei Bai. SKT: Skill-Use Training at Scale via Verified Synthetic Data Generation. arXiv preprint arXiv:2608.02287, 2026. URL https://arxiv.org/abs/2608.02287.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Songjun Tu, Chengdong Xu, Qichao Zhang, Yiwen Ma, Yaocheng Zhang, Linjing Li, Dong Li, Xiangyuan Lan, and Dongbin Zhao. UCOB: Learning to Utilize and Evolve Agentic Skills via Credit-Aware On-Policy Bidirectional Self-Distillation. arXiv preprint arXiv:2606.29502, 2026. URL https://arxiv.org/abs/2606.29502.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Trans. Mach. Learn. Res.*, 2024, 2024.

Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, et al. Evolver: Self-evolving llm agents through an experience-driven lifecycle. *arXiv preprint arXiv:2510.16079*, 2025.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, et al. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*, 2026a.

Tianle Xia, Lingxiang Hu, Yiding Sun, Ming Xu, Lan Xu, Siying Wang, Wei Xu, and Jie Jiang. GraSP: Graph-Structured Skill Compositions for LLM Agents. arXiv preprint arXiv:2604.17870, 2026b. URL https://arxiv.org/abs/2604.17870.

Chuan Xiao, Zhengbo Jiao, Shaobo Wang, Wei Wang, Bing Zhao, Hu Wei, Linfeng Zhang, and Lin Qu. Socratic-SWE: Self-Evolving Coding Agents via Trace-Derived Agent Skills. arXiv preprint arXiv:2606.07412, 2026. URL https://arxiv.org/abs/2606.07412.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, et al. Memrl: Self-evolving agents via runtime reinforcement learning on episodic memory. *arXiv preprint arXiv:2601.03192*, 2026.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL `https://ojs.aaai.org/index.php/AAAI/article/view/29936`.

Han Zhou, Xingchen Wan, Ivan Vulić, and Anna Korhonen. Agentic policy optimization via instruction-policy co-evolution. *arXiv preprint arXiv:2512.01945*, 2025.

APPENDIX

A BENCHMARK DETAILS

ALFWorld (Shridhar et al., 2020) provides text-based household tasks aligned with the embodied environments in ALFRED. Agents explore scenes and manipulate objects to place one or two objects, clean, heat, or cool an object before placement, or examine an object under a light source; we report task success on seen and unseen environments.

Search benchmarks. We use NQ and HotpotQA as training sources and seen evaluation benchmarks, and hold out PopQA, TriviaQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle for unseen evaluation. Across these benchmarks, agents retrieve passages through search and are evaluated by normalized exact match against reference answers.

Natural Questions (NQ) (Kwiatkowski et al., 2019) contains real Google Search queries with answers annotated in Wikipedia. Its naturally occurring questions evaluate the ability to locate relevant evidence and recover factual answers.

HotpotQA (Yang et al., 2018) contains Wikipedia-based questions that require combining information across documents, including questions that connect entities or compare their attributes. Sentence-level supporting facts identify the evidence needed to answer each question.

PopQA (Mallen et al., 2023) consists of factual questions derived from Wikidata triples, with entities spanning different levels of popularity. Its coverage of less popular entities tests access to knowledge that may be difficult to recover from model parameters alone.

TriviaQA (Joshi et al., 2017) pairs questions written by trivia enthusiasts with independently collected evidence from Wikipedia and the web. The questions cover diverse topics and often require matching clues in the question to information in retrieved passages.

2WikiMultiHopQA (Ho et al., 2020) combines Wikipedia text and Wikidata relations to construct multi-hop questions with explicit reasoning paths. It evaluates the ability to connect intermediate facts and follow the dependencies needed to reach an answer.

MuSiQue (Trivedi et al., 2022) composes single-hop questions into connected questions requiring two to four reasoning steps. Its construction filters shortcuts, emphasizing cases where a later step depends on information obtained earlier.

Bamboogle (Press et al., 2023) contains manually written two-hop questions selected because direct search queries produced incorrect answer snippets during dataset construction. It tests whether an agent can decompose a question and combine the resulting facts into a correct answer.

B TRAINING DETAILS

We conduct policy training on eight NVIDIA A800 GPUs. Our continued-training stage starts from the selected SkillRL checkpoint with a learning rate of 10^{-6}. For ALFWorld, we use 16 tasks per training batch, eight rollouts per task. For search, the corresponding settings are 256 tasks, four rollouts per task. The maximum prompt and response lengths are 8,192 and 1,024 tokens for ALFWorld, and 4096 and 512 tokens for search. Episodes are limited to 50 interaction steps in ALFWorld and 16 in search.

For all baselines, we follow the training hyperparameters and experimental settings reported in their original papers, except that methods requiring a teacher use GLM-5.2 (Zeng et al., 2026) in place of the original teacher model. The policy backbones remain the Qwen models specified in Section 5.1.

C ADDITIONAL SKILL CASES

Figure 8 extends the case study in Section 5.3 with three forms of added guidance. The first qualifies an absolute location claim, the second orders visible-target checks before container search, and the third specifies a tie-break when observations do not distinguish candidate receptacles. These additions make earlier recommendations more explicit about uncertainty, action order, and applica-

(a) More cautious location inference

Context. A temperature-modified target has an unknown location; the agent is considering the matching appliance as its first search location.

Past guidance	**Added guidance**
`dyn_043` "The object is stored at room temperature in cabinets, drawers, or shelves, **not inside the appliance**."	`dyn_063` "A 'cool apple' is **not necessarily in the fridge**; a 'hot cup' is **not necessarily in the microwave**. Always search standard storage locations first."

Refinement. The added guidance expresses location uncertainty while retaining a preference for searching standard storage locations first.

(b) More explicit action ordering

Context. Open surfaces and closed containers are both search candidates; the current observation may already show a reachable target.

Past guidance	**Added guidance**
`gen_002` (visible target) "**take it before moving elsewhere**" `dyn_031` (container search) "you **must open it and examine its contents** before making any judgment about what is inside or moving on."	`dyn_067` "**Before opening any container** to search for an object, **first check if the target is visible** on any open surface in the current observation."

Refinement. The added rule specifies a local order: check for visible targets before opening containers, making the interaction between the earlier recommendations explicit.

(c) A default with specific applicability conditions

Context. The agent must choose among multiple receptacles of the same type, and no observation distinguishes the candidates.

Past guidance	**Added guidance**
`dyn_035` "After identifying the target and a plausible search location, **commit to that action** without exploring all alternatives exhaustively."	`dyn_079` "When **multiple identical receptacles exist and no observation distinguishes them**, always navigate to the **lowest-numbered one first** to prevent reasoning loops and decision paralysis."

Refinement. The added rule supplies a concrete tie-break, specifying both when the default applies and which search action to take.

Figure 8: Case study of compensatory skill additions.

bility. They illustrate how the library can refine its guidance through complementary rules, without implying that every earlier limitation has been resolved.

D Additional Skill-Library Analysis

Figure 9 demonstrates three complementary views of the evolving skill library. The displayed trends show lower proportions of unreliable skills, growth dominated by Support and Prerequisite edges, and higher skill–query similarity, with larger changes on Pick two and Look tasks. These views characterize teacher-assessed reliability, relational structure, and retrieval relevance in the evolving skill library.

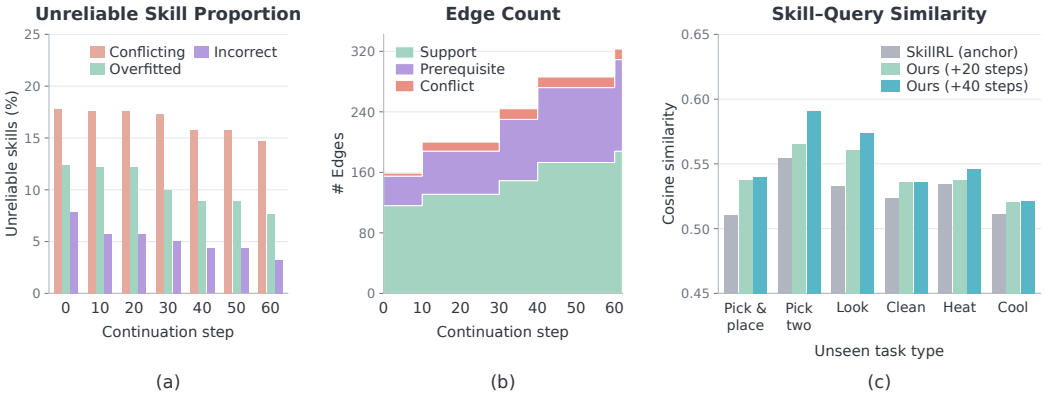


Figure 9: Skill-library trends during continued training. (a) Proportions of skills flagged as conflicting, overfitted, or incorrect. (b) Edge counts by relation type. (c) Mean cosine similarity between unseen task queries and their top-six retrieved general skills, grouped by task type.

E Representative Teacher Prompts

We provide templates for GLM-5.2 (Zeng et al., 2026): one for each synthesis branch and one for independent target-relevance review. The inputs include the graph annotations from Section 4.1; braces mark instance-specific fields.

Node-conditioned task synthesis prompt

Create one new task that exercises or tests the selected skill within the seed's permitted edits.

Inputs

Seed task and permitted edits: {seed_task, allowed_edits}

Skill conditions, behavior, effect, status, and exceptions: {skill_node}

Environment rules, legal goals, or source passages: {grounding}

Follow the node status.

– **Reliable**: vary instance details within the assessed scope, preserving the task type and the skill's contribution.

– **Conditional**: satisfy and identify the additional preconditions.

– **Questionable**: test the disputed assumption in a feasible setting; identify feedback that would support or contradict it, and a valid alternative if needed.

Ground the task. For ALFWorld, select a legal goal without changing scene objects or initial state. For search, derive a new factual question and an unambiguous short answer from the passages. Change the requirement or information sought, not merely the wording. Do not invent facts or capabilities, expose construction notes, or reveal the answer in the question.

Node-conditioned task synthesis prompt (continued)

Return only JSON: `status` (`proposed`/`reject`), `tasks` (array: one task or empty on rejection), `changes` (seed edits), `target_check`, `reason`. Task fields: `instruction`, `goal_or_answer`, supplied `grounding_ids`. Record the condition, decision, and skill use/adaptation in `target_check`; explain rejections in `reason`.

Relation-conditioned task synthesis prompt

Create one new task within permitted seed edits. Realize the skill relation through solution requirements, not merely by mentioning both skills.

Inputs

Seed task and permitted edits: {seed_task, allowed_edits}

Skills, assessments, relation, and direction or conflict conditions: {skill_edge}

Environment rules, legal goals, or source passages: {grounding}

Follow the relation.

– **Support**: return one task with a shared goal and distinct, compatible contributions from both skills.

– **Prerequisite**: return one task where the enabling skill establishes a state or fact required by the dependent skill. Identify the result and its use; ordering independent actions is insufficient.

– **Conflict**: return one task exposing incompatible recommendations at the same decision point. Identify overlapping stated conditions and why the choice affects goal completion.

Ground the task. For ALFWorld, select legal goals while preserving scene inventory and initial state. For search, ask for new information and jointly ground the question, its unambiguous short answer, and every intermediate fact in supplied passages. Preserve entity identity, time scope, and comparison units; exclude construction notes, answer leakage, and strategy hints.

Return only JSON: `status` (`proposed`/`reject`), `tasks` (array: one task or empty on rejection), `changes` (seed edits), `target_check`, `reason`. Task fields: `instruction`, `goal_or_answer`, supplied `grounding_ids`. Record skill roles and relation evidence in `target_check`; explain rejections in `reason`.

Independent target-relevance review prompt

Review one candidate task's grounding and target relevance without rewriting it. Check the generator's claims against supplied evidence.

Inputs

Single task and synthesis target: {candidate, target}

Seed, permitted edits, rules, and grounding: {source_context}

Available traces or retrieved evidence: {solution_evidence}

Check grounding. Verify instruction–answer or instruction–goal consistency, seed constraints, environment rules, and support for required objects, facts, and intermediate steps. For search, check scope, answer uniqueness, and a new information need. Mentioning an answer does not establish that it answers the question.

Locate the target in a concrete decision or step.

– **Reliable/Conditional**: applicable condition and supported behavior.

– **Questionable**: disputed assumption, testing evidence, and alternative if it fails.

– **Support**: distinct contributions to a shared goal.

– **Prerequisite**: enabling result and dependent use.

– **Conflict**: overlapping stated conditions within the task, incompatible recommendations at a concrete decision point, and evidence that the disputed choice affects goal completion.

Independent target-relevance review prompt (continued)

Reject targets supported only by wording or generator claims. Alternative valid solutions alone do not invalidate relevance. Mark missing evidence `uncertain`; approve the task only if both grounding and target relevance pass.

Return only JSON: `grounding` and `target_relevance` verdicts (`pass`/`fail`/`uncertain`), `evidence_refs`, `reason`. Cite supplied states, passages, or trace steps. Solvability requires separate planner or blind solver verification.

F Task Synthesis Case Studies

Figure 10 illustrates three skill relations through their task queries. Support combines object and lighting requirements, and Prerequisite makes retrieved values the inputs to comparison. The Conflict example exposes incompatible placement decisions within the same two-object goal.

(a) ALFWorld: Support

`exa_002`: Grab Target First
When the target is visible but the desk lamp has not been found, **take the target** so it can be carried to the lamp.

`loo_002`: Switch Lamp On
On reaching the desk lamp, **use it to satisfy the light condition**, before or immediately after handling the target.

Relation. `exa_002` $\xleftrightarrow{\text{Support}}$ `loo_002`. One skill secures the target object; the other establishes the lighting condition. Both contribute to the same examination goal.

Synthesized query. "examine the pen with the desklamp"

(b) Search: Prerequisite

`mul_003`: Collect-Then-Compare
Retrieve concrete values for all items before drawing a comparative conclusion, avoiding premature or unsupported answers.

`com_005`: Explicit Comparative Reasoning
State each normalized value and **perform the logical comparison explicitly** to derive a traceable answer.

Relation. `mul_003` $\xrightarrow{\text{Prerequisite}}$ `com_005`. The first skill retrieves the two elevations, providing the factual inputs required by the second skill's comparison.

Synthesized query. "Which entity has the lower value when comparing the elevation of Sia La measured in meter and the elevation of Bilafond La measured in meter: Sia La or Bilafond La?"

(c) ALFWorld: Conflict

`dyn_024`: Collect All Before Any Place
For a goal requiring multiple objects, **collect every required item before placing any**; all items must first be in hand.

`pic_004`: Place Before More Search
When holding a required object and the destination is known and reachable, **place it immediately**, then resume searching for any remaining items.

Relation. `dyn_024` $\xleftrightarrow{\text{Conflict}}$ `pic_004`. With the first keychain in hand and the second still needed, one skill **forbids placement** while the other **requires placement now**. ALFWorld's empty-hand pickup rule prevents collecting the second item while holding the first.

Synthesized query. "put two keychains on the dining table"

Figure 10: Skill relations and task queries. Skill descriptions are shortened.