SELF-EVOLVING MULTIMODAL SKILL ROUTING WITH DUAL-TIMESCALE EXPERIENCE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

External skill libraries expand the capabilities of multimodal agents, but routing queries to the accurate multimodal skills becomes increasingly challenging as these libraries scale. Existing static routers leave post-execution feedback untapped, preventing routing policies from improving with experience. In this work, we formalize self-evolving multimodal skill retrieval, in which a router continually adapts multimodal query and skill representations through execution feedback. Our key insight is that these representations should evolve at different timescales: queries benefit from rapid adaptation leveraging past interactions of similar requests, whereas skill profiles demand stable semantic consolidation across diverse task contexts. Motivated by this, we propose MM-RouterEvolve, a new self-evolving framework that couples query-side and skill-side adaptation through a dual-timescale experience memory. It organizes multimodal queries, retrieved skills, and feedback verifying their compatibility into unified experience units. A bounded short-term memory uses recent routing outcomes to contextualize incoming queries, while a persistent long-term memory consolidates execution feedback to refine skill profiles and distinguish functionally similar skills. This design connects immediate routing adaptation with cumulative refinement of the skill library. Evaluated on two benchmarks, i.e., MM-SkillBench-SE (multimodal) and SkillRouter (text-only), MM-RouterEvolve substantially outperforms static baselines and unilateral query/skill adaptation paradigms, showing that experience-driven self-evolution of both query and skill representations yields more robust routing.

1 INTRODUCTION

Multimodal agents (Yao et al., 2023; Schick et al., 2023; Wang et al., 2024; Zhou et al., 2024) increasingly rely on external skill libraries to access specialized procedures, domain-specific tools, and expert capabilities required for complex tasks. As these libraries grow, accurately identifying the appropriate skill for an incoming user request becomes critical for scalable agent systems.

Early efforts (Qin et al., 2024; Shi et al., 2025; Zheng et al., 2026) typically tackle this challenge within classic retrieval pipelines, framing skill routing as matching textual queries against skill descriptions. These approaches mostly rely on static representations of queries and skills, and treat each routing decision in isolation. Consequently, useful feedback from previous executions remains unused, preventing the router from improving with this feedback. As illustrated in Fig. 1(A), although a prior execution has resolved an ambiguity by ruling out an incorrect skill, a static router fails to exploit this experience for subsequent, semantically related queries. Such ambiguities thus inevitably recur rather than being converted into reusable routing expertise.

In this paper, we investigate this dilemma beyond text-based paradigms in the broader setting of multimodal skill routing, where queries and skills are coupled with visual contexts, e.g., reference images for queries and capability demonstrations for skills. Unlike static routers, we formalize self-evolving multimodal skill retrieval and propose continually adapting both query and skill with verified feedback from previous routing executions over time. Crucially, we recognize that this adaptation should occur at different timescales for queries versus skills. User queries often carry ambiguous or incomplete descriptions. They benefit from rapid adaptation by retrieving similar requests and their verified outcomes from recent interactions, which provides immediate guidance on

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

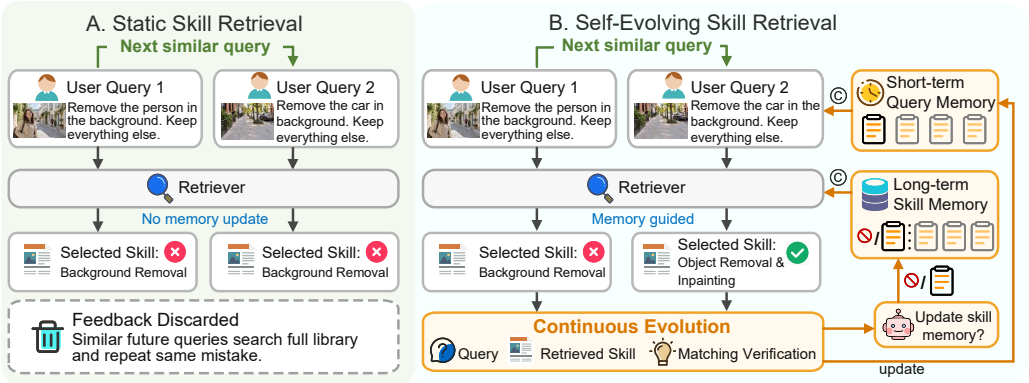

Figure 1: Comparison between static skill retrieval and our self-evolving MM-RouterEvolve. (A) Static router discards post-execution feedback and relies solely on frozen query/skill representations, leaving recurring matching ambiguities unresolved across similar requests. (B) MM-RouterEvolve transforms verified routing feedback into evolutionary signals via a dual-timescale memory architecture, combining short-term query contextualization with long-term skill profile refinement.

which skills succeeded or failed in similar contexts. In contrast, skills are shared, reusable assets across diverse tasks, meaning that rapidly updating their profiles based on short-term feedback risks overfitting to local noise and distorting their core capabilities. This tension motivates a novel memory design: an agile short-term buffer for query contextualization and a stable long-term store for skill consolidation.

Specifically, we present MM-RouterEvolve that progressively co-evolves query and skill representations grounded in interaction outcomes through a dual-timescale experience memory, thereby steadily improving skill matching over time. The conceptual overview is shown in Fig. 1(B). MM-RouterEvolve first encapsulates each previous routing feedback into a unified experience unit containing the multimodal query, retrieved skill and matching verification. These structured experiences are subsequently leveraged to jointly drive the evolution of both query and skill representations. On the query side, MM-RouterEvolve maintains a bounded short-term memory of recent routing experiences, organized as a rolling, fixed-capacity sliding window. When an incoming query arrives, experiences from similar prior requests are retrieved to adapt its representation, enriching ambiguous intents with operational context and disambiguation cues for more precise skill matching. On the skill side, MM-RouterEvolve maintains a persistent long-term memory for each skill. Rather than overwriting canonical definitions based on recent interactions, this memory aggregates routing outcomes across diverse tasks. These traces are combined with the original skill specifications to incrementally calibrate skill representations. By continuously incorporating feedback from routing decisions improved through representation co-evolution, MM-RouterEvolve drives a self-reinforcing cycle of routing refinement without offline model retraining. To facilitate training, we first collect $\sim$35K textual skills, augmenting each skill with a visual card demonstrating its capabilities and procedures, and synthesizing corresponding multimodal queries (e.g., a source image and instruction for image editing). We then construct relevant/irrelevant variants for each query, and pair both queries and skills with verified successful/unsuccessful retrieval execution entries, yielding a comprehensive benchmark MM-SkillBench-SE.

Our contributions are summarized as follows: (1) We present MM-RouterEvolve, a self-evolving multimodal skill routing framework that turns post-execution feedback from passive system outcomes into an active evolutionary signal for adapting both query and skill representations. Moreover, we propose a novel dual-timescale experience memory that couples short-term query contextualization with long-term skill refinement. (2) We introduce MM-SkillBench-SE, a large-scale benchmark of approximately 35K skills for multimodal skill routing, which extends text-only query/skill with visual contexts and verified routing experiences. (3) Extensive experiments on MM-SkillBench-SE demonstrate that MM-RouterEvolve consistently outperforms both static routing baselines and single-sided adaptation paradigms across multiple evolutionary rounds through joint query–skill self-evolution. Additional zero-shot evaluation on the text-only SkillRouter benchmark further validates the generalizability of our method.

2 Related Work

2.1 Multimodal Retrieval

Early text-based retrieval methods (Mikolov et al., 2013; Pennington et al., 2014) map linguistic inputs to static, fixed-dimensional vector spaces, while subsequent multimodal approaches (Radford et al., 2021; Li et al., 2022; Zhai et al., 2023) extend this paradigm to vision-language matching by projecting images and text into a shared representation space via dual encoders. Recent work has focused on developing universal multimodal embedding models (Jiang et al., 2025; Liu et al., 2025; Gu et al., 2025) built upon multimodal large language models (MLLMs). VLM2Vec (Jiang et al., 2025) adapts MLLMs into general-purpose multimodal embedding models through contrastive training, supporting retrieval across text, images, and mixed multimodal inputs. Later, LamRA (Liu et al., 2025) repurposes MLLMs for both retrieval and reranking, combining multimodal instruction tuning with pointwise and listwise ranking objectives. More recently, Qwen3VL-Emb (Li et al., 2026) introduces a unified multimodal embedding model that supports not only text and image retrieval but also video retrieval.

2.2 Tool and Skill Retrieval in Agents

Early tool-learning studies focus on enabling LLMs to understand and invoke external APIs. Gorilla (Patil et al., 2024) and ToolLLM (Qin et al., 2024) investigate large-scale tool learning by enabling LLMs to understand API specifications, generate tool calls, and interact with external functions through tool-use data and instruction tuning. As tool libraries grow, recent work has moved beyond tool understanding and execution to the study of retrieval-based tool selection at scale. For instance, ToolRerank (Zheng et al., 2024) introduces adaptive reranking for tool candidates by leveraging tool hierarchy and structural information. Other efforts explore contrastive learning (Gao et al., 2021; Wang et al., 2022) or instruction-based tool matching (Qin et al., 2024) to address this challenge. In addition to individual tools, recent agents organize reusable capabilities as skills, which include not only tool interfaces but also task descriptions, execution procedures, and constraints. As one of the pioneering methods targeting skill retrieval, SkillRouter (Zheng et al., 2026) reveals that short skill descriptions fail to distinguish similar skills while full skill profiles provide richer retrieval signals, motivating its text-based routing model. R3-Skill (Wang et al., 2026) further improves skill routing by exploiting rejected skills for a given query to learn finer-grained query–skill matching.

2.3 Agent Self-Improvement

Existing methods primarily explore agent self-improvement by optimizing different components within the agent framework. At the component level, for instance, OPRO (Yang et al., 2024) iteratively refines an LLM-based prompt enhancer using previously evaluated instructions and their execution scores. Voyager (Wang et al., 2024) autonomously discovers, documents, and reuses executable skills in open-ended environments through an evolving skill library. Chain-of-Verification (Dhuliawala et al., 2024) strengthens evaluation reliability by deriving factual checks from an initial response, answering them separately, and producing a corrected final response. At a broader system level, EvoAgentX (Wang et al., 2025) provides a unified framework for multi-agent workflow optimization by integrating existing algorithms such as TextGrad (Yuksekgonul et al., 2024), AFlow (Zhang et al., 2025a), and MIPRO (Opsahl-Ong et al., 2024) to refine agent prompts, configurations and workflow structures based on task-level evaluation.

3 Method

3.1 Overview

In this work, we study multimodal skill routing in modern agent systems. Let $\mathcal{S} = \{s_j\}_{j=1}^{N}$ be a large pool of N skills, where each skill s_j comprises a text summary, an execution schema, and a visual skill card that vividly demonstrates its capabilities and workflow. For a query q_i consisting of a textual instruction and optional visual inputs (e.g., a reference image for personalized content generation), the router aims to retrieve the optimal skill $s^* \in \mathcal{S}$ to perform the task.

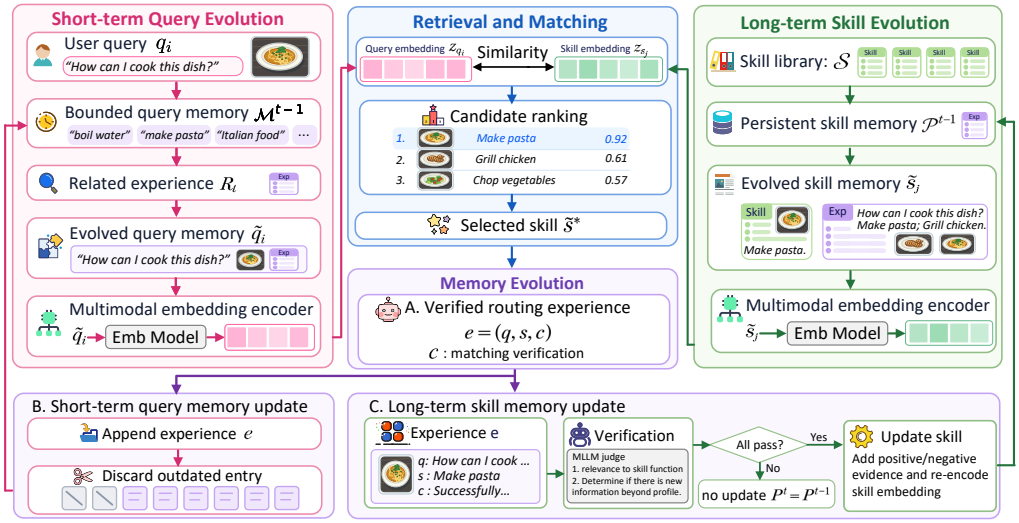

Figure 2: Overview of our MM-RouterEvolve. The framework enables self-evolving multimodal skill retrieval via a dual-timescale experience memory. Specifically, a short-term query memory adapts query representations using recent routing experiences from similar requests, while a long-term skill memory refines skill profiles using accumulated feedback across diverse task contexts. The co-evolved query and skill representations are matched to retrieve skills, with post-execution verification updating both memories to sustain a self-reinforcing improvement cycle.

Addressing the limitations of static retrieval features, we formulate multimodal skill routing as a self-evolving retrieval process over multiple rounds, where verified execution outcomes are treated as experience to improve subsequent query–skill matching. We therefore introduce **MM-RouterEvolve** to jointly evolve query and skill embeddings through a dual-timescale experience memory, which consists of a short-term query-side memory $\mathcal{M}$ and a long-term skill-side memory $\mathcal{P}$. Both memories are updated over time and leveraged to dynamically adapt features for incoming queries and candidate skills.

At the t-th round, entries from $\mathcal{M}^{t-1}$ and $\mathcal{P}^{t-1}$ are retrieved to augment the query q_i and candidate skill $s_j \in \mathcal{S}$ into enriched representations $\tilde{q}_i^t$ and $\tilde{s}_j^t$, respectively. Particularly, for each skill s_j, there is a dedicated entry set $P_j^{t-1} \in \mathcal{P}^{t-1}$. Next, a shared embedding model f_θ, built upon a multimodal large language model (MLLM) (Li et al., 2026), encodes both inputs into $z_{q_i}^t = f_\theta(\tilde{q}_i^t)$ and $z_{s_j}^t = f_\theta(\tilde{s}_j^t)$. The routing score between the query and candidate skill is computed via inner product: $\phi(z_{q_i}^t, z_{s_j}^t) = (z_{q_i}^t)^\top z_{s_j}^t$, and the optimal skill is selected as $\tilde{s}_t^* = \arg\max_{s_j \in \mathcal{S}} \phi(z_{q_i}^t, z_{s_j}^t)$.

Upon task execution, the routing outcome is verified as a success or failure, yielding an experience entry $e_t = (q_i, s_j, c_{ij})$, where c_{ij} indicates the feedback. Based on consolidation rules of different time scales, e_t is selectively incorporated into $\mathcal{M}^{t-1}$ and $\mathcal{P}^{t-1}$, yielding updated memories $\mathcal{M}^t$ and $\mathcal{P}^t$. These refreshed memories are subsequently utilized in future interactions to progressively improve query-skill matching without online model fine-tuning. The overview is shown in Figure 2.

3.2 Self-Evolving Multimodal Retrieval with Dual-Timescale Memory

Our proposed MM-RouterEvolve couples query-side and skill-side adaptation through a dual-timescale experience memory.

Query Evolution via Short-term Memory. The query-side memory serves as a short-term, verified context store that helps interpret incoming requests. Before interaction t, MM-RouterEvolve maintains a bounded memory

$$\mathcal{M}^{t-1} = \{e_k\}_{k=1}^{|\mathcal{M}^{t-1}|}, \qquad |\mathcal{M}^{t-1}| \le K,$$

where $e_k = (q_k, s_k, c_k)$ represents a verified routing experience and K is the memory capacity.

Given the current query q_i, a retriever $R(\cdot)$ first selects the most relevant experiences from the memory $\mathcal{M}^{t-1}$. The query is then augmented with these retrieved experiences and mapped into the embedding space by the shared multimodal encoder f_θ. This process is formalized as:

$$R_t = R(q_i, \mathcal{M}^{t-1}), \qquad \tilde{q}_i^t = [q_i; R_t], \qquad z_{q_i}^t = f_\theta(\tilde{q}_i^t). \tag{1}$$

The contextualized representation $z_{q_i}^t$ is then utilized to match the top candidate skill $\tilde{s}_t^*$ from the skill library. Once the interaction is executed and verified, the resulting experience $e_t = (q_i, \tilde{s}_t^*, c_t)$ is incorporated into the memory through a rolling update operator U_q:

$$\mathcal{M}^t = U_q(\mathcal{M}^{t-1}, e_t), \qquad |\mathcal{M}^t| \le K.$$

When the memory reaches capacity K, the oldest entries are discarded, keeping the memory focused on recent interactions and reducing the impact of outdated history.

This bounded design directly reinforces query self-evolution. As routing confusions are progressively resolved over multiple rounds, recent memory entries become more reliable. By prioritizing these fresh experiences, the retriever $R(\cdot)$ provides more accurate context for new queries. This establishes a positive feedback loop: refined past experiences improve current query representations, leading to more accurate routing and continuously boosting query–skill matching.

Skill Evolution via Long-term Memory. The skill-side memory accumulates verified evidence across diverse task contexts to refine skill profiles. Before interaction t, each skill s_j is associated with a profile in the long-term memory: $P_j^{t-1} = (C_j, E_j^{t-1}) \in \mathcal{P}^{t-1}$, where C_j represents the static multimodal skill content and E_j^{t-1} stores historical execution outcomes linked to this skill. To prepare for retrieval at interaction t, the base profile C_j is augmented with its accumulated experiences E_j^{t-1} and encoded by the shared multimodal model f_θ:

$$\tilde{s}_j^t = [C_j; E_j^{t-1}], \qquad z_{s_j}^t = f_\theta(\tilde{s}_j^t). \tag{2}$$

Because the canonical content C_j remains untouched, this design grounds skill evolution in interaction feedback while preserving the core semantics of the original skill definitions.

Let s_j denote the retrieved skill in the experience tuple e_t at interaction t. MM-RouterEvolve employs an experience admission function $G(\cdot)$ as a controller to determine whether e_t should be integrated into E_j^{t-1}. We instantiate $G(\cdot)$ using an MLLM that evaluates the incoming experience e_t against the current profile P_j^{t-1} under two key criteria: (i) whether e_t is relevant to the function of s_j, and (ii) whether it provides new information beyond the current profile. Experiences that are irrelevant or redundant are rejected. The process can be formalized as

$$\bar{e}_t = G(P_j^{t-1}, e_t), \tag{3}$$

where $\bar{e}_t = e_t$ if the experience is accepted, and $\bar{e}_t = \emptyset$ if it is rejected. Upon admission, the approved experience is appended to the persistent memory of skill s_j, while a rejected experience leaves the profile unchanged:

$$E_j^t = E_j^{t-1} \cup \bar{e}_t, \qquad P_j^t = (C_j, E_j^t). \tag{4}$$

Both positive and negative experiences are intentionally collected: positive experiences identify relevant task scenarios for the skill, while negative ones mark contexts where the skill is unsuitable.

This design is motivated by the need to balance stability and adaptability in skill profiles. Unlike queries that benefit from rapid, short-term adaptation, skills are reused across diverse tasks and require reliable long-term consolidation. By selectively admitting only high-quality experiences over extended operational periods, the skill memory progressively enriches the static profile. This steady accumulation sharpens the distinction between functionally similar skills, leading to more accurate query–skill matching.

Training. We first optimize a standard retrieval objective that assigns a higher score to the ground-truth skill than to the remaining candidates:

$$\mathcal{L}_{\text{ret}} = -\log \frac{\exp\left(\phi(z_{q_i}, z_{s_j}^\star)/\tau\right)}{\sum_{s_j \in \mathcal{C}_i} \exp\left(\phi(z_{q_i}, z_{s_j})/\tau\right)}, \tag{5}$$

where $z^{\star}_{s_j}$ denotes the representation of the ground-truth skill, and τ is the temperature parameter. where $\mathcal{C}_i$ consists of the ground-truth skill $s^{\star}_j$ and the negative skills used for training query q_i.

We further collect routing errors across held-out folds and use the incorrectly retrieved skills as hard negatives. Each misrouted query q_i, together with its ground-truth skill $s^{\star}_j$ and the incorrectly retrieved skill s^{-}_j, forms a hard-negative triplet $(q_i, s^{-}_j, s^{\star}_j)$. For each hard-negative triplet, we impose a margin constraint that separates the ground-truth skill from the incorrectly retrieved skill:

$$\mathcal{L}_{\text{hard}} = \max\left(0,\, m - \phi(z_{q_i}, z^{\star}_{s_j}) + \phi(z_{q_i}, z^{-}_{s_j})\right), \tag{6}$$

where $z^{-}_{s_j}$ is the representation of the incorrectly retrieved skill and m is the ranking margin. This objective explicitly trains the retriever on errors arising from functionally similar candidates rather than relying solely on conventional hard-negative mining. The overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{ret}} + \lambda_{\text{hard}}\mathcal{L}_{\text{hard}}, \tag{7}$$

where λ_{hard} controls the contribution of the hard-negative supervision.

3.3 Multimodal Skill Routing Benchmark with Experiences

We construct a large-scale benchmark, MM-SkillBench-SE, to facilitate the study of self-evolving multimodal skill routing. First, we curate a diverse repository of textual skills. An LLM-based semantic parser extracts a schema z_j from each raw skill source and distills it into a concise textual summary t_j of its core capability. We further synthesize a visual skill card v_j that captures key entities, execution procedures, input–output relations, and capability constraints, yielding an informative multimodal profile $s_j = (t_j, z_j, v_j)$ for each skill. Conditioned on s_j, we prompt an LLM to generate the corresponding textual request x_j. We pair each x_j with an optional visual context I_j generated by an image generator when applicable, yielding the multimodal query $q_j = (x_j, I_j)$.

Building upon these query–skill pairs, we synthesize routing experiences to enable learning from past execution feedback. For each skill s_j, we construct a set containing diverse queries that invoke the same skill capability. To simulate routing outcomes on unseen queries, we split the initial pairs into five folds, train a base retrieval model on four folds, and evaluate it on the held-out fold to collect retrieval errors. Compared with randomly sampled negatives, these mined errors capture realistic retrieval confusions and provide informative hard cases for modeling fine-grained query–skill boundaries. Each routing execution is recorded as a verified experience unit $e_m = (q_m, s_m, c_m)$, where c_m indicates the verification outcome (success or failure) between query q_m and the retrieved skill s_m. Aggregating these entries yields the complete experience memory, providing faithful supervision for self-evolving retrieval optimization. More details are provided in Appendix A.3.

4 Experiments

4.1 Experimental Setup

Benchmark and Evaluation Protocol. We evaluate MM-RouterEvolve on two skill router benchmarks: MM-SkillBench-SE and SkillRouter (Zheng et al., 2026). MM-SkillBench-SE is a multimodal benchmark in which queries contain text, input images, while each candidate skill is represented by a textual summary, a structured schema, a visual skill card. SkillRouter is a text-only benchmark in which both queries and candidate skills are represented by textual descriptions. More benchmark details are provided in Appendix A.3. For both benchmarks, all methods are evaluated by retrieving and ranking skills from a large-scale candidate pool for each query. We report Hit@1, MRR@10, and Recall@K with $K \in \{10, 20, 50\}$. Hit@1 measures whether a valid skill is ranked first. MRR@10 computes the reciprocal rank of the highest-ranked valid skill when it appears within the top 10 positions, and is zero otherwise. Recall@K indicates whether at least one valid skill is retrieved within the top K results. Higher values indicate better retrieval performance for all metrics.

Baselines. We compare MM-RouterEvolve against a broad set of retrieval baselines. For general-purpose multimodal retrieval, we include VLM2Vec-V2 (Meng et al., 2026), UniME-V2 (Gu et al., 2026), Ops-MM-Embedding-v1 (OpenSearch-AI, 2025), and Qwen3-VL-Embedding with 2B and 8B parameters (Li et al., 2026) We further evaluate the text-only Qwen3-Embedding models at 0.6B

Table 1: Comparison of skill retrieval performance on MM-SkillBench-SE.

Model	Params	Hit@1 ↑	MRR@10 ↑	R@10 ↑	R@20 ↑	R@50 ↑
VLM2Vec-V2 (Meng et al., 2026)	2B	0.270	0.349	0.540	0.630	0.732
UniME-V2 (Gu et al., 2026)	2B	0.298	0.382	0.562	0.636	0.740
Ops-MM-Embedding-v1 (OpenSearch-AI, 2025)	8B	0.414	0.503	0.690	0.756	0.830
Qwen3VL-Emb-2B (Li et al., 2026)	2B	0.438	0.529	0.722	0.788	0.862
Qwen3VL-Emb-8B (Li et al., 2026)	8B	0.472	0.558	0.766	0.816	0.872
SkillRet (Cho et al., 2026)	0.6B	0.518	0.595	0.764	0.816	0.874
R3-Skill (Wang et al., 2026)	0.6B	0.570	0.648	0.806	0.872	0.914
MM-RouterEvolve	2B	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**

Table 2: Comparison of skill retrieval performance on the SkillRouter hard setting. All methods are evaluated using text-only queries and skills.

Model	Params	Hit@1 ↑	MRR@10 ↑	R@10 ↑	R@20 ↑	R@50 ↑
VLM2Vec-V2 (Meng et al., 2026)	2B	0.453	0.536	0.582	0.622	0.700
UniME-V2 (Gu et al., 2026)	2B	0.493	0.593	0.592	0.635	0.728
Ops-MM-Embedding-v1 (OpenSearch-AI, 2025)	8B	0.640	0.720	0.673	0.723	0.767
Qwen3VL-Emb-2B (Li et al., 2026)	2B	0.587	0.655	0.639	0.731	0.802
Qwen3VL-Emb-8B (Li et al., 2026)	8B	0.560	0.667	0.675	0.727	0.766
Qwen3-Emb-0.6B Zhang et al. (2025b)	0.6B	0.573	0.651	0.644	0.706	0.747
Qwen3-Emb-8B Zhang et al. (2025b)	8B	0.642	0.713	0.683	0.745	0.796
SkillRet (Cho et al., 2026)	0.6B	0.613	0.687	0.705	0.736	0.809
R3-Skill (Wang et al., 2026)	0.6B	0.627	0.701	0.693	0.736	0.821
SkillRouter-0.6B (Zheng et al., 2026)	0.6B	0.647	0.677	0.658	0.726	0.761
SkillRouter-8B (Zheng et al., 2026)	8B	0.660	0.721	0.690	0.750	0.815
MM-RouterEvolve	2B	**0.663**	**0.742**	**0.715**	**0.759**	**0.851**

and 8B scales. For skill retrieval baselines, we compare against SkillRet (Cho et al., 2026), R3-Skill (Wang et al., 2026), and SkillRouter (Zheng et al., 2026) in both its 0.6B and 8B variants.

Training and Inference Details. MM-RouterEvolve is built upon Qwen3-VL-Embedding-2B (Li et al., 2026) and fine-tuned for multimodal skill retrieval. We train the model for one epoch across 8 GPUs with a total batch size of 128, using a learning rate of 10^{-5} and a warmup ratio of 3%. For each query, two negatives are sampled. During self-evolving inference, the trained model remains frozen, while representations for both queries and skills are adapted via our proposed dual-timescale experience memory. On the skill side, positive and negative execution outcomes across diverse tasks are accumulated and validated through an MLLM-based interface (Qwen3-VL-2B Team (2025)). Only verified outcomes are incorporated into the long-term memory to update the corresponding skill representations.

4.2 Main Results

Results on MM-SkillBench-SE. As shown in Table 1, MM-RouterEvolve achieves the best retrieval performance across all five metrics on MM-SkillBench-SE. Compared with R3-Skill, the most competitive multimodal retrieval baseline, MM-RouterEvolve improves Hit@1 from 0.570 to 0.703, MRR@10 from 0.648 to 0.785, and R@10 from 0.806 to 0.896, achieving absolute gains of 13.3%, 13.7%, and 9.0% respectively. The more pronounced gains in Hit@1 and MRR@10 highlight the superiority of MM-RouterEvolve in fine-grained discrimination among confusing candidate skills, ensuring higher routing accuracy at the top rank. Compared with its base model Qwen3VL-Emb-2B, MM-RouterEvolve improves Hit@1 from 0.438 to 0.703 and MRR@10 from 0.529 to 0.785. Scaling the static backbone to Qwen3VL-Emb-8B yields only moderate gains (with Hit@1 rising from 0.438 to 0.472 and MRR@10 from 0.529 to 0.558), yet it still lags far behind our 2B-based MM-RouterEvolve despite having four times the parameters. This comparison indicates that simply scaling a static retriever cannot bridge the performance gap, whereas self-evolution through routing experiences provides much more effective adaptation for distinguishing closely related skills.

Results on SkillRouter. Table 2 reports performance under the hard setting of the text-only benchmark SkillRouter, which introduces distractor skills that exhibit superficial topical overlap with the target skill but differ in underlying functionality. This benchmark evaluates whether a retriever can distinguish subtle functional details instead of relying solely on lexical similarity. Under this challenging setting, MM-RouterEvolve achieves the best performance across all five metrics. Specifically, it surpasses SkillRet by improving Hit@1 from 0.613 to 0.663 and MRR@10 from 0.687 to 0.742, achieving relative gains of 8.2% and 8.0%, respectively. Notably, despite relying on a 2B backbone without text-specific optimization, MM-RouterEvolve outperforms SkillRouter-8B, an 8B baseline specifically trained for text-based retrieval, achieving absolute gains of 2.1% and 2.5% on

Table 3: Ablation study of MM-RouterEvolve on MM-SkillBench-SE.

Group	Setting	Hit@1 ↑	MRR@10 ↑	R@10 ↑	R@20 ↑	R@50 ↑
(A)	Static	0.594	0.668	0.818	0.854	0.882
	Query evolution only	0.644	0.736	0.871	0.893	0.913
	Skill evolution only	0.651	0.742	0.876	0.900	0.944
	Dual-timescale evolution	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**
(B)	No experience	0.594	0.668	0.818	0.854	0.882
	Negative experience only	0.657	0.731	0.687	0.874	0.923
	Positive experience only	0.670	0.744	0.806	0.893	0.929
	Positive + Negative experience	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**
(C)	Always	0.668	0.701	0.812	0.848	0.898
	Confidence-based	0.697	0.752	0.858	0.873	0.913
	MLLM-based	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**

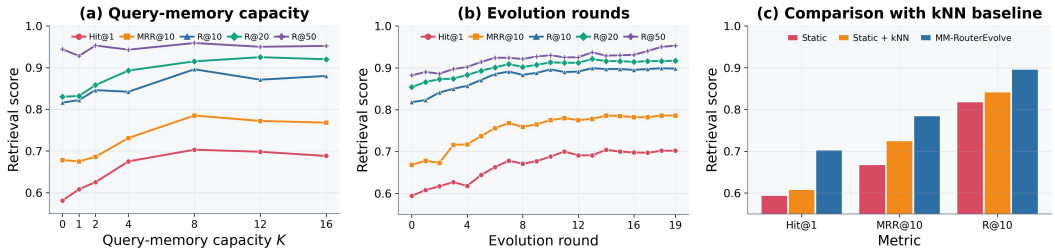

Figure 3: Analysis of query memory capacity, evolution dynamics, and self-evolution.

MRR@10 and R@10, respectively. These results confirm our self-evolution mechanism remains effective even in text-only skill routing, showing broad applicability across different modalities. More results under the easy setting can be found in Appendix A.1.

4.3 Ablation Study and Discussions

Effect of Dual-Timescale Evolution. Table 3 (A) isolates the contributions of query-side and skill-side evolution in MM-RouterEvolve. Relative to the static baseline, enabling query evolution alone raises Hit@1 from 0.594 to 0.644 and MRR@10 from 0.668 to 0.736. On the skill side, evolving representations in isolation yields even larger improvements, pushing Hit@1 to 0.651 and MRR@10 to 0.742. Combining both mechanisms yields the best overall performance, reaching 0.703 in Hit@1 and 0.959 in R@50. These consistent gains verify that the two evolution processes provide mutually complementary benefits. Adapting user queries with historical experience from similar requests resolves immediate intent ambiguities, whereas accumulating execution feedback across diverse tasks invoking the same skill calibrates skill representations for reliable matching.

Effect of Positive/Negative Experiences. Table 3 (B) examines how different routing outcomes contribute to our memory-driven self-evolution. **Positive** experiences represent successful retrievals, whereas **negative** ones capture failed routing attempts. Compared with the baseline without experience, relying solely on negative experience lifts Hit@1 from 0.594 to 0.657 and MRR@10 from 0.668 to 0.731. Meanwhile, positive experience alone brings more pronounced improvements, driving Hit@1 and MRR@10 to 0.670 and 0.744, respectively. Combining both forms of feedback (**Positive + Negative**) achieves the best overall performance, increasing Hit@1 from 0.594 to 0.703 and R@10 from 0.818 to 0.896. These results reveal distinct functional roles for the two feedback sources. Successful retrievals reinforce valid semantic mappings, helping the model recognize diverse user intents that target the same skill. In contrast, failed routing attempts penalize spurious correlations, preventing the retriever from selecting superficially plausible distractors.

Effect of Experience Admission Strategies in Skill Memory. Table 3 (C) examines three distinct strategies for admitting experiences into long-term skill memory: **Always**, **Confidence-based**, and **MLLM-based**. Specifically, **Always** retains all entries, **Confidence-based** filters candidates using embedding similarity with a threshold of 0.7, and **MLLM-based** assesses each experience for skill relevance and informativeness via an MLLM. Unconditionally incorporating all experiences (Always) yields 0.668 in Hit@1 and 0.701 in MRR@10. Confidence-based filtering improves these metrics to 0.697 and 0.752, respectively, demonstrating that discarding noisy experiences alleviates

memory corruption and reduces false matches. MLLM-based admission delivers the strongest performance across all metrics, lifting MRR@10 to 0.785 and R@50 to 0.959. This advantage stems from the deep multimodal comprehension of MLLMs. While embedding similarity scores primarily capture surface-level relevance, the MLLM assesses whether an experience is functionally relevant and introduces non-redundant information to the existing skill memory. This selective admission provides high-quality context for updating skill representations.

Effect of Query Memory Capacity. We examine the sensitivity to query memory capacity K in Figure 3 (a). As K grows from 0 to 8, retrieval performance improves steadily, with Hit@1 rising from 0.581 to 0.703. This gain indicates that a larger capacity allows the model to gather richer context from recent interactions, providing more sufficient signals to disambiguate incoming queries and adapt their representations. However, expanding the capacity further to $K = 12$ and $K = 16$ leads to a slight decline across top-ranked metrics (Hit@1, MRR@10, and R@10). This performance drop occurs because an overly large buffer introduces more irrelevant candidate experiences, which increases distractors and lowers the accuracy of matching truly relevant entries for the current query. This trend supports our design of modeling query-side memory as a bounded short-term store. We therefore set $K = 8$ as the default query memory capacity.

Effect of Evolution Rounds. Figure 3 (b) shows how performance improves as our dual-timescale memory accumulates routing experience over successive rounds. Round 0 represents the cold-start baseline where all memories are empty ($\mathcal{M}^0 = \emptyset$ and $E_j^0 = \emptyset$), meaning routing relies entirely on the static embedding model. In each subsequent round, verified outcomes are incorporated into the query and skill memories to assist future routing decisions. Performance rises notably during the initial rounds, with Hit@1 increasing from 0.594 at round 0 to 0.627 at round 3. This early gain indicates that introducing even a small set of verified experiences effectively supplies context, helping resolve ambiguous queries where the static embedding model struggles. As interactions accumulate, retrieval metrics continue to advance, reaching 0.703 in Hit@1 by round 19. The slower growth in later rounds occurs for two main reasons. First, as the memory expands, newly acquired experiences increasingly overlap with existing ones, yielding diminishing information gains. Second, while the system has successfully resolved most requests that the static model failed on, the remaining errors consist largely of exceptionally difficult cases, leaving less room for further improvement.

Effect of Experience-driven Self-evolution. We compare MM-RouterEvolve with a kNN memory retrieval baseline to examine whether directly retrieving similar historical experiences is sufficient for multimodal skill routing. As shown in Figure 3 (c), augmenting the static retriever with kNN retrieved experiences improves Hit@1 from 0.594 to 0.608 and MRR@10 from 0.668 to 0.725, confirming that past routing feedback provides helpful context signals. However, MM-RouterEvolve achieves much higher performance, boosting Hit@1 and MRR@10 to 0.703 and 0.785, respectively. This substantial gap indicates that effectively leveraging experience requires more than naive similarity matching. In contrast to a static pool that treats past interactions as fixed references, MM-RouterEvolve enables self-evolution through a dual-timescale experience memory. This mechanism allows the system to continuously resolve decision confusions across successive rounds, transforming newly refined experiences into stronger routing capability.

5 Conclusion

In this work, we formalized self-evolving multimodal skill retrieval, where post-execution feedback is transformed into active experience to continually improve query–skill matching. We proposed MM-RouterEvolve, a self-evolving framework that organizes multimodal queries, retrieved skills, and execution feedback into unified experience units, and jointly adapts query and skill representations without retraining the underlying retriever. Driven by their distinct adaptation requirements, MM-RouterEvolve employs a dual-timescale experience memory. Specifically, a bounded short-term memory rapidly contextualizes incoming queries using recent routing outcomes, while persistent long-term memories consolidate verified feedback to refine skill profiles. In addition, we present a dedicated benchmark, termed MM-SkillBench-SE, to support experience-driven multimodal skill routing. Experiments on MM-SkillBench-SE and the text-only SkillRouter benchmark show that MM-RouterEvolve consistently outperforms static retrieval baselines and unilateral query/skill adaptation paradigms, achieving sustained gains across successive evolution rounds.

AI USE STATEMENT

Generative AI tools were used to aid and polish the writing of this paper and to generate synthetic data used in dataset construction. In this work, we used GPT-5.6 (OpenAI, 2026) during manuscript preparation to improve grammar, sentence structure, and clarity. The authors take full responsibility for the paper's scientific content, technical accuracy, and conclusions. For MM-SkillBench-SE construction, we used Gemma4-31B (Team, 2026) to assist in generating candidate query variations and refining textual skill descriptions, and Qwen-Image (Wu et al., 2025) to generate visual summaries for queries and skill representations. Generated queries and visual content underwent quality filtering to ensure consistency with the corresponding skill functionalities and benchmark objectives. The final benchmark only includes content that satisfies these quality criteria.

ETHICS STATEMENT

We have read the ICLR 2027 Code of Ethics and confirm our adherence to it. This work studies self-evolving multimodal skill retrieval, with the goal of improving how multimodal agents identify appropriate skills from large-scale skill libraries. The research does not involve human subjects or personally identifiable information. As generative models are used to construct part of the synthetic data, we apply filtering and verification procedures to reduce potential generation errors and biases. In practical deployments of multimodal agent systems with skill routing, appropriate safety checks and permission controls should be incorporated to mitigate potential risks associated with unintended skill execution. To the best of our knowledge, all datasets, pretrained models, and external resources used in this work are employed in accordance with their respective licenses, and all related work has been properly cited.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed method, data construction procedure, training settings, evaluation protocols, and ablation configurations in the main paper and appendix. To further support reproducibility, we will release the code, dataset, data-processing scripts, and evaluation pipeline associated with this work.

REFERENCES

Hongcheol Cho, Ryangkyung Kang, and Youngeun Kim. Skillret: A large-scale benchmark for skill retrieval in llm agents. *arXiv preprint arXiv:2605.05726*, 2026.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In *ACL*, 2024.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP*, 2021.

Tiancheng Gu, Kaicheng Yang, Ziyong Feng, Xingjun Wang, Yanzhao Zhang, Dingkun Long, Yingda Chen, Weidong Cai, and Jiankang Deng. Breaking the modality barrier: Universal embedding learning with multimodal llms. In *ACM MM*, 2025.

Tiancheng Gu, Kaicheng Yang, Kaichen Zhang, Xiang An, Ziyong Feng, Yueyi Zhang, Weidong Cai, Jiankang Deng, and Lidong Bing. Unime-v2: Mllm-as-a-judge for universal multimodal embedding learning. In *AAAI*, 2026.

Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhu Chen. Vlm2vec: Training vision-language models for massive multimodal embedding tasks. In *ICLR*, 2025.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026.

Yikun Liu, Yajie Zhang, Jiayin Cai, Xiaolong Jiang, Yao Hu, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. Lamra: Large multimodal model as your advanced retrieval assistant. In *CVPR*, 2025.

Rui Meng, Ziyan Jiang, Ye Liu, Mingyi Su, Xinyi Yang, Yuepeng Fu, Can Qin, Raghuveer Thirukovalluru, Xuan Zhang, Zeyuan Chen, Ran Xu, Caiming Xiong, Yingbo Zhou, Wenhu Chen, and Semih Yavuz. VLM2vec-v2: Advancing multimodal embedding for videos, images, and visual documents. *Transactions on Machine Learning Research*, 2026.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR*, 2013.

OpenAI. Gpt-5.6. `https://openai.com/`, 2026.

OpenSearch-AI. Ops-mm-embedding-v1: Unified multimodal embedding model. *arXiv preprint arXiv*, 2025.

Krista Opsahl-Ong, Michael J. Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In *EMNLP*, 2024.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. In *NeurIPS*, 2024.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *ICLR*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023.

Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models. In *ACL*, 2025.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Qwen Team. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

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. *Transactions on Machine Learning Research*, 2024.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Yingxu Wang, Siwei Liu, Jinyuan Fang, and Zaiqiao Meng. EvoAgentX: An automated framework for evolving agentic workflows. In *EMNLP*, 2025.

Zifei Wang, Wei Wen, Qiang Ji, Keyu Chen, Ruizhi Qiao, and Xing Sun. Skill is not document: Query-conditioned compatibility for llm agent skill routing. *arXiv preprint arXiv:2606.03565*, 2026.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Shengming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *ICLR*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *ICLR*, 2025a.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

YanZhao Zheng, ZhenTao Zhang, Chao Ma, YuanQiang Yu, JiHuai Zhu, Yong Wu, Tianze Xu, Baohua Dong, Hangcheng Zhu, Ruohui Huang, et al. Skillrouter: Skill routing for llm agents at scale. *arXiv preprint arXiv:2603.22455*, 2026.

Yuanhang Zheng, Peng Li, Wei Liu, Yang Liu, Jian Luan, and Bin Wang. Toolrerank: Adaptive and hierarchy-aware reranking for tool retrieval. In *LREC-COLING*, 2024.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *ICLR*, 2024.

A Appendix

A.1 More Results

Results on the SkillRouter (Easy Setting). As shown in Table 4, MM-RouterEvolve achieves the best retrieval performance across all five metrics on the SkillRouter easy setting. Compared with SkillRet, MM-RouterEvolve improves Hit@1 from 0.593 to 0.691 and MRR@10 from 0.726 to 0.770. This gain also extends to R@10, which increases from 0.708 to 0.734. Together, these results show that MM-RouterEvolve ranks the target skill higher and retrieves it more reliably within the top-K results. A similar advantage is observed over SkillRouter-8B, despite its substantially larger 8B backbone: MM-RouterEvolve improves Hit@1 from 0.683 to 0.691 and MRR@10 from 0.745 to 0.770, while increasing R@50 from 0.837 to 0.853. This comparison suggests that iteratively refining query–skill representations with verified routing experience provides benefits beyond simply increasing the capacity of a static retriever.

Retrieve-and-Rerank Results on SkillRouter. Table 5 reports the retrieve-and-rerank results under the SkillRouter hard setting. For each pipeline, the embedding model first retrieves the top-K candidate skills, which are then reordered by the corresponding reranker, and all metrics are computed on the final ranked list. Compared with the strongest competing pipeline, SkillRouter-Emb-0.6B $\times$

Table 4: Comparison of skill retrieval performance on the SkillRouter easy setting. All methods are evaluated using text-only queries and skills.

Model	Params	Hit@1 ↑	MRR@10 ↑	R@10 ↑	R@20 ↑	R@50 ↑
VLM2Vec-V2 (Meng et al., 2026)	2B	0.487	0.559	0.564	0.668	0.733
UniME-V2 (Gu et al., 2026)	2B	0.527	0.624	0.628	0.660	0.781
Ops-MM-Embedding-v1 (OpenSearch-AI, 2025)	8B	0.633	0.724	0.677	0.759	0.807
Qwen3VL-Emb-2B (Li et al., 2026)	2B	0.620	0.693	0.666	0.783	0.826
Qwen3VL-Emb-8B (Li et al., 2026)	8B	0.633	0.709	0.718	0.785	0.830
Qwen3-Emb-0.6B Zhang et al. (2025b)	0.6B	0.607	0.688	0.652	0.744	0.798
Qwen3-Emb-8B Zhang et al. (2025b)	8B	0.653	0.730	0.720	0.771	0.836
SkillRet (Cho et al., 2026)	0.6B	0.593	0.726	0.708	0.772	0.837
R3-Skill (Wang et al., 2026)	0.6B	0.673	0.755	0.713	0.773	0.803
SkillRouter-0.6B (Zheng et al., 2026)	0.6B	0.672	0.720	0.688	0.742	0.801
SkillRouter-8B (Zheng et al., 2026)	8B	0.683	0.745	0.712	0.791	0.837
MM-RouterEvolve	2B	**0.691**	**0.770**	**0.734**	**0.802**	**0.853**

Table 5: Comparison of skill retrieve-and-rerank performance on the SkillRouter Hard benchmark. All methods are evaluated using text-only queries and skills.

Pipeline	Hit@1 ↑	MRR@10 ↑	R@10 ↑
Qwen3-Emb-0.6B × Qwen3-Rank-0.6B Zhang et al. (2025b)	0.625	0.685	0.671
Qwen3-Emb-8B × Qwen3-Rank-8B Zhang et al. (2025b)	0.683	0.745	0.702
Qwen3VL-Emb-2B × Qwen3VL-Rank-2B Li et al. (2026)	0.625	0.703	0.667
SkillRouter-Emb-0.6B × SkillRouter-Rank-0.6B Zheng et al. (2026)	0.715	0.765	0.710
R3-Skill-Emb-0.6B × R3-Skill-Rank-0.6B Wang et al. (2026)	0.640	0.714	0.661
MM-RouterEvolve	**0.721**	**0.767**	**0.723**

SkillRouter-Rank-0.6B, MM-RouterEvolve improves Hit@1 from 0.715 to 0.721 and R@10 from 0.710 to 0.723. Notably, MM-RouterEvolve also outperforms the substantially larger Qwen3-Emb-8B × Qwen3-Rank-8B pipeline across all metrics. These results show that improving the retrieval stage with experience-driven representation self-evolution also provides better candidate sets for reranking, leading to higher final retrieval accuracy.

Retrieve-and-Rerank Results on MM-SkillBench-SE. Table 6 reports the retrieve-and-rerank results on MM-SkillBench-SE. MM-RouterEvolve achieves the best performance across all three retrieval metrics, outperforming all compared pipelines. Compared with R3-Skill-Emb-0.6B × R3-Skill-Rank-0.6B, MM-RouterEvolve improves Hit@1 from 0.652 to 0.734 and MRR@10 from 0.708 to 0.792. Notably, MM-RouterEvolve also surpasses the larger Qwen3VL-Emb-8B × Qwen3VL-Rank-8B pipeline, improving Hit@1 from 0.671 to 0.734 and R@10 from 0.809 to 0.904. These results demonstrate that allowing query and skill representations to self-evolve within the retrieve-and-rerank pipeline yields greater gains in candidate discrimination than merely increasing the capacity of retrievers and rerankers.

A.2 More Discussions

Effect of Memory Relevance. We further investigate whether retrieval improvements arise merely from providing additional historical context or require experiences that are relevant to the current query or skill. In this discussion, MM-RouterEvolve is evaluated in only a single round to mitigate the effect of self-evolution across multiple executions. Replacing the query memory with randomly sampled experiences reduces Hit@1 from 0.594 to 0.571, while randomizing the skill memory similarly degrades performance to 0.582. Corrupting both memories simultaneously further decreases Hit@1 to 0.569 and R@10 to 0.801. In contrast, MM-RouterEvolve achieves 0.703 Hit@1 and 0.785 MRR@10 when using relevance-guided experience selection. These results therefore show that memory retrieval is beneficial only when the augmented experiences are semantically relevant to the current decision, whereas simply increasing context volume does not improve routing and can instead introduce distracting signals.

Effect of Multimodal Skill Representations. We next analyze how different components of the skill representation affect retrieval performance. Replacing raw skill sources with concise text summaries improves Hit@1 from 0.587 to 0.590 and MRR@10 from 0.704 to 0.726, indicating that concise summaries retain the key functional information required for retrieval while filtering out unnecessary implementation details. Adding the structured schema further increases Hit@1 to 0.633, showing that systematically organizing functional intents, inputs, outputs, and execution constraints

Table 6: Comparison of skill retrieve-and-rerank performance on MM-SkillBench-SE.

Pipeline	**Hit@1** ↑	**MRR@10** ↑	**R@10** ↑
Qwen3VL-Emb-2B × Qwen3VL-Rank-2B Li et al. (2026)	0.515	0.632	0.776
Qwen3VL-Emb-8B × Qwen3VL-Rank-8B Li et al. (2026)	0.671	0.729	0.809
R3-Skill-Emb-0.6B × R3-Skill-Rank-0.6B Wang et al. (2026)	0.652	0.708	0.806
MM-RouterEvolve	**0.734**	**0.792**	**0.904**

Table 7: Ablation study of memory relevance on MM-SkillBench-SE. Random Query Memory and Random Skill Memory replace the relevant experiences with randomly sampled experiences.

Memory Setting	**Hit@1** ↑	**MRR@10** ↑	**R@10** ↑	**R@20** ↑	**R@50** ↑
Static	0.594	0.668	0.818	0.854	0.882
Random Query Memory	0.571	0.646	0.806	0.854	0.872
Random Skill Memory	0.582	0.650	0.812	0.856	0.883
Random Query + Skill Memory	0.569	0.646	0.801	0.850	0.863
MM-RouterEvolve	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**

Table 8: Ablation study of skill representation components on MM-SkillBench-SE. We progressively transform the raw skill source into a concise text summary t_j, augment it with a structured schema z_j, and further incorporate a visual skill card v_j.

Skill Representation	**Hit@1** ↑	**MRR@10** ↑	**R@10** ↑	**R@20** ↑	**R@50** ↑
Raw Skill Sources	0.587	0.704	0.783	0.806	0.868
Text Summary (t_j)	0.590	0.726	0.809	0.829	0.899
Text Summary t_j + Schema z_j	0.633	0.745	0.828	0.844	0.906
Text Summary t_j + Schema z_j + Visual Card v_j	**0.703**	**0.785**	**0.896**	**0.915**	**0.959**

provides discriminative signals for distinguishing related skills. Finally, incorporating the visual skill card yields the strongest performance, improving Hit@1 from 0.633 to 0.703 and R@10 from 0.828 to 0.896. This improvement shows that the additional visual representation contributes complementary information beyond the textual summary and structured schema, supporting the use of multimodal skill profiles for fine-grained skill routing.

A.3 MM-SkillBench-SE Construction

Experience Annotation for Multimodal Skill Retrieval. Figure 4 presents an example instance in MM-SkillBench-SE. Unlike conventional retrieval settings that only match the current query with static skill representations, MM-SkillBench-SE additionally provides verified routing experiences collected from previous interactions. Each experience records the original query, the retrieved skill candidates, and the execution feedback.

Dataset Statistics. Figure 5 presents the statistics of MM-SkillBench-SE. As shown in Figure 5(a), the benchmark covers 13 functional categories, including document processing, video generation, image generation, social media, scheduling, database analysis, reasoning, coding, etc. Figure 5(b) illustrates the distribution of skill length. The skill descriptions exhibit an approximately normal distribution in length, with a median of 1,270 tokens, indicating that most skills contain rich information beyond short names or simple descriptions, including execution procedures, input-output specifications, and usage constraints. Figure 5(c) shows the query length distribution across the training and evaluation sets, where most queries are concise user instructions, thereby posing substantial challenges for skill retrieval.

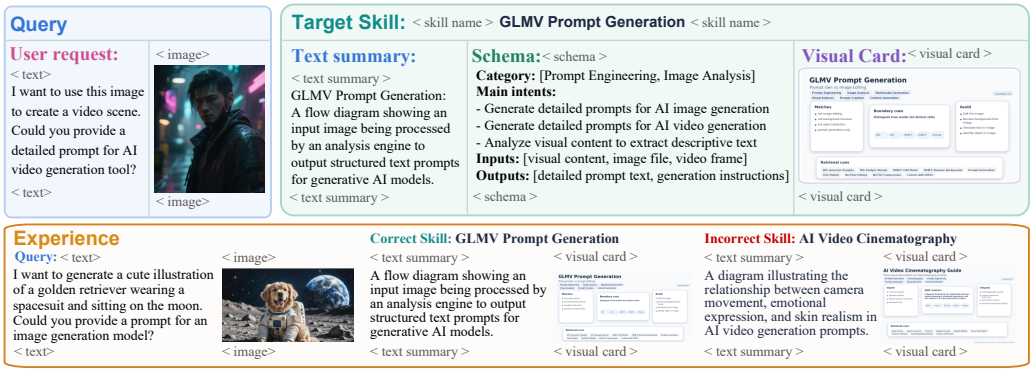

Figure 4: An example instance in our MM-SkillBench-SE.

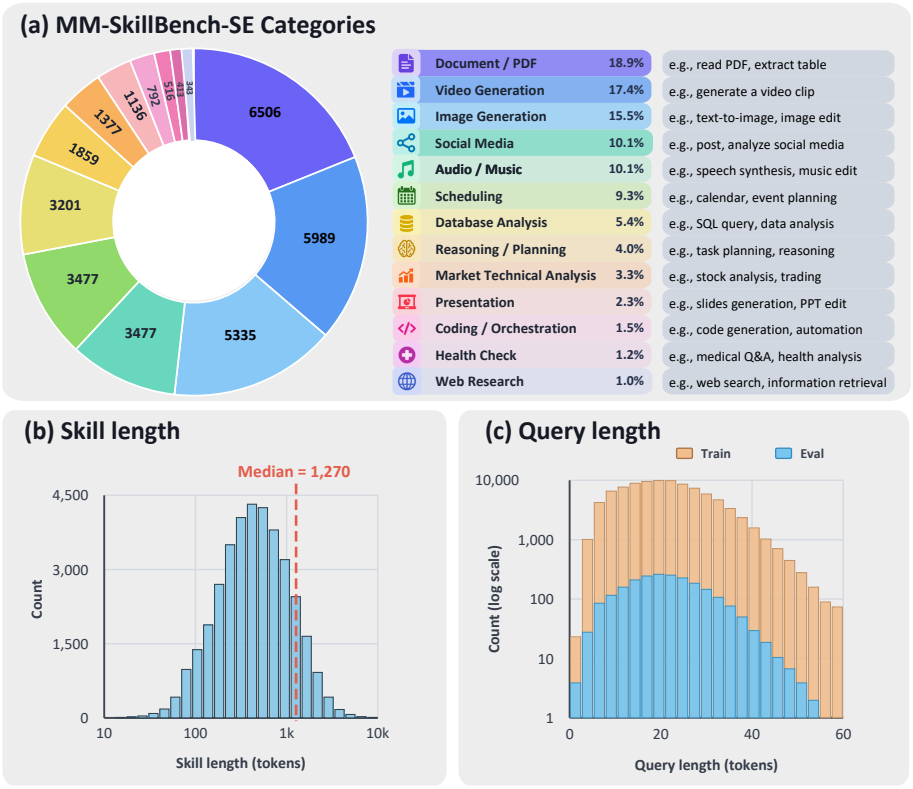

Figure 5: (a) Skill distribution across 13 functional categories. (b) Skill length distribution with the median token length marked. (c) Query length distributions of the training and evaluation sets.