Skill is Not One-Size-Fits-All: Model-Aware Skill Alignment for LLM Agents

Anonymous ACL submission

Abstract

LLM agents increasingly retrieve externally curated *skills*—procedural instructions retrieved at decision time—to improve performance on long-horizon interactive tasks. Existing skill libraries are typically treated as model-agnostic, reusing the same skill formulations across backbones with substantially different capacities and behaviors. However, our controlled experiments across multiple model scales show that skill effectiveness is strongly model-dependent: a skill that benefits one backbone can harm another. Motivated by this observation, we propose MASA (*Model-Aware Skill Alignment*), a framework that adapts skills to each target backbone without modifying agent weights. MASA operates in two stages: (1) a hierarchical skill evolution pipeline that iteratively rewrites general and task-specific skills using hill climbing and UCB-driven tree search, guided by environment feedback and model capability profiles; and (2) a lightweight model-conditioned skill rewriter trained on evolution trajectories to reproduce the adaptation in a single forward pass. Experiments across three interactive environments and four backbones show that MASA consistently achieves the best overall performance, with gains of up to 25.8 points over the strongest baseline. The learned rewriter further generalizes to unseen tasks and environments without additional search, consistently outperforming a much larger teacher LLM at a fraction of the inference cost. Our code is publicly available.[1]

1 Introduction

LLM agents increasingly solve long-horizon interactive tasks, including web navigation (Ouyang et al., 2026), embodied control (Lu et al., 2026), and tool use (Schick et al., 2023; Jiang et al., 2026; Wang et al., 2024a; Hsiao et al., 2025). A common approach to steer these agents without modifying model weights is to retrieve short pieces of procedural knowledge—which we call *skills*—from an external library at each step (Wang et al., 2023, 2024b,c; Chen et al., 2024; Zhao et al., 2024; Ma et al., 2026). Existing skill-library systems, whether hand-crafted (Zhu et al., 2023) or distilled from agent trajectories (Zhao et al., 2024; Chen et al., 2024; Xia et al., 2026; Wang et al., 2025a), typically construct a single shared library and reuse it across different LLM backbones. In practice, deployment constraints such as latency budgets, inference cost, and hardware availability mean that real-world agent systems must operate with backbones of vastly different scales rather than simply relying on the strongest available model (Yao et al., 2025; Zheng et al., 2025). This deployment heterogeneity raises a critical question for skill-library design: can a single skill formulation serve models with substantially different capacities equally well?

To examine this, we experiment on ALFWorl (Shridhar et al., 2020) (full setup and analysis in §2): keeping the principles of a skill library fixed, we vary only its granularity and evaluate four Qwen3 backbones (4B–32B) (Yang et al., 2025). As Figure 1 shows, the optimal granularity varies

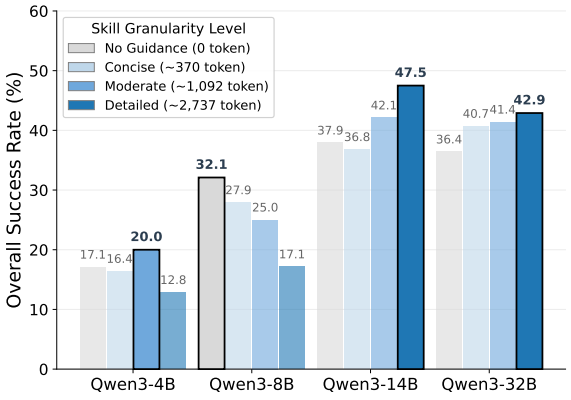

Figure 1: **Skill granularity is not one-size-fits-all.** ALFWorld success rate (%) for four Qwen3 backbones under a `No-Skill` control and three granularity levels (`Concise`, `Moderate`, `Detailed`). The optimal level differs across backbones.

[1] https://anonymous.4open.science/r/MASA-BE5F/

across models; indeed, a skill that boosts one backbone can actively degrade another. A parallel experiment on the Gemma3 family (Appendix C.3) confirms that the same pattern holds across families, and that models of the same size but from different families also prefer different skill formulations. This observation suggests that the effectiveness of a skill library depends not only on what knowledge it encodes, but also on how that knowledge is expressed relative to the target model's capacity: when the expression is misaligned, retrieved skills distract rather than help. A well-designed skill library should amplify the strengths of its target backbone, unlocking capabilities that generic, model-agnostic skills cannot.

We pursue this goal with **MASA**, **M**odel-**A**ware **S**kill **A**lignment, a framework that aligns the formulation of a skill library with each target backbone without modifying agent weights. MASA treats skill alignment as a hierarchical search problem driven by environment feedback. It first runs a *hierarchical model-conditioned skill evolution*: a stronger teacher LLM iteratively rewrites skills guided by a capability profile of the target model, applying hill-climbing over general skills and UCB-driven tree search over task-specific skills. To eliminate the costly teacher at deployment, the discovered rewrites train a lightweight model-conditioned skill rewriter that adapts skills in a single forward pass, outperforming the teacher while being orders of magnitude cheaper.

Our main contributions are as follows:

- We empirically demonstrate that different models require different skill formulations: the same skill library that benefits one backbone can actively degrade another. This finding challenges the one-size-fits-all assumption and motivates model-aware skill alignment.
- We propose MASA, a framework that aligns skill formulations with each target backbone. It combines iterative search to evolve optimal skills with a lightweight rewriter that transforms unaligned skills into model-appropriate ones.
- We evaluate MASA across three diverse environments and four Qwen3 backbones, achieving the highest success rate with gains up to $+25.8$ points. MASA-rewriter further generalizes to unseen tasks and environments in a single forward pass, outperforming a much larger teacher LLM at negligible cost.

2 Preliminary Study: One Skill Library Does Not Fit All

Before introducing MASA, we ask whether a single skill library serves all model scales equally. To isolate the effect of skill form from skill content, we keep the underlying *principles* fixed and vary only the *granularity* of their textual expression.

2.1 Setup

We use ALFWorld (Shridhar et al., 2020), a text-based household task suite spanning six task types, and evaluate on the validation split. We compare four Qwen3 backbones (4B/8B/14B/32B) (Yang et al., 2025) that differ primarily in capacity while sharing the same architecture and training regimen. We design one `No Skill` control and three skill-granularity levels that encode identical behavioral principles but differ in representational depth. Following prior work, we adopt the skill library of Xia et al. (2026) as the `Moderate` variant and construct the `Concise` and `Detailed` variants through controlled rewriting that preserves the underlying principles while adjusting granularity (see Table 4 in Appendix C.1 for side-by-side examples). All three levels use the same retrieval pipeline, ensuring that observed differences are attributable to granularity alone.

2.2 Findings

Figure 1 reports overall ALFWorld success rates.

Finding 1: The optimal skill form is model-dependent, and mismatches can hurt. No single granularity level is uniformly optimal across models. Qwen3-4B performs best with `Moderate skills` while Qwen3-14B and Qwen3-32B achieve their highest scores with `Detailed skills`. Notably, Qwen3-8B performs best under the `No Skill` condition (32.1%) and all three skill variants reduce performance. Importantly, this does not suggest that skills are inherently incompatible with Qwen3-8B. Trajectory inspection reveals that, without external guidance, Qwen3-8B often follows short and effective action chains that directly solve the task. Misaligned skills instead introduce procedural reasoning patterns that override these naturally concise action chains, causing the model to over-explore or deliberate unnecessarily. This suggests that the effectiveness of a skill depends not only on its content but also on whether its expression is compatible with the model's default problem-solving strategy.

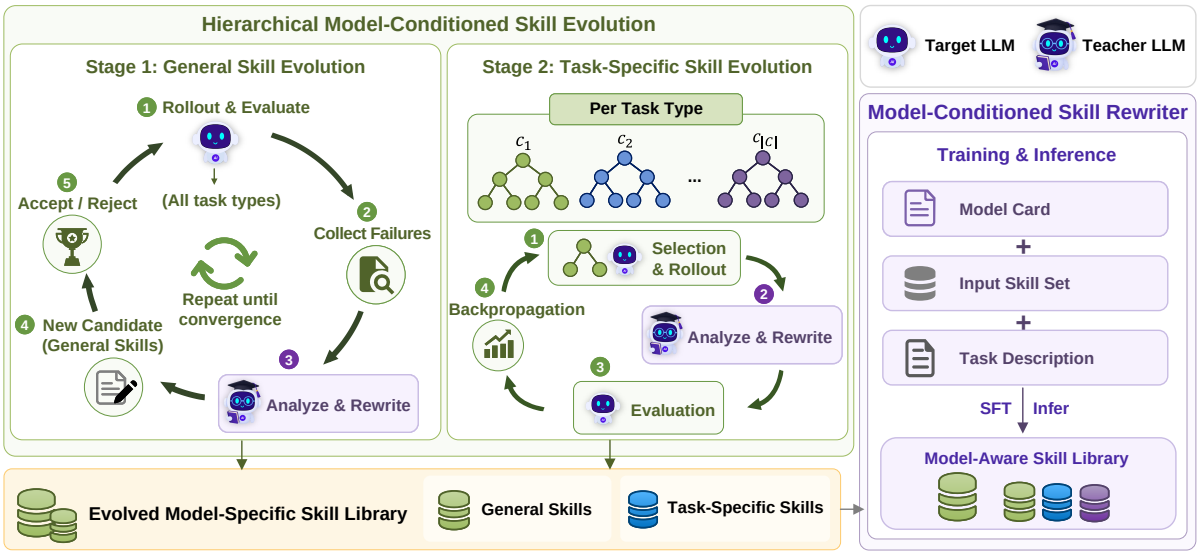

Figure 2: The overall framework of MASA.

Finding 2: The granularity–performance relationship is non-monotonic and defies simple heuristics. It is unclear how skill granularity should scale with model capability: smaller models may benefit from concise guidance due to limited context utilization capacity, yet they may also require more explicit procedural supervision because of weaker reasoning abilities. Our results show that neither direction holds consistently. Qwen3-32B underperforms Qwen3-14B by 4.6 points under `Detailed` despite being twice the size, inverting the usual scaling trend. For Qwen3-4B, performance does not monotonically improve in either direction: `Moderate` outperforms both `Concise` and `Detailed`, indicating that the optimum lies at an intermediate level that cannot be reached by simply "adding more detail" or "stripping to minimum." This complexity necessitates search-based rather than rule-based skill adaptation.

Finding 3: Performance varies sharply across task types. Per-task breakdown (Appendix C.2) reveals that within a given model–granularity pairing, success rates can vary by over 60 points across task types—a spread far exceeding the differences between granularity levels for any single task. For example, Qwen3-14B with `Concise skills` scores 74.2 on PICK but only 13.7 on COOL. Some task types benefit from detailed skills regardless of model size, while others are insensitive or even harmed. This heterogeneity suggests that global optimization alone is insufficient—skill alignment must also operate at the task-type level to address the distinct demands of each task type.

A parallel experiment on the Gemma3 family (4B/12B/27B) reveals the same scale-dependent trend (Appendix C.3), suggesting the phenomenon generalizes across model families.

Implications. Together, the three findings impose concrete design requirements:

(i) *Model-conditioned:* the optimal skill form varies per backbone, so alignment must be explicitly conditioned on the target model's capacity (Finding 1).

(ii) *Search-based rather than heuristic:* the relationship between skill granularity and performance is non-monotonic and model-specific, ruling out simple alignment rules (Finding 2).

(iii) *Task-type-specific:* within the same backbone, different task types respond differently to the same skills, requiring per-task-type adaptation in addition to global optimization (Finding 3).

We further note that our controlled study varies only one axis of skill form (textual granularity) while holding content fixed. In practice, misalignment can also arise from differences in decision strategy, framing, or format, suggesting that a complete solution must perform open-ended, model-aware rewriting. MASA is designed to address all three requirements.

3 Method: MASA

We present MASA, a framework that *conditions* skill evolution on the capability profile of a target backbone, yielding skill libraries specifically

adapted to each model rather than relying on a universal, model-agnostic formulation. MASA comprises two complementary components: a search-time skill evolution pipeline (Section 3.2) that evolves skills under explicit capability conditioning provided by a structured *model card*, and a deployment-time skill rewriter (Section 3.3) that learns this model-conditioned rewriting policy and adapts new skills in a single forward pass. An overview of the framework is shown in Figure 2.

3.1 Problem Formulation and Skill Library

Agent setup. A frozen LLM agent F interacts with environment $\mathcal{E}$. At each step t, the agent receives observation o_t, retrieves relevant skills from a skill library $\mathcal{S}$, and produces an action a_t :

$$a_t \sim F\Big(\cdot \mid \tau_{<t},\ \hat{\mathcal{S}}_t\Big), \quad \hat{\mathcal{S}}_t = \mathrm{TopK}(\mathcal{S}, o_t, k), \tag{1}$$

where $\tau_{<t} = (o_1, a_1, \ldots, o_{t-1}, a_{t-1})$ is the interaction history and TopK retrieves the k most relevant skills by cosine similarity. The backbone F remains frozen throughout, and the sole optimization variable is the skill library $\mathcal{S}$.

Hierarchical skill library. Following Xia et al. (2026), we structure $\mathcal{S}$ into two levels: *general skills* $\mathcal{S}^G$ (cross-task strategy principles) and *task-specific skills* $\mathcal{S}^T = \{\mathcal{S}^{T_c}\}_{c \in \mathcal{C}}$, where each $\mathcal{S}^{T_c}$ contains action guidelines tailored to task type c and $\mathcal{C}$ is the set of all task types. At inference, a lightweight encoder (Qwen3-Embedding-0.6B) separately retrieves top-k_G general skills and top-k_T task-specific skills for the current observation.

Model card. The key conditioning signal in MASA is the *model card* $\mathcal{M}_F$, a structured profile of a target backbone F. Each card contains: (i) *architecture metadata* (model family, parameter count, layer/attention configuration, context window), (ii) *training provenance* (training data scale, multilingual support), and (iii) *capability profile* (strengths and weaknesses of the backbone). The construction protocol is detailed in Appendix D.

Objective. We define a per-episode adjusted reward R that balances task completion against skill-induced stalling:

$$R(F, \mathcal{S}, e) = \mathrm{SR}(F, \mathcal{S}, e) - \lambda \cdot \mathrm{NHR}(F, \mathcal{S}, e), \tag{2}$$

where e denotes a single episode, $\mathrm{SR} \in \{0, 1\}$ is task success, NHR is the *nothing-happens rate*—the fraction of steps after which the environment state remains unchanged, serving as a proxy for skill-induced stalling (e.g., the agent repeatedly issuing ineffective or invalid actions), and $\lambda \in [0, 1]$ controls the penalty strength. The overall optimization objective seeks the skill library maximizing expected adjusted reward over a set of evaluation episodes $\mathcal{D}$:

$$\mathcal{S}_F^\star = \underset{\mathcal{S}}{\mathrm{argmax}}\ \mathbb{E}_{e \sim \mathcal{D}}[R(F, \mathcal{S}, e)]\,, \tag{3}$$

where $\mathcal{S}_F^\star$ denotes the optimal skill library adapted to backbone F.

3.2 Hierarchical Model-Conditioned Skill Evolution

The skill evolution pipeline is a teacher-driven search over skill texts. A stronger *teacher* LLM T (i) analyzes failure trajectories of F to produce a structured failure attribution and (ii) rewrites skills conditioned on the model card $\mathcal{M}_F$, steering edits toward formulations compatible with F's observed characteristics.

Two-stage optimization. The pipeline optimizes $\mathcal{S}^G$ and $\{\mathcal{S}^{T_c}\}$ in separate stages, motivated by both computational efficiency and conceptual separation. From a computational perspective, a single edit to $\mathcal{S}^G$ requires evaluation over the full task suite, whereas edits to $\mathcal{S}^{T_c}$ affect only a single task type. From a modeling perspective, the two skill levels encode fundamentally different forms of knowledge. General skills capture backbone-level behavioral guidance that is intended to transfer across tasks (e.g., "always verify your action parsed correctly"), while task-specific skills encode domain procedures tailored to particular environments (e.g., "check the fridge before the counter"). Therefore, separating the two stages simplifies credit assignment across the two skill levels while substantially reducing search cost.

Stage 1: General skills via iterative hill climbing. General skills encode high-level behavioral priors that affect agent behavior across many task types. Evaluating a candidate general skill requires running the agent across the full task suite and aggregating feedback over diverse environments, making exhaustive search prohibitively expensive. We therefore optimize $\mathcal{S}^G$ via iterative hill climbing (Russell, 2010), which provides a simple and effective strategy for progressively improving the current skill set under environment feedback.

Each iteration proceeds as follows. *Rollout*: the target model F equipped with the current best general skills is rolled out across all task types to compute the reward across episodes. *Analysis*: the teacher collects failed trajectories from these rollouts and produces a structured failure attribution focusing on general behavioral deficiencies rather than task-specific procedural gaps. *Rewrite*: given the current best skill set, the failure attribution, the model card $\mathcal{M}_F$, and the K highest-reward skill sets from all previous iterations (which help the teacher learn from the full optimization trajectory rather than only the most recent failure), the teacher outputs a revised general skill set. *Accept/Reject*: the new candidate is evaluated on the full task suite and accepted only if it achieves a higher reward than the current best. Search terminates after at most I iterations or after p consecutive iterations without improvement.

Stage 2: Task-specific skills via per-type tree search. Unlike general skills, task-specific skills encode domain procedures where multiple structurally different strategies may be effective for the same task type. This motivates a tree-structured search that can explore diverse branches rather than committing to a single refinement path. We run an independent tree search per task type c, where each node holds a candidate task-specific skill set $\mathcal{S}^{T_c}$ and each edge corresponds to a teacher rewrite.

Each iteration proceeds in four steps. *Selection*: starting from the root, UCB1 (Kocsis and Szepesvári, 2006) is applied recursively to select the most promising leaf node, balancing exploitation of high-reward nodes with exploration of less-visited ones. *Expansion*: the target model F is rolled out on type-c tasks using the selected node's skill set, and the teacher collects failed trajectories, produces a failure attribution, and outputs a revised task-specific skill set—forming a new child node. *Evaluation*: the new child's skill set is evaluated on type-c tasks to obtain its average reward. *Back-propagation*: the reward is propagated from the new node back to the root, updating visit counts and value estimates along the path. Per-type trees are independent and fully parallelizable.

Overall, the two stages run sequentially: $\mathcal{S}_F^{G\star}$ obtained in Stage 1 is held fixed throughout Stage 2, and the final output is a *model-specific* skill library $\mathcal{S}_F^\star = (\mathcal{S}_F^{G\star}, \{\mathcal{S}_F^{T_c\star}\}_{c\in\mathcal{C}})$. The detailed procedures are given in Algorithms 1 and 2, and further details of the two-stage search are provided in Appendix E.

3.3 Model-Conditioned Skill Rewriter

The skill evolution pipeline delivers strong skill libraries but requires substantial compute (hundreds to thousands of full-environment rollouts) and an environment-provided reward signal. MASA-Rewriter addresses this by learning the rewriting policy that the evolution pipeline implicitly executes, enabling cheap adaptation to new domains and tasks without further environment interaction.

Training data. Each training instance maps an input skill set to the corresponding evolved optimum:

$$(\mathcal{M}_F,\ \mathcal{S}_{F_{\text{in}}},\ d) \longrightarrow \mathcal{S}_F^\star, \tag{4}$$

where $\mathcal{M}_F$ is the model card, d is the task description, and $\mathcal{S}_{F_{\text{in}}}$ is an input skill set (either general $\mathcal{S}^G$ or task-specific $\mathcal{S}^{T_c}$). The output $\mathcal{S}_F^\star$ is always drawn from the evolution pipeline's high-scoring skill sets. To ensure the rewriter learns to improve skills regardless of their initial quality, $\mathcal{S}_{F_{\text{in}}}$ is deliberately sampled from sources spanning a wide quality range: (i) *search intermediates* at early, mid, and late stages of the evolution pipeline; (ii) *cross-model transfers*—optimal skills from a different backbone; (iii) *one-shot teacher rewrites* without iterative search; and (iv) *augmented variants* (noisy, partial, or verbose perturbations of existing skills). This diversity exposes the rewriter to the full range of input conditions it may encounter at deployment. Additional details are provided in Appendix F.

Training. We instantiate the skill rewriter with Qwen3-4B, a lightweight backbone chosen to keep deployment cost minimal while retaining sufficient capacity for structured rewriting. The model is trained via supervised fine-tuning (SFT) with cross-entropy loss:

$$\mathcal{L} = -\mathbb{E}_{\mathcal{D}_{\text{train}}}[\log\, p_\theta(\mathcal{S}_F^\star \mid \mathcal{M}_F,\ \mathcal{S}_{F_{\text{in}}},\ d)]\,. \tag{5}$$

Inference. At deployment, given the target backbone's model card $\mathcal{M}_F$, an input skill set $\mathcal{S}_{F_{\text{in}}}$, and the task description d, the skill rewriter produces an adapted skill set in a single forward pass:

$$\mathcal{S}_F^\star = f_\theta(\mathcal{M}_F,\ \mathcal{S}_{F_{\text{in}}},\ d)\,, \tag{6}$$

without requiring any environment interaction or iterative search.

Complementary roles. The skill evolution pipeline provides per-backbone upper bounds via explicit search and produces the skill rewriter's

Model	Method	ALFWorld								WebShop		
		Pick	Look	Clean	Heat	Cool	Pick2	SR ↑	Steps ↓	SR ↑	Score ↑	Steps ↓
Qwen3-4B	No Skill	20.0	15.4	18.5	18.8	16.0	12.5	17.1	44.6	23.0	42.2	9.5
	+ Base Skill	20.0	30.8	29.6	12.5	20.0	8.3	20.0	42.3	19.4	34.8	11.0
	+ DS-Adapter	**28.6**	30.8	37.0	18.8	**32.0**	12.5	27.1	40.0	19.2	25.7	12.4
	+ MASA	25.7	**53.8**	**40.7**	**37.5**	24.0	**20.8**	**31.4**	**38.4**	**26.4**	**49.1**	**8.4**
Qwen3-8B	No Skill	54.3	**46.2**	29.6	6.2	24.0	20.8	32.1	39.1	4.6	32.7	10.0
	+ Base Skill	17.1	38.5	40.7	31.2	20.0	12.5	25.0	40.5	6.0	32.6	9.3
	+ DS-Adapter	25.7	38.5	44.4	25.0	16.0	16.7	27.1	39.6	4.4	18.2	12.6
	+ MASA	**62.9**	38.5	**70.4**	**75.0**	**56.0**	**37.5**	**57.9**	**29.2**	**28.6**	**60.1**	**4.7**
Qwen3-14B	No Skill	65.7	38.5	25.9	43.8	16.0	29.2	37.9	36.7	2.8	19.9	12.7
	+ Base Skill	68.6	46.2	44.4	25.0	20.0	33.3	42.1	34.1	1.6	14.8	13.5
	+ DS-Adapter	68.6	**53.8**	40.7	18.8	40.0	29.2	44.3	34.8	2.0	12.6	13.6
	+ MASA	**85.7**	**53.8**	**81.5**	**56.2**	**44.0**	**45.8**	**64.3**	**25.7**	**29.2**	**54.4**	**8.0**
Qwen3-32B	No Skill	48.6	**46.2**	44.4	25.0	32.0	16.7	36.4	37.0	6.6	35.2	9.9
	+ Base Skill	48.6	**46.2**	40.7	50.0	44.0	20.8	41.4	35.6	7.2	24.2	12.0
	+ DS-Adapter	51.4	38.5	59.3	37.5	44.0	29.2	45.0	32.2	3.6	14.3	13.3
	+ MASA	**57.1**	**46.2**	**77.8**	**81.3**	**76.0**	**54.2**	**65.7**	**24.3**	**34.6**	**59.9**	**7.3**

Table 1: Performance on ALFWorld and WebShop. ALFWorld reports per-task and average success rate (SR %), and average interaction steps across all task types; WebShop reports average SR (%), score, and average steps. **Bold** marks the best within each backbone.

training signal for the skill rewriter. MASA-Rewriter amortizes this search into a single forward pass, enabling rapid adaptation without environment interaction. The evolution pipeline is preferred when rollout budget permits thorough optimization, whereas the skill rewriter is better suited to compute-constrained deployment scenarios.

4 Experiments

We evaluate whether model-conditioned skill evolution outperforms model-agnostic baselines across diverse environments and backbones, and whether MASA-Rewriter can generalize the learned rewriting policy to held-out task types and unseen environments without additional search.

4.1 Experimental Setup

Environments. We evaluate on three environments spanning distinct action spaces and reasoning demands. (i) ALFWorld (Shridhar et al., 2020) is a text-based embodied environment where agents complete household tasks (e.g., heating, cleaning, picking up objects) by issuing text commands to navigate rooms and interact with objects. It contains six task types with varying difficulty. (ii) WebShop (Yao et al., 2022a) simulates online shopping: agents navigate a realistic web interface, search for products, compare attributes, and make purchase decisions that satisfy natural-language user specifications. (iii) Search-augmented QA requires agents to retrieve and synthesize information from web search results. We include seven benchmarks covering both single-hop (NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023)) and multi-hop reasoning (HotpotQA (Yang et al., 2018), 2Wiki (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2023)).

Backbones and baselines. Target agents are Qwen3-{4B, 8B, 14B, 32B} (Yang et al., 2025); [2] the teacher LLM is DeepSeek-V4-Pro (DeepSeek-AI, 2026). We compare against three baselines: (1) *No Skill* (the raw backbone without any skill augmentation), (2) *Base Skill* (the initial skill library from SkillRL (Xia et al., 2026), shared across all backbones without model-specific adaptation), and (3) *DS-Adapter* (a one-shot teacher rewrite that adapts the Base Skill library conditioned on the model card $\mathcal{M}_F$, without iterative search).
The Base Skill library also serves as the initialization $\mathcal{S}_F^{G_0}$ and $\mathcal{S}_F^{T_{c_0}}$ for MASA's evolution pipeline.

4.2 Skill Evolution Evaluation

Table 1 reports ALFWorld and WebShop results across all four backbones.

[2] All Qwen3 backbones are used in non-thinking mode. This choice reflects typical deployment scenarios where latency and token budgets are constrained, and ensures that observed performance differences are attributable to skill design rather than reasoning-mode configuration.

Model	Method	Single-Hop QA			Multi-Hop QA				Avg.
		NQ†	TriviaQA⋆	PopQA⋆	HotpotQA†	2Wiki⋆	MuSiQue⋆	Bamboogle⋆	
Qwen3-4B	No Skill	29.4	51.0	37.2	27.7	22.8	6.4	9.3	32.9
	+ Base Skill	34.5	**57.4**	38.2	28.5	24.4	7.8	10.1	35.5
	+ DS-Adapter	33.0	56.5	**41.8**	**28.6**	23.9	9.3	12.9	36.2
	+ MASA	**35.5**	55.3	38.9	27.4	**27.0**	**9.4**	**61.3**	**36.9**
Qwen3-8B	No Skill	19.1	46.5	30.3	24.8	**30.6**	6.7	**68.1**	31.3
	+ Base Skill	34.0	**58.5**	38.8	**28.6**	25.9	6.2	10.1	36.1
	+ DS-Adapter	33.2	57.6	38.7	27.8	22.9	5.6	7.7	35.0
	+ MASA	**36.4**	56.7	**39.0**	**28.6**	25.7	**10.0**	62.5	**37.2**
Qwen3-14B	No Skill	33.8	60.2	40.6	31.7	26.8	7.6	10.5	37.6
	+ Base Skill	35.3	60.5	39.5	32.7	**30.3**	**11.4**	**15.3**	38.5
	+ DS-Adapter	33.9	60.2	39.5	31.6	28.5	9.2	12.5	37.7
	+ MASA	**35.6**	**61.8**	**40.7**	**32.8**	30.0	9.7	8.9	**39.0**
Qwen3-32B	No Skill	29.1	59.8	38.3	32.2	29.3	8.6	64.5	38.1
	+ Base Skill	33.8	61.4	39.3	33.8	26.0	11.7	**67.7**	38.7
	+ DS-Adapter	34.4	61.5	**40.6**	34.0	32.0	11.6	64.1	40.6
	+ MASA	**37.0**	**61.6**	40.0	**34.2**	**35.6**	**11.8**	66.1	**41.5**

Table 2: Search-augmented QA results (success rate %). Skill evolution is conducted on NQ and HotpotQA; † and ⋆ indicate in-domain and out-of-domain datasets, respectively. **Bold** marks the best within each backbone.

ALFWorld. MASA achieves the highest average success rate for every backbone: 31.4 (4B), 57.9 (8B), 64.3 (14B), and 65.7 (32B), with gains of +4.3, +25.8, +20.0, and +20.7 over the strongest baseline respectively. We highlight several observations:

(1) Per-task dominance. Beyond the aggregate, MASA achieves the best per-task SR in most individual task types. For Qwen3-14B and 32B, MASA ranks first on *all six* task types simultaneously, indicating that the evolved skills improve overall performance without sacrificing coverage across tasks.

(2) Model-agnostic skills can hurt. Base Skill and DS-Adapter exhibit severe performance drops on individual tasks, indicating that generic or one-shot adapted skills can introduce model-specific conflicts. In contrast, MASA avoids these regressions through iterative model-conditioned search.

(3) Scaling behavior. For 8B and above, the backbones have sufficient capacity to leverage model-specific skills, yielding substantial improvements. The gain on 4B is comparatively modest, likely due to the backbone's inherent capability ceiling limiting how much skill guidance can help.

(4) Inference efficiency. MASA consistently reduces average interaction steps (e.g., 8B: 39.1 → 29.2; 14B: 36.7 → 25.7). By tailoring skills to each backbone's specific behavior patterns, MASA helps the agent locate target objects and execute correct action sequences more precisely, reducing redundant exploration and failed attempts.

WebShop. MASA again achieves the highest success rate and score for every backbone, substantially outperforming all baselines. WebShop reveals a critical challenge for larger Qwen3 models:

(1) Larger models perform worse than 4B without adaptation. Notably, 8B/14B/32B baselines all underperform 4B on WebShop (e.g., 14B No Skill: 2.8% vs. 4B No Skill: 23.0%). We trace this to excessive chain-of-thought generation: larger models produce verbose reasoning preambles before each action, inflating action length and exhausting the step budget on deliberation rather than environment interaction (detailed statistics in Appendix G). Since model-agnostic skills are not designed to address this model-specific behavioral pattern, they provide limited benefit and in some cases further degrade performance.

(2) MASA addresses this challenge. By evolving skills conditioned on each backbone's behavioral profile, MASA guides models toward effective action patterns—achieving SR of 26.4 (4B), 28.6 (8B), 29.2 (14B), and 34.6 (32B), far surpassing all baselines. The efficiency gain is also notable: baselines that do succeed average 12–13 steps, whereas MASA achieves higher SR in only 7–8 steps, dropping to just 4.7 steps on 8B.

Search-augmented QA. Table 2 shows that MASA achieves the highest average SR for every backbone. Skill evolution is conducted only on NQ and HotpotQA, yet the gains generalize strongly to out-of-domain benchmarks (⋆)—e.g.,

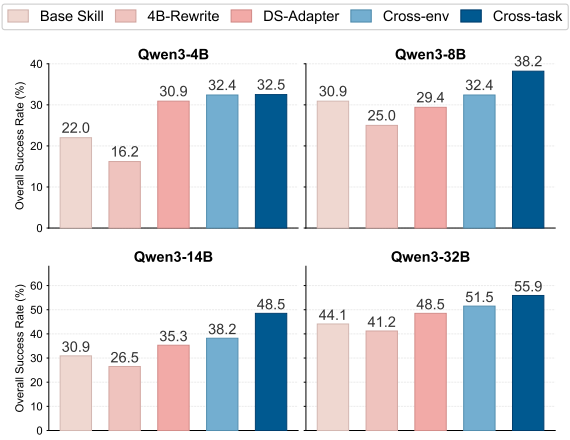

Figure 3: OOD generalization of MASA-Rewriter on held-out ALFWorld task types. Pink bars denote baselines and blue bars denote MASA-Rewriter variants.

on 4B, MASA improves Bamboogle from 12.9 (best baseline) to 61.3. On the largest backbone (32B), MASA ranks first on 5 out of 7 datasets. These results demonstrate that the evolved skills capture transferable strategies for retrieval and reasoning, rather than overfitting to the datasets used during skill evolution.

4.3 Skill Rewriter Generalization

We evaluate whether MASA-Rewriter can adapt skills for task types not seen during its training, by holding out three ALFWorld task types (CLEAN, HEAT, COOL) and asking MASA-Rewriter to produce task-specific skills for these types. The general skills remain unchanged.

We compare against three baselines: *Base Skill*, *4B-Rewrite* (Qwen3-4B used as a rewriter without SFT—i.e., the same architecture as MASA-Rewriter but without learned rewriting ability), and *DS-Adapter* (one-shot teacher rewrite targeting the specific held-out task). The two MASA-Rewriter variants differ in training data composition:

Cross-environment transfer. MASA-Rewriter is trained exclusively on skill evolution traces from Search and WebShop, then applied to ALFWorld without any in-environment data. Despite the substantial environment gap (different action spaces and observation formats), Cross-env MASA-Rewriter outperforms DS-Adapter on all four backbones (Figure 3), with gains of +1.5 (4B), +3.0 (8B), +2.9 (14B), and +3.0 (32B). This demonstrates that the learned rewriting policy captures model-specific adaptation patterns that transfer across environments—the rewriter can produce useful skills even without exposure to the target environment during training.

Cross-task transfer. Building on Cross-env, this variant additionally uses evolution traces from three other ALFWorld task types (PICK, LOOK, PICK2), excluding the held-out evaluation types. Cross-task MASA-Rewriter achieves substantially larger gains: +8.8 (8B), +13.2 (14B), and +7.4 (32B) over DS-Adapter. The gains over Cross-env are especially large on 8B (+5.8) and 14B (+10.3), suggesting that even skills from unrelated task types provide valuable supervision for adapting to environment-specific interaction patterns and observation structures.

Notably, MASA-Rewriter (4B parameters) consistently surpasses DS-Adapter powered by DeepSeek-V4, demonstrating that a small trained rewriter can outperform a much larger teacher at a fraction of the inference cost.

Due to space constraints, additional materials including extended related work discussion, ablations, supplementary validation on Gemma3, qualitative examples, and full hyperparameter details are reported in Appendices A–I.

5 Conclusion

We presented MASA, motivated by the observation that the one-size-fits-all assumption in agent skill design breaks down across model scales. MASA addressed this through hierarchical skill evolution and a lightweight model-conditioned rewriter that amortizes search into a single forward pass. Across three environments and four Qwen3 backbones, MASA achieved the best success rate in all settings, with gains up to +25.8 points. The rewriter further generalized to unseen tasks and environments at negligible deployment cost.

We hope this work motivates treating skills as model-aware artifacts that should be adapted to their target backbone rather than shared uniformly across models of different capacities. With proper alignment, even compact models can exhibit behaviors traditionally associated with frontier-scale systems, enabling more accessible and resource-efficient deployment. Looking forward, we envision MASA-Rewriter as a lightweight plug-and-play middleware that automatically rewrites existing skill libraries for new backbones, requiring no environment rollouts, retraining, or manual prompt engineering. This positions skill alignment as infrastructure rather than a per-deployment engineering effort.

Limitations

Our empirical evidence is currently restricted to the Qwen3 family (4B/8B/14B/32B); extending the skill evolution and rewriter to cover more model families—both open-weight (e.g., Llama (Grattafiori et al., 2024), Mistral (Liu et al., 2026)) and proprietary (e.g., GPT-o3 (OpenAI, 2025), Claude (Anthropic, 2024))—and more diverse environments would further strengthen the generality of the skill rewriter, though it requires substantially more compute. In particular, applying the evolution pipeline to closed-source models demands hundreds of environment rollouts through paid APIs, making the per-backbone search cost significantly higher than for locally hosted models; the the skill rewriter rewriter offers a partial remedy by amortizing this cost once trajectories from a few backbones are available.

Additionally, the skill rewriter is trained on skill-evolution trajectories collected from ALFWorld, WebShop, and Search-QA, and the evolution pipeline itself relies on environments that provide automatic success/failure signals (e.g., task completion flags) to judge whether a rewritten skill is effective. Incorporating domains without such built-in reward signals (e.g., open-ended web tasks or real-world applications (Sun et al., 2025)) would require designing external evaluators or human annotations, but would enable the framework to serve an even broader range of agent applications.

Ethical Considerations

Data and Licensing. MASA does not introduce new data collection from human subjects; all experiments use standard public benchmarks (ALFWorld, WebShop, and open-domain QA datasets) and publicly released models accessed in accordance with their respective licenses.

Safety of Agent Empowerment. By improving the effectiveness of LLM agents through skill adaptation, MASA may also increase the capability of agents operating in interactive environments. Overall, the framework should be deployed in safety-critical or high-risk settings with additional monitoring, policy constraints, and human oversight.

Bias and Reliability of Evolved Skills. The skill evolution pipeline may inherit biases or unsafe heuristics from the trajectories and feedback used during optimization. Evolved skill libraries should therefore be inspected and validated before deployment.

References

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. Technical report, Anthropic.

Le Chen, Erhu Feng, Yubin Xia, and Haibo Chen. 2026. Skvm: Revisiting language vm for skills across heterogenous llms and harnesses. *arXiv preprint arXiv:2604.03088*.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. 2024. Automanual: Constructing instruction manuals by llm agents via interactive environmental learning. *Advances in Neural Information Processing Systems*, 37:589–631.

Yuyan Chen, Zhihao Wen, Ge Fan, Zhengyu Chen, Wei Wu, Dayiheng Liu, Zhixu Li, Bang Liu, and Yanghua Xiao. 2023. Mapo: Boosting large language model performance with model-adaptive prompt optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3279–3304.

DeepSeek-AI. 2026. Deepseek-v4 technical report. Technical report, DeepSeek-AI.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2024. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *International Conference on Learning Representations*, volume 2024, pages 34133–34156.

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

Vincent Hsiao, Mark Roberts, and Leslie Smith. 2025. Procedural knowledge improves agentic llm workflows. *arXiv preprint arXiv:2511.07568*.

Yanna Jiang, Delong Li, Haiyu Deng, Baihe Ma, Xu Wang, Qin Wang, and Guangsheng Yu. 2026. Sok: Agentic skills–beyond tool use in llm agents. *arXiv preprint arXiv:2602.20867*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. 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)*, pages 1601–1611.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 4.

Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *European conference on machine learning*, pages 282–293. Springer.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, and 1 others. 2026. Ministral 3. *arXiv preprint arXiv:2601.08584*.

Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Chengcheng Han, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. 2026. Skill0: In-context agentic reinforcement learning for skill internalization. *arXiv preprint arXiv:2604.02268*.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. 2026. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. 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)*, pages 9802–9822.

OpenAI. 2025. Introducing openai o3 and o4-mini. Technical report, OpenAI.

Siru Ouyang, Jun Yan, Yanfei Chen, Rujun Han, Zifeng Wang, Bhavana Dalvi Mishra, Rui Meng, Chun-Liang Li, Yizhu Jiao, Kaiwen Zha, and 1 others. 2026. Skillos: Learning skill curation for self-evolving agents. *arXiv preprint arXiv:2605.06614*.

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 the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.

Stuart J Russell. 2010. *Artificial intelligence a modern approach*. Pearson Education, Inc.

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

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, volume 2024, pages 25055–25083.

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

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

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, and 1 others. 2023. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, and 1 others. 2025. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5555–5579.

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

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Jiongxiao Wang, Qiaojing Yan, Yawei Wang, Yijun Tian, Soumya Smruti Mishra, Zhichao Xu, Megha Gandhi, Panpan Xu, and Lin Lee Cheong. 2025a. Reinforcement learning for self-improving agent with skill library. *arXiv preprint arXiv:2512.17102*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Yaxuan Wang, Quan Liu, Zhenting Wang, Zichao Li, Wei Wei, Yang Liu, and Yujia Bao. 2025b. Promptbridge: Cross-model prompt transfer for large language models. *arXiv preprint arXiv:2512.01420*.

Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, and 1 others. 2024b. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models.

IEEE Transactions on Pattern Analysis and Machine Intelligence, 47(3):1894–1907.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024c. Agent workflow memory. *arXiv preprint arXiv:2409.07429*.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, and 1 others. 2026. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. 2025. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. In *International Conference on Learning Representations*, volume 2025, pages 79822–79843.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pages 12028–12068.

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

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

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

Zhiwei Yao, Yang Xu, Hongli Xu, Yunming Liao, and Zuan Xie. 2025. Efficient deployment of large language models on resource-constrained devices. *arXiv preprint arXiv:2501.02438*.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642.

Yue Zheng, Yuhao Chen, Bin Qian, Xiufang Shi, Yuanchao Shu, and Jiming Chen. 2025. A review on edge large language models: Design, execution, and applications. *ACM Computing Surveys*, 57(8):1–35.

Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, and 1 others. 2023. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory. *arXiv preprint arXiv:2305.17144*.

A Related Work

LLM agents and skill libraries. Equipping LLM agents with reusable procedural knowledge is a scalable approach to improve agent performance without modifying model weights. Early efforts such as ReAct (Yao et al., 2022b) and Reflexion (Shinn et al., 2023) leverage textual feedback as in-context skill; Voyager (Wang et al., 2023) maintains a growing skill library for Minecraft; JARVIS-1 (Wang et al., 2024b) and Ghost-in-the-Minecraft (Zhu et al., 2023) cache successful behaviors for replay at inference; AgentTrek (Xu et al., 2025) bootstraps web agents with synthesized trajectories; AutoManual (Chen et al., 2024) induces an *operating manual* from interaction traces; and ExpeL (Zhao et al., 2024) distills cross-trial experiences into a reusable insights library. More recent systems further elevate skills into first-class agent components: SkillRL (Xia et al., 2026) distills trajectories into a hierarchical SkillBank and recursively evolves skills with the agent policy; and SkillOS (Ouyang et al., 2026) learns a long-horizon curator that inserts, updates, and deletes skills in an external SkillRepo. Notably, SkVM (Chen et al., 2026) also identifies the model-skill mismatch problem—reporting that 87% of tasks have at least one LLM that gains no benefit from the same skill—and addresses it by compiling skills into optimized runtime formats (e.g., code solidification, parallelization) to reduce latency and token cost. MASA shares the same motivation but pursues a complementary direction: rather than compiling skills for execution efficiency, we *rewrite the natural-language expression* of skills to match each backbone's comprehension and reasoning style, directly improving task success rate.

Model-aware adaptation and prompt optimization. LLM behavior is highly sensitive to instruction phrasing even under semantically equivalent prompts (Sclar et al., 2024), motivating methods that tailor prompts to specific backbones. Teacher-driven search methods such as OPRO (Yang et al., 2024) and EvoPrompt (Guo et al., 2024) iteratively refine a single instruction for a given task, yet treat

the target model as fixed context—the same output applies regardless of backbone. MAPO (Chen et al., 2023) and PromptBridge (Wang et al., 2025b) further account for model identity by optimizing or transferring individual task instructions across backbones, yet they operate on single monolithic prompts in non-agent settings rather than on retrievable multi-entry skill libraries used at agent decision time. MASA differs in two key respects: (i) the optimization target is a dynamically retrieved *skill library* rather than a monolithic prompt, and (ii) the evolutionary search is jointly steered by a structured model capability profile and environment reward signals, explicitly conditioning skill expression on target-model characteristics. Furthermore, we train a lightweight skill rewriter that amortizes the expensive search process into a single forward pass—conceptually related to distilling costly inference-time computation into efficient learned models (Singh et al., 2023)—enabling skill adaptation to new backbones without repeated search.

B Ablations

Two-stage evolution pipeline (Table 3a). We ablate the two-stage search structure by replacing each stage's evolved skills with one-shot teacher (DeepSeek-V4) rewrites, isolating the contribution of each search stage. *w/o Task-specific* retains MASA-evolved general skills but substitutes teacher-written task-specific skills (i.e., Stage 1 only), while *w/o General* retains MASA-evolved task-specific skills but uses teacher-written general skills (i.e., Stage 2 only). Both stages contribute to the full pipeline, but their relative importance is environment- and scale-dependent. On ALFWorld, removing task-specific search causes the largest drops for Qwen3-8B (-25.0) and Qwen3-32B (-15.7), indicating that per-task-type procedural guidance is critical for these backbones. Conversely, removing general search most severely affects Qwen3-14B (-16.4), suggesting that high-level behavioral priors are essential when the model has sufficient capacity to follow them but still benefits from strategic framing. On WebShop, removing general skills is catastrophic for 8B/14B/32B (SR drops to single digits), while removing task-specific skills has a comparatively modest effect. This asymmetry reflects the nature of each environment—WebShop demands consistent high-level decision strategies that general skills encode, whereas ALFWorld requires fine-grained procedural sequences that task-specific skills address.

Rewriter model card (Table 3b). We ablate the model card input to the MASA-Rewriter by comparing performance with and without the target model's capability card, using both training data variants: *Cross-env* (trained on Search + WebShop) and *Cross-task* (trained on Search + WebShop + ALFWorld Pick/Look/Pick2). All results are average SR on three held-out ALFWorld tasks (Clean/Heat/Cool). Removing the model card consistently degrades performance. For Cross-task, the gap is especially pronounced on Qwen3-14B, confirming that the card provides critical conditioning signal for smaller backbones. For Cross-env, model card removal also causes substantial drops. Overall, these results suggest that the model card provides useful backbone-specific conditioning signals that help the rewriter generate more appropriate skill adaptations.

Variant	4B	8B	14B	32B
ALFWorld				
Full pipeline	**31.4**	**57.9**	**64.3**	**65.7**
w/o Task-specific	25.0	32.9	63.6	50.0
w/o General	25.0	50.0	47.9	64.3
WebShop				
Full pipeline	**26.4**	**28.6**	**29.2**	**34.6**
w/o Task-specific	22.4	25.6	24.2	31.8
w/o General	23.4	7.2	10.2	9.6

(a) The search-based evolution pipeline.

Variant	4B	8B	14B	32B
Cross-env				
w/ Model Card	**32.4**	**32.4**	**38.2**	**51.5**
w/o Model Card	14.7	23.5	33.8	39.7
Cross-task				
w/ Model Card	**32.5**	**38.2**	**48.5**	**55.9**
w/o Model Card	8.8	30.9	20.6	42.5

(b) Model card conditioning in the MASA-Rewriter.

Table 3: Ablation studies of MASA.

C Preliminary Study: Supplementary Details

C.1 Skill Variant Comparison

Table 4 shows concrete examples of the three non-empty ALFWorld skill variants used in the preliminary study. All variants keep the same skill IDs and task coverage; what changes is how much procedural text is exposed to the agent. We use bold text to highlight trigger conditions, executable steps, and failure-prevention cues added by the more detailed

variants.

C.2 Per-Task Breakdown: Qwen3

Table 5 expands the overall numbers visualized in Figure 1 into per-task success rates for each (model, skill) cell. The breakdown is computed on the same ALFWorld validation set. Note the large within-condition swings across task types (e.g., Qwen3-14B Concise: 74.2 on PICK vs. 13.7 on COOL; Qwen3-4B Detailed: 1.6 on PICK vs. 46.7 on LOOK), which are substantially larger than cross-condition differences and motivate the task-specific tree-search stage of the evolution pipeline.

C.3 Supplementary Validation: Gemma3

To verify that the scale-dependent granularity pattern is not unique to Qwen3, we repeat the fixed-granularity sweep on Gemma3 backbones (4B/12B/27B) (Kamath et al., 2025). It supports the motivating conclusion: the best skill form is model-dependent rather than universally transferable. Gemma3-4B and Gemma3-12B are strongest with Concise skills, while Gemma3-27B reaches its best success rate with Detailed skills. Figure 4 shows the overall results and Table 6 gives the per-task breakdown.

Comparing models of the same parameter count across families further isolates the effect of architecture and training from that of scale alone. Gemma3-4B achieves its best performance with `Concise` skills, whereas Qwen3-4B peaks under `Moderate` skills (Figure 1 vs. Figure 4). Despite identical parameter budgets, the two models respond to skill granularity in qualitatively different ways—confirming that the optimal skill form is determined by a model's overall characteristics (architecture, training data, alignment procedure) rather than parameter count alone. This observation reinforces the necessity of conditioning skill adaptation on a rich model profile rather than relying on scale as a proxy.

D Model Card Construction

Each model card is constructed from a fixed rubric combining public documentation and automated analysis:

1. *Architecture metadata.* Model family, variant name, parameter count, architecture type, layer/attention configuration, context window, and vocabulary size—sourced directly from the published model card or config files.

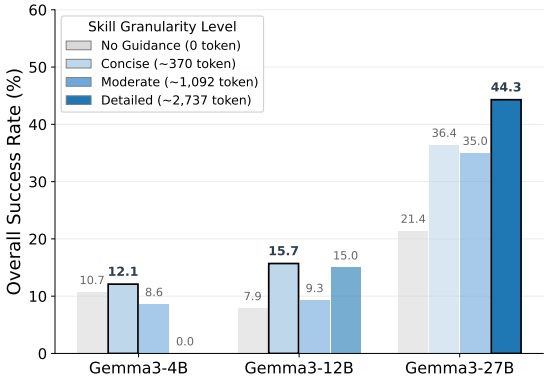

Figure 4: The supplementary validation of Gemma family.

2. *Training provenance.* Whether the checkpoint is base or instruction-tuned, the alignment pipeline (e.g., SFT + DPO + GRPO), training data scale, and multilingual support—sourced from official documentation.
3. *Capability profile.* Strengths are extracted from the model's official release notes (e.g., "strong at math and code generation"). Weaknesses are generated by the teacher LLM summarizing behavioral patterns observed during a small set of preliminary rollouts (Section 2), produced automatically without human annotation.

Note that the card does not include any downstream evaluation results (e.g., ALFWorld success rates) or oracle style labels (e.g., *prefers_concise*). The preliminary rollouts used for weakness summarization are disjoint from the evaluation set.

Below is the card for Qwen3-4B; cards for the remaining backbones follow the same template.

```
# Model Card: Qwen3-4B
# Source: https://huggingface.co/Qwen/
    Qwen3-4B

# === Architecture Metadata ===
family: "Qwen3"
variant: "4B"
parameter_count: "4B"
architecture: "dense-transformer"
num_layers: 36
hidden_size: 2560
num_attention_heads: 32
num_kv_heads: 8
context_window: 32768
vocab_size: 151936

# === Training Provenance ===
base_or_instruct: "instruct"
alignment_method: "SFT + DPO + GRPO"
training_data_size: "36T tokens"
multilingual: true

# === Official Capabilities (from
    release notes) ===
strengths: "math, code generation,
```

Algorithm 1 Stage 1: General Skill Search (Hill Climbing)

Require: Target model F, model card $\mathcal{M}_F$, teacher T, eval set $\mathcal{D}$ (sampled from training episodes), initial general skills $\mathcal{S}_F^{G_0}$, max iterations I, patience p, history size K
Ensure: Optimized general skills $\mathcal{S}_F^{G\star}$

$\mathcal{S}_F^{G\star} \leftarrow \mathcal{S}_F^{G_0}$ {current best general skill set}
$R^\star \leftarrow \mathrm{Eval}(F, \mathcal{S}_F^{G\star}, \mathcal{D})$ {its average reward across all task types}
$\mathcal{H} \leftarrow \{(\mathcal{S}_F^{G_0}, R^\star)\}$ {search history: (skill set, reward) pairs}
for $i = 1$ **to** I **do**
  // Rollout & Analysis
  $\mathcal{F}_i \leftarrow \mathrm{CollectFailures}(F, \mathcal{S}_F^{G\star}, \mathcal{D})$
  $\mathrm{attr}_i \leftarrow T.\mathrm{Analyze}(\mathcal{F}_i)$ {structured failure attribution}
  // Rewrite
  $\mathcal{S}_F^{G_i} \leftarrow T.\mathrm{Rewrite}(\mathcal{S}_F^{G\star}, \mathrm{attr}_i, \mathrm{TopK}(\mathcal{H}, K), \mathcal{M}_F)$
  // Accept / Reject
  $R_i \leftarrow \mathrm{Eval}(F, \mathcal{S}_F^{G_i}, \mathcal{D})$
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{(\mathcal{S}_F^{G_i}, R_i)\}$
  if $R_i > R^\star$ **then**
    $\mathcal{S}_F^{G\star} \leftarrow \mathcal{S}_F^{G_i}$; $R^\star \leftarrow R_i$ {accept}
  end if
  if no improvement for p consecutive iterations **then**
    break
  end if
end for
return $\mathcal{S}_F^{G\star}$

```
    instruction
  following, multilingual, tool use,
      thinking
  mode support"

# === Observed Weaknesses (teacher-
    summarized) ===
weaknesses: "limited reasoning depth due
     to small
  parameter count, may struggle with
      complex
  multi-step planning"
```

E Skill Evolution Pipeline Details

This appendix provides the full algorithmic procedures (Algorithms 1 and 2) and hyperparameters for the two-stage skill evolution pipeline described in Section 3.2. All evolution experiments use the original training split of each environment for exploration; the evaluation results reported in the main paper are obtained on the held-out test split.

E.1 Stage 1: Hill Climbing

Maximum iterations I=10; patience p=3 (early stopping after 3 consecutive iterations without improvement); top-K=5 highest-reward historical skill sets provided to the teacher at each iteration. A candidate general skill set is accepted if and only if its average adjusted reward strictly exceeds the current best.

Algorithm 2 Stage 2: Task-Specific Skill Search (Per-Type Tree Search)

Require: Target model F, model card $\mathcal{M}_F$, teacher T, fixed general skills $\mathcal{S}_F^{G\star}$, initial task-specific skills $\{\mathcal{S}_F^{T_{c_0}}\}_{c\in\mathcal{C}}$, iterations J
Ensure: Optimized task-specific skills $\{\mathcal{S}_F^{T_c\star}\}_{c\in\mathcal{C}}$

for each task type $c \in \mathcal{C}$ **in parallel do**
  Initialize tree root with $\mathcal{S}_F^{T_{c_0}}$
  for $j = 1$ **to** J **do**
    // Selection
    $n \leftarrow \mathrm{UCB1Select}(\mathrm{root})$ {select leaf via Eq. 7}
    // Expansion
    $\mathcal{F} \leftarrow \mathrm{CollectFailures}(F, \mathcal{S}_F^{G\star}, \mathcal{S}_{F,n}^{T_c}, c)$
    $\mathrm{attr} \leftarrow T.\mathrm{Analyze}(\mathcal{F})$ {failure attribution}
    $\mathcal{S}_F'^{T_c} \leftarrow T.\mathrm{Rewrite}(\mathcal{S}_{F,n}^{T_c}, \mathrm{attr}, \mathcal{M}_F)$
    Add $\mathcal{S}_F'^{T_c}$ as child of node n
    // Evaluation
    $R' \leftarrow \mathrm{Eval}(F, \mathcal{S}_F^{G\star}, \mathcal{S}_F'^{T_c}, c)$
    // Backpropagation
    Update visit counts and value estimates from new node to root
  end for
  $\mathcal{S}_F^{T_c\star} \leftarrow$ skill set of the highest-value node
end for
return $\{\mathcal{S}_F^{T_c\star}\}_{c\in\mathcal{C}}$

E.2 Stage 2: UCB-Driven Tree Search

At each iteration, the node n maximizing the following UCB1 score is selected:

$$\mathrm{UCB1}(n) = \bar{R}(n) + C\sqrt{\frac{\ln N_{\text{parent}}}{N_n}}, \quad (7)$$

where $\bar{R}(n)$ is the mean adjusted reward of node n and all its descendants, N_n is the visit count of node n, N_{parent} is the visit count of its parent, and C=1.4 is the exploration constant. We run J=10 iterations per task type with N=100 episodes per node evaluation.

F Skill Rewriter Training Details

We perform full-parameter SFT on Qwen3-4B in BF16 precision. Training uses AdamW (lr 1e−5, cosine schedule, warmup ratio 0.1, gradient checkpointing), effective batch size 4 (per-device 1× gradient accumulation 4), 5 epochs, and max sequence length 4096. We select the best checkpoint based on training loss convergence.

The training data consists of pairs for in-domain tasks, with data augmentation including noisy inputs (noise ratio 0.3), partial inputs (keep ratio 0.6), and cross-model transfer pairs. We train two rewriter variants: a combined rewriter on 769 samples from all three environments (ALFWorld

Pick/Look/Pick2 only—excluding the held-out types, WebShop, and Search), and an environment-specific rewriter on 499 samples (WebShop + Search only).

G WebShop Supplementary Results

G.1 Trajectory Analysis: Why Larger Models Fail

We analyze failed WebShop trajectories to understand why larger Qwen3 models (8B/14B/32B) perform worse than 4B under baseline conditions (Section 4.2).

Table 7 reveals a striking pattern: Qwen3-4B produces concise, action-only outputs (0% steps with chain-of-thought, ∼73 chars per action), while 8B/14B/32B prepend extensive reasoning preambles before each action command. Qwen3-14B is the most severe case, with 97% of steps containing verbose reasoning. This behavior exhausts the fixed step budget on deliberation rather than environment interaction—the agent "thinks" through multiple options but never completes enough purchase actions to succeed.

Model	CoT (%)	Action Len.
Qwen3-4B	0	73 chars
Qwen3-8B	57	1,021 chars
Qwen3-14B	97	574 chars
Qwen3-32B	66	491 chars

Table 7: WebShop trajectory statistics. CoT: fraction of steps containing reasoning preambles.

G.2 Per-Category Breakdown

Table 8 provides the full per-category success rate breakdown. Several observations stand out:

- For 8B/14B/32B baselines, most categories have near-zero SR, consistent with the verbose-reasoning bottleneck identified above.
- MASA achieves the best SR in the vast majority of categories across all backbones, with particularly large gains on OTHER and ELECTRONICS.
- The improvement is broad rather than category-specific: MASA does not exploit a single easy category to inflate the average but improves performance across the board.

H Skill Rewriter OOD: Per-Task Breakdown

Figure 5 shows the per-task SR breakdown for the OOD generalization experiment (Section 4.3).

Cross-task transfer. Adding ALFWorld PICK/LOOK/PICK2 traces to the training set (dark blue) yields consistent improvements over Cross-env on all three tasks. The gains are most pronounced on COOL (e.g., 14B: 24.0→44.0; 32B: 48.0→52.0) and CLEAN (e.g., 8B: 29.6→51.9; 14B: 40.7→51.9), indicating that in-environment traces help the rewriter learn ALFWorld-specific action patterns such as navigation sequences and object interaction protocols. On HEAT, Cross-task improves for 4B and 14B but is comparable to Cross-env for 8B and 32B, suggesting that Heat-specific patterns are already partially captured by the cross-environment signal.

Cross-environment transfer. Trained only on Search and WebShop traces, the Cross-env rewriter (light blue) shows notable strengths on HEAT across all backbones—particularly 14B (56.2%) and 32B (56.2%)—substantially exceeding DS-Adapter. On CLEAN and COOL, Cross-env performance is more mixed: it matches or slightly exceeds DS-Adapter for 4B and 8B, but falls short on some larger-backbone cells (e.g., 14B CLEAN: 40.7 vs. DS-Adapter 40.7, tied). This suggests that cross-environment transfer is most effective when the target task involves decision patterns (e.g., sequential verification in HEAT) that overlap with those in the training environments.

I Qualitative Analysis: Evolved Skill Examples

Table 9 presents a case study of how MASA adapts skills differently for each backbone on the same subtask—WebShop's color-matching decision, identified as the highest-failure-rate subtask during skill evolution.

Rather than producing minor wording variations, the evolution pipeline discovers qualitatively distinct strategies tailored to each model's dominant failure mode:

- Qwen3-4B tends to guess visually similar colors. The evolved skill imposes a strict binary rule: match exactly or leave immediately.
- Qwen3-8B abandons products too easily, scoring zero. The evolved skill encourages buying approximate matches for partial credit.
- Qwen3-14B wastes steps deliberating on bad products. The evolved skill enforces a one-step fast-fail exit when no exact match exists.
- Qwen3-32B mishandles coded color names (e.g.,

`d01green`). The evolved skill provides an algorithmic substring-matching procedure with tie-breaking rules.

This demonstrates that model-conditioned adaptation operates at the level of *decision strategy*—the same problem requires fundamentally different solutions depending on how each backbone fails.

J The Use of Large Language Models (LLMs)

In this paper, large language models were utilized exclusively for grammatical polishing and stylistic refinement, aimed at enhancing the clarity and readability of our presentation of results.

The following pages contain supplementary tables and figures.

(a) General Skill: *Systematic Exploration*

Granularity	Skill Text
CONCISE	**Principle:** Search all surfaces and containers once before revisiting.
MODERATE	**Principle:** Search every plausible surface or container exactly once before revisiting; prioritize unopened or unseen locations to cover the whole room methodically. **When to apply:** Anytime the goal object count is not yet met and unexplored locations remain.
DETAILED	**Principle:** When searching for an object, follow these steps exactly: Step 1: Make a mental list of ALL possible locations in the room (countertop 1, countertop 2, shelf 1, drawer 1, cabinet 1, fridge 1, etc.). Step 2: Visit each location one by one using 'go to [location] [number]' (e.g., 'go to countertop 1'). Step 3: For closed containers (drawer, cabinet, fridge, safe, microwave), always use 'open [container] [number]' to check inside. Step 4: Read the observation carefully — look for the exact name of the target object. Step 5: Mark each location as 'checked' mentally and do NOT go back to it. Step 6: Only after checking ALL locations in the room, consider that the object may not be present. EXAMPLE: Looking for a mug → 'go to countertop 1' → check → 'go to countertop 2' → check → 'go to shelf 1' → check → 'open cabinet 1' → check inside → continue until found. **When to apply:** At the VERY START of every task that involves finding or locating any object. This is always your first action — never skip the systematic search.

(b) Task-Specific Skill: *Open Then Heat*

Granularity	Skill Text
CONCISE	**Principle:** Open microwave, put object in, heat it.
MODERATE	**Principle:** Upon reaching the microwave with the target in hand, always open the door, place the object inside, and execute the heat action before leaving. **When to apply:** Immediately after navigating to the microwave with the target object held.
DETAILED	**Principle:** The microwave heating sequence must be executed in this EXACT order: (1) 'go to microwave 1' — navigate to the microwave. (2) 'open microwave 1' — the door must be open to put things in. (3) 'put [object] in/on microwave 1' — place the object inside. (4) 'heat [object] with microwave 1' — execute the heating action. (5) 'open microwave 1' — open the door again to retrieve (if needed). (6) 'take [object] from microwave 1' — take the now-heated object. COMMON MISTAKE: Trying to 'heat' without first putting the object in the microwave → fails. ANOTHER MISTAKE: Forgetting to open the microwave before putting the object in → fails. **When to apply:** When you are holding the target object and ready to heat it. Execute this exact 6-step sequence.

Table 4: Examples from the ALFWorld skill-library variants used in the preliminary study. We show the same cross-task skill and task-specific skill under three granularity levels: CONCISE, MODERATE, and DETAILED. The empty-bank control (NO SKILL) is omitted for brevity.

Model	Skill	Pick	Clean	Heat	Cool	Pick2	Look	Overall
Qwen3-4B	No Skill	20.0	18.5	18.8	16.0	12.5	15.4	17.1
	Concise	3.2	17.9	37.5	9.1	10.7	40.0	16.4
	Moderate	20.0	29.6	12.5	20.0	8.3	30.8	**20.0**
	Detailed	1.6	16.1	9.3	6.8	10.7	46.7	12.8
Qwen3-8B	No Skill	54.3	29.6	6.2	24.0	20.8	46.2	**32.1**
	Concise	32.2	33.9	25.0	11.3	25.0	40.0	27.9
	Moderate	17.1	40.7	31.2	20.0	12.5	38.5	25.0
	Detailed	6.5	17.9	21.9	13.6	10.7	50.0	17.1
Qwen3-14B	No Skill	65.7	25.9	43.8	16.0	29.2	38.5	37.9
	Concise	74.2	26.8	18.8	13.7	30.3	43.4	36.8
	Moderate	68.6	44.4	25.0	20.0	33.3	46.2	42.1
	Detailed	64.5	46.4	18.8	34.1	46.4	66.7	**47.5**
Qwen3-32B	No Skill	48.6	44.4	25.0	32.0	16.7	46.2	36.4
	Concise	56.5	41.0	37.5	18.2	33.9	56.6	40.7
	Moderate	48.6	40.7	50.0	44.0	20.8	46.2	41.4
	Detailed	54.8	46.4	31.2	36.4	28.6	60.0	**42.9**

Table 5: Per-task ALFWorld success rate (%) for the four Qwen3 backbones under each skill granularity condition.

Model	Skill	Pick	Clean	Heat	Cool	Pick2	Look	Overall
Gemma3-4B	No Skill	22.6	0.0	0.0	0.0	7.1	40.0	10.7
	Concise	22.6	7.1	6.2	0.0	7.1	33.3	**12.1**
	Moderate	3.2	7.1	0.0	4.5	7.1	40.0	8.6
	Detailed	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Gemma3-12B	No Skill	16.1	3.6	0.0	0.0	3.6	26.7	7.9
	Concise	32.3	10.7	12.5	0.0	7.1	33.3	**15.7**
	Moderate	9.7	10.7	0.0	0.0	7.1	33.3	9.3
	Detailed	38.7	21.4	6.2	0.0	3.6	6.7	15.0
Gemma3-27B	No Skill	22.6	14.3	18.8	13.6	25.0	40.0	21.4
	Concise	61.3	21.4	31.2	9.1	42.9	46.7	36.4
	Moderate	51.6	32.1	18.8	4.5	42.9	53.3	35.0
	Detailed	41.9	60.7	31.2	31.8	35.7	66.7	**44.3**

Table 6: Per-task ALFWorld success rate (%) for the three Gemma3 backbones under each skill granularity condition.

		Per-Category SR (%)							
Model	Method	Apparel	Other	Footwear	Home	Elec.	Access.	Beauty	Avg.
Qwen3-4B	No Skill	18.6	40.0	11.1	19.0	71.4	20.0	16.7	23.0
	+ Base Skill	16.1	27.0	26.7	19.0	28.6	10.0	16.7	19.4
	+ DS-Adapter	20.3	20.0	13.3	14.3	28.6	10.0	16.7	19.2
	+ MASA	**23.2**	**38.0**	**28.9**	**19.0**	28.6	**20.0**	**16.7**	**26.4**
Qwen3-8B	No Skill	2.6	11.0	0.0	4.8	14.3	**20.0**	0.0	4.6
	+ Base Skill	5.5	9.0	0.0	0.0	14.3	**20.0**	16.7	6.0
	+ DS-Adapter	4.8	5.0	0.0	0.0	0.0	10.0	16.7	4.4
	+ MASA	**27.7**	**41.0**	**13.3**	**14.3**	**57.1**	10.0	**33.3**	**28.6**
Qwen3-14B	No Skill	1.0	6.0	6.7	4.8	0.0	10.0	0.0	2.8
	+ Base Skill	0.3	2.0	4.4	4.8	0.0	20.0	0.0	1.6
	+ DS-Adapter	0.3	6.0	2.2	4.8	0.0	10.0	0.0	2.0
	+ MASA	**32.8**	**33.0**	**6.7**	**9.5**	**28.6**	**30.0**	**16.7**	**29.2**
Qwen3-32B	No Skill	4.2	16.0	2.2	9.5	0.0	10.0	0.0	6.6
	+ Base Skill	4.8	14.0	4.4	9.5	14.3	20.0	0.0	7.2
	+ DS-Adapter	2.3	8.0	2.2	4.8	0.0	10.0	0.0	3.6
	+ MASA	**32.2**	**48.0**	**26.7**	**19.0**	**42.9**	**40.0**	**33.3**	**34.6**

Table 8: WebShop per-category success rate (%). **Bold** marks the best within each backbone.

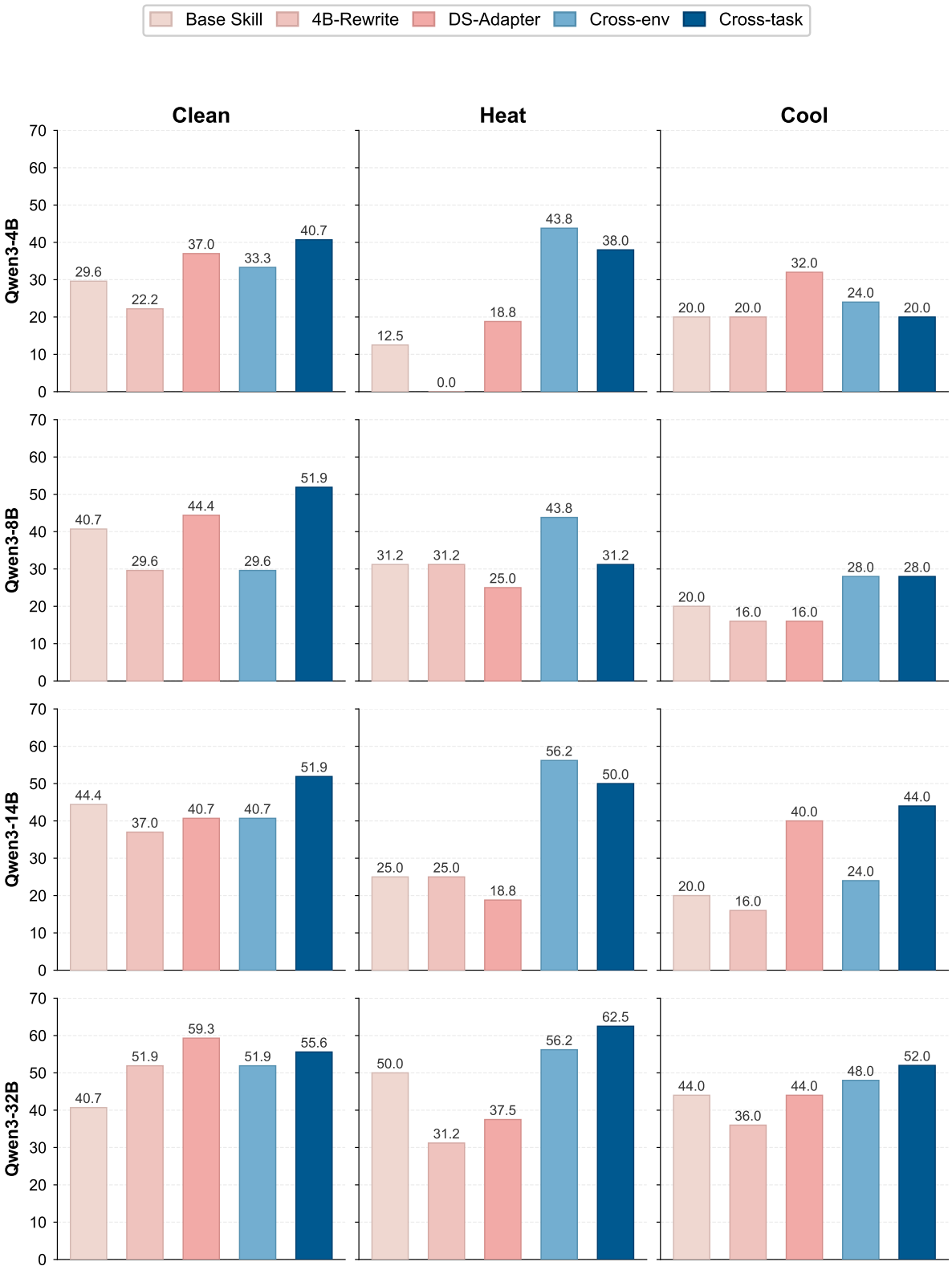

Figure 5: Per-task OOD generalization of MASA-Rewriter. Rows: target backbones (4B–32B). Columns: held-out task types (Clean, Heat, Cool).

Model	Evolved Skill Text	Failure Mode	Strategy
4B	When selecting a color variant, match the EXACT string from the task requirement to the available options. 'navy blue' $\neq$ 'light blue' $\neq$ 'navy'. 'c3-black' $\neq$ 'c-black'. If the exact color name is not available in the options list, this product CANNOT satisfy your requirement—**leave immediately**. **Do NOT select an approximate or similar color.**	Picks visually similar colors by guessing	Strict binary match: exact or leave
8B	Scan ALL color options. If the EXACT color name appears, click it. 'c3-black' is NOT the same as 'a6-black'. If the exact color is NOT available but a SIMILAR one exists (e.g., goal says 'green', options have 'e-green'), **select the CLOSEST match and proceed to buy**. A close color match gives partial credit which is better than 0. **Even if the product doesn't perfectly match—BUY IT.**	Abandons products too easily (0 credit)	Flexible match: buy anyway for partial credit
14B	If your required color is NOT in the admissible actions list, **click 'back to search' immediately**. Do not try similar colors. Do not try similar sizes. **One glance at options → if exact match missing → back to search. Takes 1 step, not 5.**	Wastes steps deliberating on bad products	Fast-fail: 1-step exit if no exact match
32B	**Match color by checking if the task's required color name appears as a SUBSTRING** in any admissible action, or vice versa. 'patina green' matches 'patina green' (exact) but NOT 'yellow'. 'green' matches 'a1-green' or 'd01green' (contains). **When multiple options contain the color word, prefer the one that matches more of the full color name.**	Mishandles coded names (e.g. d01green)	Algorithmic: substring matching with preference rule

Table 9: WebShop color-matching skill evolved by MASA for four backbones. Red: rigid rejection rule (4B); Blue: flexible buy-anyway heuristic (8B); Teal: 1-step fast-fail exit (14B); Violet: algorithmic substring matching (32B). Each strategy targets the dominant failure mode of its target model.