# SKILLGRAPH: CO-EVOLVING SKILLS AND COLLABORATION WITH VISUAL FEEDBACK

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Visual Multi-Agent Systems (VMAS) extend vision-language models (VLMs) through collaborative reasoning, but adapting agent perception and communication to the visual demands of each input remains challenging. Existing approaches often treat visual evidence acquisition, communication topology, and experience refinement separately, limiting how evolving skills can shape subsequent team behavior. We introduce **SkillGraph**, a framework for *vision-driven self-adaptation* that couples these processes through visual information. Given an image and a question, SkillGraph retrieves reusable skills to instantiate agents. A Multimodal Graph Transformer (MMGT) combines skill representations with question and image features to compute agent-specific selective attention and visually grounded representations. The resulting attention maps define agent-specific image crops, which are provided to the corresponding agents alongside the original image. The visually grounded representations further predict a directed communication graph, allowing visual context to guide both evidence acquisition and collaboration. During training, failed executions are diagnosed to refine a multimodal Skill Bank, whose updated skill representations are fed back into MMGT to reshape subsequent visual attention and communication topology. Experiments across four benchmarks, five representative multi-agent structures, and four VLM backbones show that SkillGraph consistently improves performance while reducing inference-time token consumption. Controlled analyses further validate the contributions of visual conditioning, agent-specific visual inputs, and skill evolution.

## 1 INTRODUCTION

Visual Multi-Agent Systems (VMAS) combine the perceptual and reasoning capabilities of vision-language models (VLMs) through collaborative problem solving (Li et al., 2023; Yu et al., 2025b; Hong et al., 2024; Qian et al., 2024; Wu et al., 2024; Yu et al., 2026). To handle diverse visual tasks, these systems need to adapt how their agents acquire evidence and coordinate their reasoning. The image itself provides essential information for this adaptation: similar questions can involve small textual details, dispersed objects, or complex spatial relationships, each requiring a different organization of visual processing and collaboration. This raises a fundamental question: *How can visual information drive the self-adaptation of a multi-agent system?*

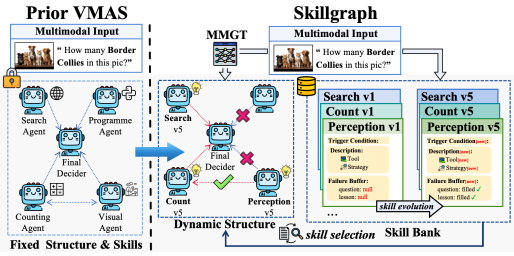



Figure 1: **Overview of SkillGraph.** Unlike prior VMAS with fixed communication and static skills, SkillGraph jointly adapts agent-specific visual attention and communication topology, while refining agent skills through an evolving Skill Bank.

Recent methods have made progress toward adaptive multimodal reasoning. DART (Sivakumaran et al., 2026) recruits tools based on disagreements among agents, while XSkill (Jiang et al., 2026a) accumulates visually grounded experiences and skills to improve subsequent agent execution. These methods demonstrate the value of adaptive tool use and experience accumulation. However, neither explicitly couples agent-specific visual evidence selection with an evolving team communication structure. This coupling matters because changes in an agent's strategy can alter the visual evidence it



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.

needs, while the distribution of that evidence can change which agents should exchange information. Supporting this interaction remains a central challenge for visual multi-agent adaptation.

Our key intuition is to use visual information as a shared basis for adapting both agent perception and team coordination. Each agent's visual focus should reflect its skills and the current image, while its collaboration with other agents should be grounded in the same visual context. Experience can then refine this process by updating the skills that guide both decisions. Following this principle, we propose **SkillGraph**, a framework for *vision-driven self-adaptation* that connects visual evidence acquisition, collaborative reasoning, and skill evolution.

Given an image and a question, SkillGraph retrieves reusable skills to instantiate agents. A Multimodal Graph Transformer (MMGT) combines skill representations with the question and image features to compute per-agent selective attention. The resulting attention maps determine agent-specific crops, which accompany the original image throughout subsequent agent and tool calls. The visually conditioned agent representations also determine a directed communication graph, allowing visual information to guide both evidence acquisition and collaboration. During training, multimodal diagnosis of failed executions refines the Skill Bank, and the updated skill representations condition subsequent attention and topology predictions. This feedback loop enables visual experience to shape future team behavior while keeping the underlying VLMs frozen. During evaluation, the Skill Bank and model parameters remain fixed, while visual attention and communication graphs are computed for each input.

Experiments across visual question-answering benchmarks and VLM backbones demonstrate the effectiveness of SkillGraph. Controlled analyses show that visual conditioning improves adaptation decisions, agent-specific crops improve evidence acquisition, and skill evolution further improves subsequent visual attention and collaboration. Together, these results support the value of connecting visual information to both task-specific configuration and experience-based refinement. Our main contributions are summarized as follows:

- We investigate vision-driven self-adaptation in visual multi-agent systems and show that adapting visual evidence acquisition and collaboration provides benefits beyond refining agent instructions alone.
- We propose SkillGraph, which couples skill-conditioned selective attention, persistent global and local visual inputs, and visually conditioned communication through a feedback loop of skill evolution.
- We provide controlled evaluations that distinguish the contributions of visual conditioning, agent-specific visual inputs, and experience-based refinement, alongside analyses of performance, computational cost, and robustness.

## 2 Related Work

**Visual Multi-Agent Systems and Communication Topology.** Collaboration graphs involve communication topology and agent capability. In Visual Multi-Agent Systems (VMAS) both are fixed before deployment: orchestrators dispatch perceivers and vision-expert tools (Wang et al., 2026b; Zhang et al., 2026b; Jiang et al., 2024), while document, chart and medical systems use predefined roles (Han et al., 2025; Yu et al., 2025a; Li et al., 2025; Feng et al., 2025). These systems remain vulnerable to error propagation (Yu et al., 2025b; Wu et al., 2025) and degradation with collaboration depth (Yu et al., 2026). VS-Bench further shows that strong perception does not ensure competent multi-agent behaviour (Xu et al., 2026b). Learned topology is almost entirely text-only, with methods optimizing fixed configurations offline (Zhuge et al., 2024; Hu et al., 2024), predicting query-adaptive graphs over fixed agent pools (Zhang et al., 2024a; Gao et al., 2025), or evolving agents and orchestrators (Xue et al., 2025; Dang et al., 2025), yet none of these predictors is informed by how each agent's own competence has changed. This leaves routing conditioned on text alone and disconnected from evolving skill states. SkillGraph closes both gaps by conditioning topology prediction on multimodal query content and feeding updated skill representations back into the predictor.

**Self-Improving Agents with Skill Libraries.** Skill libraries accumulate reusable experience as natural-language memories, executable code, or workflow templates (Zhao et al., 2024; Wang et al., 2023; Zhang et al., 2024b), supporting web navigation and computer control (Wang et al.,

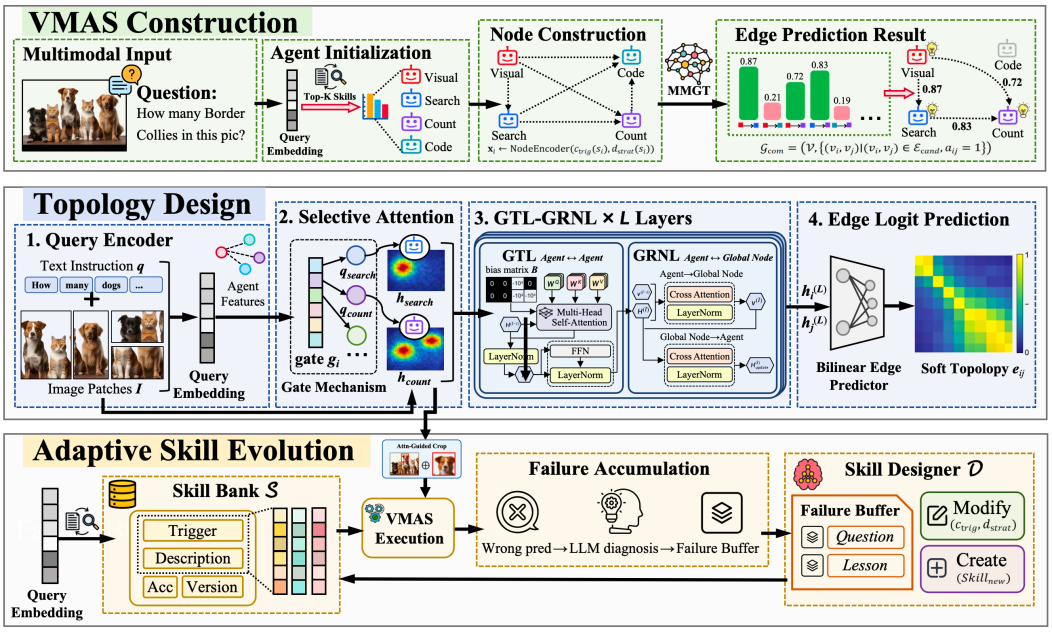



Figure 2: **Overview of SkillGraph.** Unlike prior VMAS with fixed communication and static skills, SkillGraph jointly computes agent-specific visual attention and communication topology from current skills and visual input, while refining agent skills through an evolving Skill Bank.

2025b; Kuroki et al., 2024). Recent studies extend retrieval toward dynamic skill injection and the joint evolution of skills and policies through reinforcement learning (Wang et al., 2025a; Xia et al., 2026; Alzubi et al., 2026). Other work examines skill organization, routing, and composition benchmarks (Liang et al., 2026; Zheng et al., 2026; Chen et al., 2026). Multimodal approaches further connect textual procedures with visual evidence and accumulated experience, extending skill reuse to visually grounded tasks (Jiang et al., 2026a; Zhang et al., 2026a; Xu et al., 2026a; Jiang et al., 2026b). Despite this progress, existing frameworks often treat skill repositories and collaboration structures as separate components, while making limited use of visual features. In SkillGraph, collaborative executions provide failure feedback to refine skills, whose updated representations subsequently condition visual attention and communication topology.

## 3 METHOD

Given an image-question pair $\mathcal{Q} = (q, \mathcal{I})$, a VMAS of $N$ agents collaborates to produce an answer $\hat{a}$. SkillGraph adapts both the agents' visual inputs and their communication graph to the current query. As shown in Figure 2, retrieved skills condition selective attention within MMGT, whose attention maps guide crop extraction and whose visually grounded agent representations support graph prediction, as detailed in Section 3.1. Agents then solve the query using the original image, their own crops, and messages from their predecessors. During training, multimodal failure feedback refines the Skill Bank as described in Section 3.2, while task rewards optimize MMGT through the co-evolution loop presented in Section 3.3. At inference, skills and model parameters remain fixed, while visual inputs and communication graphs are computed per query.

### 3.1 VISION-DRIVEN TEAM ADAPTATION

**Skill Retrieval and Agent Initialization.** SkillGraph instantiates agents $\mathcal{V} = \{v_1, \ldots, v_N\}$ with skills from a persistent Skill Bank $\mathcal{S}$. Each skill supplies a trigger condition $c_{\text{trig}}$ and a reasoning strategy $\rho_{\text{strat}}$, with its full structure detailed in Appendix A.1. A frozen $D$-dimensional sentence encoder maps the question to $\mathbf{q}_{\text{text}} \in \mathbb{R}^D$ and each skill to a node feature

$$\mathbf{x}_i \leftarrow \text{NodeEncoder}\big(c_{\text{trig}}(s_i),\, \rho_{\text{strat}}(s_i)\big) \in \mathbb{R}^D. \tag{1}$$

These skill embeddings are cached in $\mathbf{M} \in \mathbb{R}^{|\mathcal{S}| \times D}$. The top $N$ skills by cosine similarity to $\mathbf{q}_{\text{text}}$ are assigned to $v_1, \ldots, v_N$ in descending order of similarity. Active node features are refreshed when skills are retrieved or evolved. Both selective attention and topology prediction depend on $\mathbf{x}_i$, allowing skill revisions to influence an agent's visual focus and communication relationships.

#### 3.1.1 Skill-Conditioned Selective Attention

Different skills may require different evidence from the same image. MMGT therefore conditions each agent's attention on its skill representation and the shared multimodal task context. A frozen CLIP encoder yields $N_{\text{p}}$ patch tokens of width $D_{\text{img}}$, $\mathbf{Q}_{\text{img}} \in \mathbb{R}^{N_{\text{p}} \times D_{\text{img}}}$. A global task context is formed with a residual text shortcut that retains the question representation alongside the attended visual features:

$$\mathbf{r} = \mathbf{W}_{\text{r}}\,\text{LN}\Big(\text{CrossAttn}\big(\mathbf{W}_{\text{t}}\mathbf{q}_{\text{text}},\, \mathbf{Q}_{\text{img}}(\mathbf{W}^{r}_{\text{img}})^{\top}\big) + \mathbf{W}_{\text{t}}\mathbf{q}_{\text{text}}\Big) \in \mathbb{R}^{d}, \tag{2}$$

where $\text{CrossAttn}(\mathbf{u}, \mathbf{Z})$ is multi-head attention with query $\mathbf{u}$, keys and values $\mathbf{Z}$, and its own input and output projections, LN is LayerNorm, $d$ is the MMGT width, and every $\mathbf{W}$ is a learned projection. Each agent projects its skill feature to $\mathbf{h}_i^{\text{proj}} = \text{GELU}(\text{LN}(\mathbf{W}_{\text{node}}\mathbf{x}_i))$ and forms its own image query $\tilde{\mathbf{q}}_i = \mathbf{W}_{\text{sel}}\mathbf{h}_i^{\text{proj}} + \mathbf{g}_i \odot \mathbf{r}$, where the gate $\mathbf{g}_i = \sigma(\mathbf{W}_g[\mathbf{W}_{\text{sel}}\mathbf{h}_i^{\text{proj}} \,\|\, \mathbf{r}])$ decides how much task context to inject. The agent then attends over the patches, with a residual on its skill feature:

$$(\mathbf{z}_i, \boldsymbol{\alpha}_i) = \text{CrossAttn}(\tilde{\mathbf{q}}_i, \ldots), \qquad \mathbf{h}_i^{(0)} = \text{LN}(\mathbf{h}_i^{\text{proj}} + \mathbf{z}_i), \tag{3}$$

where $\sigma$ is the sigmoid, $\|$ is concatenation, $\odot$ is the elementwise product, $\mathbf{g}_i \in [0,1]^d$ gates how much of the global context $\mathbf{r}$ each agent injects into its own query, and $\boldsymbol{\alpha}_i \in \mathbb{R}^{N_{\text{p}}}$ is the head-averaged attention weight of agent $v_i$ over the patches; the residual on $\mathbf{h}_i^{\text{proj}}$ preserves the skill-derived feature. Stacking gives $\mathbf{H}^{(0)} \in \mathbb{R}^{N \times d}$, and the relay node is initialized as $\mathbf{r}^{(0)} \leftarrow \mathbf{r}$. The attention maps are exported for crop extraction, while $\mathbf{H}^{(0)}$ continues through MMGT for topology prediction, grounding both decisions in the same skill-conditioned visual information.

#### 3.1.2 Visually Conditioned Topology Prediction

Communication depends on how agents' skills and visual observations relate to one another. MMGT uses Graph Transformer Layers (GTL) to model relations between agent states and Global Relay Node Layers (GRNL) to share team-level context before predicting directed edges. Communication is restricted to a candidate edge set $\mathcal{E}_{\text{cand}} \subseteq \mathcal{V} \times \mathcal{V}$ fixed by a base topology specified in Section 4.1.

Each skill maps to a stage, *perceiver*, *reasoner*, or *verifier*, which new skills inherit via `parent_id`. A role-affinity mask $\mathbf{A}_{\text{role}} \in \{0,1\}^{N \times N}$ is rebuilt on every query: $[\mathbf{A}_{\text{role}}]_{ij} = 1$ marks that $v_j$ belongs to the stage feeding into $v_i$ along the cycle perceiver $\rightarrow$ reasoner $\rightarrow$ verifier $\rightarrow$ perceiver. Diagonal entries are 1, and all remaining entries are 0. This mask provides a soft stage prior for attention over agent representations. MMGT stacks $L$ alternating GTL and GRNL over $\mathbf{H}^{(0)}$. Each GTL applies self-attention with the role prior as an additive bias, written here for a single head,

$$\mathbf{H}' = \text{LN}\Big(\mathbf{H}^{(\ell-1)} + \text{softmax}\Big(\tfrac{\mathbf{H}^{(\ell-1)}\mathbf{W}_Q(\mathbf{H}^{(\ell-1)}\mathbf{W}_K)^{\top}}{\sqrt{d}} + \lambda\,\mathbf{A}_{\text{role}}\Big)\mathbf{H}^{(\ell-1)}\mathbf{W}_V\mathbf{W}_O\Big), \tag{4}$$

with a learned scalar $\lambda$, followed by a post-norm GELU feed-forward sublayer $\hat{\mathbf{H}}^{(\ell)} = \text{LN}(\mathbf{H}' + \text{FFN}(\mathbf{H}'))$. Each GRNL then updates the relay node from all agents and broadcasts it back:

$$\bar{\mathbf{r}}^{(\ell)} = \text{LN}\big(\mathbf{r}^{(\ell-1)} + \text{CrossAttn}(\mathbf{r}^{(\ell-1)}, \hat{\mathbf{H}}^{(\ell)})\big), \qquad \mathbf{r}^{(\ell)} = \text{LN}\big(\bar{\mathbf{r}}^{(\ell)} + \text{FFN}_{\text{r}}(\bar{\mathbf{r}}^{(\ell)})\big), \tag{5}$$

$$\mathbf{H}^{(\ell)} = \text{LN}\big(\hat{\mathbf{H}}^{(\ell)} + \mathbf{1}_N(\mathbf{W}_{\text{v2n}}\mathbf{r}^{(\ell)})^{\top}\big), \tag{6}$$

where $\mathbf{1}_N$ is the all-ones vector, and $\text{FFN}_{\text{r}}$ is a feed-forward sublayer analogous to FFN but dedicated to the relay node. Aggregation and broadcast allow edge decisions to incorporate team-level context. A direction-aware bilinear predictor with scalar bias $b_{\text{e}}$ models asymmetric information dependencies:

$$e_{ij} = (\mathbf{h}_i^{(L)})^{\top}\mathbf{W}_{\text{edge}}\,\mathbf{h}_j^{(L)} + b_{\text{e}}. \tag{7}$$

The logits are jointly min–max rescaled to $\tilde{e}_{ij} \in [-1, 1]$, yielding query-conditioned edge sampling probabilities $p_{ij} = \sigma(\tilde{e}_{ij}/\tau)$ at temperature $\tau$. Edges are then sampled sequentially from $\mathcal{E}_{\text{cand}}$, while choices that would introduce a cycle are excluded, yielding

$$\mathcal{G}_{\text{com}} = \big(\mathcal{V},\, \{(v_i, v_j) \in \mathcal{E}_{\text{cand}} \mid a_{ij} = 1\}\big). \tag{8}$$

GTL and GRNL operate on representations within MMGT, while $\mathcal{G}_{\text{com}}$ specifies actual inter-agent message dependencies and admits a topological execution order. Details of score normalization and DAG-constrained sampling are given in Appendix A.2.

#### 3.1.3 Attention-Guided Collaborative Execution

To make MMGT's visual focus available to the execution agents, we convert the attention maps $\boldsymbol{\alpha}_i$ from eq. (3) into auxiliary image crops. This turns skill-conditioned attention into additional visual views for each agent, without introducing an additional localization module or crop-specific training objective.

Let $\mathbf{A}_i \in \mathbb{R}^{\sqrt{N_{\text{p}}} \times \sqrt{N_{\text{p}}}}$ denote $\boldsymbol{\alpha}_i$ reshaped onto its original (square) patch grid. We transform the attention map into a spatial heatmap,

$$\mathbf{S}_i = \text{Up}\left(\text{Smooth}\left(\frac{\mathbf{A}_i - \mu(\mathbf{A}_i)}{\text{std}(\mathbf{A}_i) + \xi}\right)\right), \tag{9}$$

where $\mu(\cdot)$ and $\text{std}(\cdot)$ are taken over the grid, $\xi$ is a small constant for numerical stability, $\text{Smooth}(\cdot)$ denotes spatial smoothing, and $\text{Up}(\cdot)$ upsamples the map to the original image resolution. We then extract connected high-response regions from $\mathbf{S}_i$. For each retained component $\mathcal{C}_{i,j}$, we construct an image crop

$$\mathcal{I}_{i,j} = \text{Crop}(\mathcal{I},\, \text{Box}(\mathcal{C}_{i,j}; \epsilon)), \tag{10}$$

where $\text{Box}(\mathcal{C}_{i,j}; \epsilon)$ denotes the axis-aligned pixel-space bounding box of the component, expanded by a component-dependent margin to preserve nearby visual context. Depending on the spatial structure of the attention map, the procedure produces one local crop, two separated local crops, or a broad contextual view.

At every agent call for the query, $v_i$ receives both the original image $\mathcal{I}$ and its own crop(s) $\{\mathcal{I}_{i,j}\}$. The original image preserves global scene context, while the crops enlarge regions emphasized by the agent's attention. Keeping both available allows subsequent calls to revisit the same visual evidence. Agents execute in the topological order of $\mathcal{G}_{\text{com}}$, each conditioning on $q$, its skill $s_i$, these visual inputs, and its in-neighbours' outputs to produce $\hat{a}_i$. A skill-free decision node outside $\mathcal{V}$, shared by all methods, aggregates the agent outputs into the system answer $\hat{a}$ using the aggregation rule detailed in Appendix A.2.

### 3.2 Skill Evolution from Multimodal Failure Feedback

Execution failures provide feedback for refining existing skills. During training, a Skill Designer $\mathcal{D}$ uses diagnosed failures to update $\mathcal{S}$, so that accumulated experience can inform subsequent visual attention and collaboration. Each skill maintains a bounded failure buffer $\mathcal{F}$, a running accuracy $\pi$, and a version counter $\text{ver}$, as defined in Appendix A.1; the latter two support longitudinal analysis.

**Multimodal Failure Diagnosis.** A failure record is stored for every agent whose own answer is wrong, regardless of the system outcome. This ties each record to the agent's own prediction rather than assigning downstream errors to a correct trace. A multimodal diagnosis model receives the agent's trace, its answer $\hat{a}_i$, the ground truth $a^*$, the active skill, and the image together with the agent's crops, without which it could only inherit the agent's possibly wrong description of the scene. It returns a type in $\{\text{PERCEPTION}, \text{REASONING}, \text{AMBIGUOUS}\}$ with a lesson $\delta_i$, and $(q, \mathcal{I}, \hat{a}_i, a^*, \delta_i, \text{type})$ is appended to $\mathcal{F}(s_i)$; the reliability of this diagnosis step is evaluated in Appendix D.

**Failure-Driven Skill Updates.** Every $K$ iterations, $\mathcal{D}$ revisits hard skills $\mathcal{S}_{\text{hard}} = \{s \in \mathcal{S} \mid |\mathcal{F}(s)| \geq n_f\}$, accumulating multiple failures before proposing a revision. For each skill, it receives a failure-type histogram together with recent diagnostic lessons and detailed failure cases. Perception-dominated failures call for verification steps or a narrower trigger, whereas reasoning-dominated failures call for a constrained inference procedure. It then either *modifies* $c_{\text{trig}}$ and $\rho_{\text{strat}}$ in place, incrementing $\text{ver}$, or *creates* a skill $s_{\text{new}}$ for an uncovered sub-task. New skills become available for retrieval after the embedding cache is rebuilt. In both cases $\mathcal{F}(s) \leftarrow \emptyset$, so consumed failures cannot re-trigger the same proposal. The update rules are detailed in Appendix A.3.

### 3.3 Co-Evolution Loop and Optimization

SkillGraph connects skill revision and topology optimization through a training-time feedback loop. In $\mathcal{S} \rightarrow \mathrm{MMGT}$, updated skills are re-encoded through eq. (1), which can change attention through eq. (3), crop extraction through eq. (10), and edge prediction through eq. (7), even with fixed MMGT parameters. In $\mathrm{MMGT} \rightarrow \mathcal{S}$, the predicted graph determines which predecessor messages each instantiated agent receives, shaping the execution traces and failures used for subsequent skill updates.

MMGT parameters $\Theta$ are optimized from task rewards while the VLM backbones and encoders remain frozen. Since the graph construction described in eq. (8) involves discrete sampling, we use REINFORCE over mini-batches of size $B$:

$$\nabla_\Theta \mathcal{L}_{\mathrm{MMGT}} \approx -\frac{1}{B} \sum_{b=1}^{B} u^{(b)} \nabla_\Theta \log P_\Theta\big(\mathcal{G}_{\mathrm{com}}^{(b)} \mid \mathcal{Q}^{(b)}\big), \tag{11}$$

$$\log P_\Theta\big(\mathcal{G}_{\mathrm{com}}^{(b)} \mid \mathcal{Q}^{(b)}\big) = \sum_{(i,j)\in\mathcal{E}_{\mathrm{samp}}^{(b)}} \big[a_{ij}^{(b)} \log p_{ij}^{(b)} + (1 - a_{ij}^{(b)}) \log(1 - p_{ij}^{(b)})\big], \tag{12}$$

where $u^{(b)} = u(\hat{a}^{(b)}) \in \{0, 1\}$ is the detached correctness of the system answer and $\mathcal{E}_{\mathrm{samp}}^{(b)} \subseteq \mathcal{E}_{\mathrm{cand}}$ contains only sampled edge decisions. Topology-policy updates reshape the shared attention module through edge prediction, and each forward pass regenerates agent-specific visual views from the current attention maps. These updates alternate with periodic skill revisions, as detailed in Appendix A.4.

At inference, both $\Theta$ and $\mathcal{S}$ are frozen, and $\mathcal{D}$ is not invoked. Selective attention, crops, and the communication graph are computed for each new query, so the team adapts its visual inputs and collaboration through input conditioning without further learning.

## 4 Experiments

### 4.1 Settings

**Baselines and Benchmarks.** We compare SkillGraph with a single-agent baseline and multi-agent baselines using text-based communication under five fixed topologies. We primarily evaluate Qwen3-VL-8B (Bai et al., 2025a), with LLaVA-OneVision-Qwen2-7B (Li et al., 2024), InternVL3-8B (Zhu et al., 2025), and Qwen2.5-VL-7B-Instruct (Bai et al., 2025b) included for backbone generalization. We evaluate on MMBench (Liu et al., 2024), MathVista (Lu et al., 2023), RealWorldQA, and InfoVQA (Mathew et al., 2022), reporting accuracy for the first three and ANLS for InfoVQA. Further details on the baselines, backbones, and benchmarks are provided in Appendix B.

### 4.2 Main Results

**Performance Improvements to VMAS.** As presented in Table 1, SkillGraph improves performance across all four primary benchmarks, demonstrating its effectiveness across diverse multimodal tasks, including general perception, visual reasoning, document understanding, and real-world scene comprehension. Furthermore, as shown in Table 2, SkillGraph improves performance across diverse vision-language backbones, including LLaVA-OneVision and InternVL3, demonstrating the general applicability of our topology-conditioned co-evolutionary design across different model architectures. Results on Qwen2.5-VL-7B-Instruct are provided in Appendix C.

**Effect of Model Scale.** Table 3 evaluates SkillGraph on backbones ranging from 2B to 38B parameters. SkillGraph consistently improves performance across benchmarks and model scales, demonstrating its effectiveness across backbone capacities. The gains generally narrow as model scale increases, suggesting diminishing headroom for collaboration with stronger backbones. Despite this trend, SkillGraph continues to yield gains at larger scales.

### 4.3 Ablation Study

**Ablation of Components.** We ablate Skill Evolution, MMGT-based topology adaptation, and Crop from the full SkillGraph; the topology ablation fixes communication while retaining attention-guided

Table 1: Performance of multi-agent graph topology structures on Qwen3-VL-8B-Instruct and Qwen3-VL-8B-Thinking. Each topology baseline is compared with our proposed SkillGraph enhancement.

| Model | Method | MMBench | MathVista | RealWorldQA | InfoVQA | Average |
|---|---|---|---|---|---|---|
| Qwen3-VL-8B Instruct | DirectAnswer | 84.2 | 77.3 | 71.4 | 83.2 | 79.0 |
| | Linear | 83.7 | 79.7 | 73.4 | 82.7 | 79.9 |
| | + SkillGraph | **86.0** +2.3 | **80.9** +1.2 | **74.8** +1.4 | **84.5** +1.8 | **81.6** +1.7 |
| | Layered | 84.6 | 80.2 | 73.9 | 82.9 | 80.4 |
| | + SkillGraph | **86.6** +2.0 | **81.7** +1.5 | **75.7** +1.8 | **85.0** +2.1 | **82.3** +1.9 |
| | Centralized | 84.3 | 80.5 | 73.2 | 82.8 | 80.2 |
| | + SkillGraph | **86.4** +2.1 | **82.3** +1.8 | **75.1** +1.9 | **84.2** +1.4 | **82.0** +1.8 |
| | Random | 85.2 | 81.6 | 75.3 | 84.9 | 81.8 |
| | + SkillGraph | **86.9** +1.7 | **82.9** +1.3 | **76.3** +1.0 | **85.8** +0.9 | **83.0** +1.2 |
| | Complete | 84.9 | 81.8 | 75.2 | 85.0 | 81.7 |
| | + SkillGraph | **86.9** +2.0 | **83.3** +1.5 | **76.6** +1.4 | **85.6** +0.6 | **83.1** +1.4 |
| Qwen3-VL-8B Thinking | DirectAnswer | 85.1 | 81.2 | 73.7 | 84.8 | 81.2 |
| | Linear | 83.0 | 81.7 | 73.2 | 84.7 | 80.7 |
| | + SkillGraph | **86.1** +3.1 | **83.2** +1.5 | **74.3** +1.1 | **85.6** +0.9 | **82.3** +1.6 |
| | Layered | 84.9 | 82.1 | 73.6 | 85.3 | 81.5 |
| | + SkillGraph | **86.9** +2.0 | **84.0** +1.9 | **75.9** +2.3 | **86.5** +1.2 | **83.3** +1.8 |
| | Centralized | 84.1 | 81.6 | 73.3 | 84.9 | 81.0 |
| | + SkillGraph | **86.5** +2.4 | **83.8** +2.2 | **75.9** +2.6 | **86.3** +1.4 | **83.1** +2.1 |
| | Random | 84.8 | 84.0 | 75.9 | 86.0 | 82.7 |
| | + SkillGraph | **87.6** +2.8 | **85.4** +1.4 | **77.4** +1.5 | **86.7** +0.7 | **84.3** +1.6 |
| | Complete | 84.2 | 83.8 | 75.4 | 85.8 | 82.3 |
| | + SkillGraph | **87.5** +3.3 | **85.8** +2.0 | **78.2** +2.8 | **87.3** +1.5 | **84.7** +2.4 |

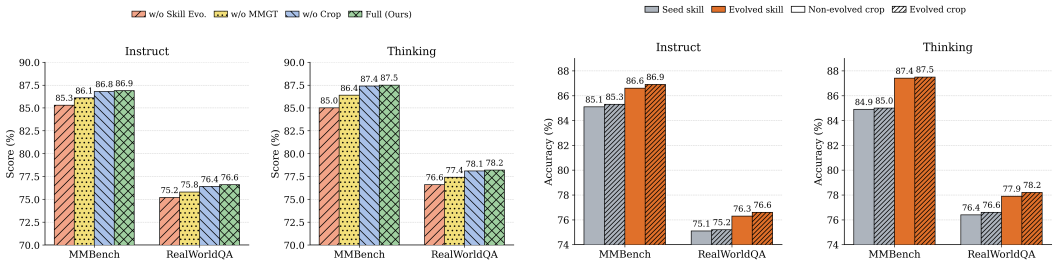
Table 2: Performance comparison across additional vision-language backbones. Improvements brought by SkillGraph are highlighted.

| Model | Method | MMBench | MathVista | RealWorldQA | InfoVQA | Average |
|---|---|---|---|---|---|---|
| InternVL3-8B | DirectAnswer | 82.2 | 71.5 | 70.7 | 77.1 | 75.4 |
| | Linear | 81.7 | 73.4 | 71.6 | 76.6 | 75.8 |
| | + SkillGraph | **84.2** +2.5 | **76.0** +2.6 | **73.9** +2.3 | **78.7** +2.1 | **78.2** +2.4 |
| | Layered | 82.5 | 74.1 | 72.2 | 76.8 | 76.4 |
| | + SkillGraph | **84.7** +2.2 | **76.6** +2.5 | **73.9** +1.7 | **79.4** +2.6 | **78.7** +2.3 |
| | Centralized | 82.1 | 73.8 | 71.5 | 76.8 | 76.1 |
| | + SkillGraph | **84.5** +2.4 | **76.6** +2.8 | **73.7** +2.2 | **78.7** +1.9 | **78.4** +2.3 |
| | Random | 82.9 | 75.2 | 73.1 | 78.5 | 77.4 |
| | + SkillGraph | **85.0** +2.1 | **77.6** +2.4 | **74.5** +1.4 | **79.5** +1.0 | **79.2** +1.8 |
| | Complete | 82.8 | 75.3 | 72.9 | 78.7 | 77.4 |
| | + SkillGraph | **85.2** +2.4 | **78.0** +2.7 | **75.2** +2.3 | **80.6** +1.9 | **79.8** +2.4 |
| LLaVA-OV-Qwen2-7B | DirectAnswer | 81.6 | 64.8 | 67.2 | 71.3 | 71.2 |
| | Linear | 81.4 | 66.5 | 68.1 | 71.1 | 71.8 |
| | + SkillGraph | **83.6** +2.2 | **68.9** +2.4 | **70.1** +2.0 | **73.4** +2.3 | **74.0** +2.2 |
| | Layered | 82.1 | 67.2 | 68.5 | 71.5 | 72.3 |
| | + SkillGraph | **83.7** +1.6 | **69.3** +2.1 | **70.4** +1.9 | **73.0** +1.5 | **74.1** +1.8 |
| | Centralized | 81.7 | 67.0 | 67.9 | 71.3 | 72.0 |
| | + SkillGraph | **84.1** +2.4 | **69.6** +2.6 | **70.4** +2.5 | **73.0** +1.7 | **74.3** +2.3 |
| | Random | 82.5 | 68.1 | 69.4 | 73.0 | 73.3 |
| | + SkillGraph | **84.2** +1.7 | **70.4** +2.3 | **70.6** +1.2 | **74.1** +1.1 | **74.8** +1.5 |
| | Complete | 82.4 | 68.4 | 69.2 | 72.7 | 73.2 |
| | + SkillGraph | **84.3** +1.9 | **70.9** +2.5 | **70.9** +1.7 | **73.2** +0.5 | **74.8** +1.6 |

visual views. As shown in Fig. 3a, all three components contribute positively across the evaluated settings, and the full model achieves the best overall performance. Skill Evolution strengthens agent-role specialization and cross-query knowledge reuse, while MMGT enables query-specific, vision-aware communication. Crop supplies agent-specific local evidence. The factorial analysis in Fig. 3b shows that attention-guided visual views provide additional gains across skill states and tasks. In several settings, these gains are larger with evolved than with seed skills, indicating an interaction between skill state and the utility of agent-specific visual inputs. Together, these results show

Table 3: Performance of SkillGraph across Qwen3-VL and InternVL3 models of varying scales under the Complete topology. Gray rows show results after applying SkillGraph to each backbone.

| Model | MMBench | MathVista | RealWorldQA | InfoVQA | Average |
|---|---|---|---|---|---|
| Qwen3-VL-4B-Instruct | 83.9 | 76.8 | 72.1 | 81.8 | 78.7 |
| + SkillGraph | **86.8** +2.9 | **80.0** +3.2 | **74.9** +2.8 | **83.0** +1.2 | **81.2** +2.5 |
| Qwen3-VL-8B-Instruct | 84.9 | 81.8 | 75.2 | 85.0 | 81.7 |
| + SkillGraph | **86.9** +2.0 | **83.3** +1.5 | **76.6** +1.4 | **85.6** +0.6 | **83.1** +1.4 |
| Qwen3-VL-32B-Instruct | 88.6 | 84.2 | 80.1 | 87.6 | 85.1 |
| + SkillGraph | **89.3** +0.7 | **85.5** +1.3 | **81.3** +1.2 | **87.8** +0.2 | **86.0** +0.9 |
| InternVL3-2B | 80.2 | 59.4 | 65.7 | 66.8 | 68.0 |
| + SkillGraph | **83.8** +3.6 | **63.1** +3.7 | **69.2** +3.5 | **70.2** +3.4 | **71.6** +3.6 |
| InternVL3-8B | 82.8 | 75.3 | 72.9 | 78.7 | 77.4 |
| + SkillGraph | **85.2** +2.4 | **78.0** +2.7 | **75.2** +2.3 | **80.6** +1.9 | **79.8** +2.4 |
| InternVL3-38B | 88.9 | 76.5 | 77.1 | 85.7 | 82.1 |
| + SkillGraph | **89.8** +0.9 | **77.9** +1.4 | **78.4** +1.3 | **86.3** +0.6 | **83.1** +1.0 |




(a) Component ablation.

(b) Factorial analysis of Skill and Crop evolution.

Figure 3: **Ablation analysis of SkillGraph.** Left: effects of removing Skill Evolution, MMGT and Crop. Right: interaction between seed/evolved skill states and non-evolved/evolved crops.

complementary contributions within SkillGraph's feedback loop: evolved skills update the capability representations consumed by MMGT, MMGT jointly computes visual attention and collaboration decisions from these representations and the query, and the resulting agent-specific attention maps make selected visual evidence available during agent execution.

**Effect of Iteration Number.** Figure 4 reports SkillGraph performance under different iteration counts. Performance improves as training proceeds across the four benchmarks. The Thinking setting consistently outperforms Instruct, and MathVista shows notable improvements, suggesting that SkillGraph can be beneficial for tasks requiring complex visual reasoning.

**Analysis of Skill Bank Evolution.** Across all benchmarks, the frequency of evolution actions (*Create* and *Modify*) decreases monotonically as training progresses, indicating that the Skill Bank stabilizes rather than expanding without bound; final bank size further varies with task complexity, with the reasoning-heavy MathVista developing a substantially larger skill inventory than the perception-dominated InfoVQA. Detailed convergence curves and a longitudinal case study of a single skill's evolution are provided in Appendix G.

**Inference Cost.** SkillGraph also lowers the VLM token cost of multi-agent inference. We compare it with the Random topology under Qwen3-VL-8B-Instruct, the strongest fixed-topology baseline in Table 1. Token counts include the image crops attached to each agent call. Across all benchmarks, SkillGraph consistently reduces per-query token usage, with reductions ranging from 14.2% to 22.4% and an average reduction of 18.7% as shown in Table 4.

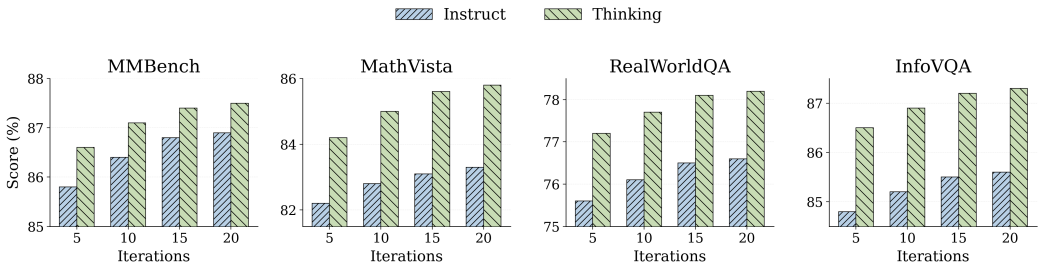



Figure 4: **Performance of SkillGraph across different iteration numbers.**

Table 4: Per-query token usage under Qwen3-VL-8B-Instruct.

| Benchmark | Random | SkillGraph | Reduc. |
|---|---|---|---|
| MMBench | 2,669 | 2,070 | 22.4% |
| MathVista | 3,187 | 2,734 | 14.2% |
| RealWorldQA | 3,002 | 2,453 | 18.3% |
| InfoVQA | 3,166 | 2,516 | 20.5% |
| Average | 3,006 | 2,443 | 18.7% |

Table 5: VLM Meta-Agent comparison under Qwen3-VL-8B-Instruct.

| Benchmark | Meta-Agent | SkillGraph | Δ |
|---|---|---|---|
| MMBench | 85.2 | 86.9 | +1.7 |
| MathVista | 82.0 | 83.3 | +1.3 |
| RealWorldQA | 75.5 | 76.6 | +1.1 |
| InfoVQA | 84.9 | 85.6 | +0.7 |
| Average | 81.9 | 83.1 | +1.2 |

**Comparison with VLM Orchestration.** Beyond fixed topologies, we compare SkillGraph with a query-adaptive VLM Meta-Agent built through context engineering. Given the image, question, and active skill descriptions, the Meta-Agent selects participating specialists and determines their execution order, followed by coordinator-based aggregation. Under the Complete base topology, SkillGraph consistently outperforms this VLM orchestration approach across all four benchmarks, with an average improvement of 1.2 points as shown in Table 5. While the Meta-Agent performs workflow planning through specialist selection and sequencing, MMGT learns query-conditioned pairwise directed dependencies among specialists from multimodal capability states, enabling parallel collaboration branches, direct specialist communication, and richer evidence propagation.

### 4.4 Case Study

Figure 5 shows two queries where the complete-graph baseline fails. In the first, it focuses on the visible stop sign on the right and concludes that only vehicles from that direction must stop. In the second, it treats the wider diameter of the right-hand candle as decisive and selects the wrong candle. SkillGraph instead retrieves query-relevant skills and predicts a query-conditioned collaboration topology, enabling more appropriate visual inspection and comparison. As a result, it correctly identifies the four-way stop and the middle candle as the largest.

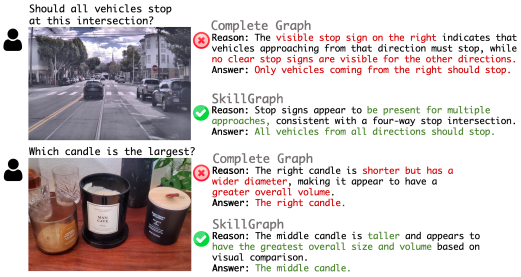



Figure 5: Qualitative comparison between the complete-graph baseline and SkillGraph.

## 5 Conclusion

We introduce SkillGraph, a framework that breaks frozen coordination in multimodal multi-agent systems by jointly adapting agent skills and communication topology. Through a vision-grounded graph transformer and a self-evolving Skill Bank, SkillGraph enables query-specific collaboration that evolves with agent expertise. Extensive experiments across diverse benchmarks demonstrate consistent improvements over existing multi-agent baselines, showing the promise of adaptive communication for scalable multimodal intelligence.

## AI USE STATEMENT

In this work, we used generative AI tools for improving the readability and grammar of the manuscript. We did not use generative AI tools for research design, algorithm development, experimental implementation, data analysis, or the generation of scientific claims. All AI-assisted edits were reviewed and verified by the authors.

## REFERENCES

Salaheddin Alzubi, Noah Provenzano, Jaydon Bingham, Weiyuan Chen, and Tu Vu. Evoskill: Automated skill discovery for multi-agent systems. *arXiv preprint arXiv:2603.02766*, 2026.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

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

Yufan Dang, Chen Qian, Xueheng Luo, Jingru Fan, Zihao Xie, Ruijie Shi, Weize Chen, Cheng Yang, Xiaoyin Che, Ye Tian, et al. Multi-agent collaboration via evolving orchestration. *arXiv preprint arXiv:2505.19591*, 2025.

Qianhan Feng, Zhongzhen Huang, Yakun Zhu, Xiaofan Zhang, and Qi Dou. Ucagents: Unidirectional convergence for visual evidence anchored multi-agent medical decision-making. *arXiv preprint arXiv:2512.02485*, 2025.

Hongcheng Gao, Yue Liu, Yufei He, Longxu Dou, Chao Du, Zhijie Deng, Bryan Hooi, Min Lin, and Tianyu Pang. Flowreasoner: Reinforcing query-level meta-agents. *arXiv preprint arXiv:2504.15257*, 2025.

Siwei Han, Peng Xia, Ruiyi Zhang, Tong Sun, Yun Li, Hongtu Zhu, and Huaxiu Yao. Mdocagent: A multi-modal multi-agent framework for document understanding. *arXiv preprint arXiv:2503.13964*, 2025.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Steven Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, volume 2024, pp. 23247–23275, 2024.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024.

Bowen Jiang, Zhijun Zhuang, Shreyas S Shivakumar, Dan Roth, and Camillo J Taylor. Multi-agent vqa: Exploring multi-agent foundation models in zero-shot visual question answering. *arXiv preprint arXiv:2403.14783*, 2024.

Guanyu Jiang, Zhaochen Su, Xiaoye Qu, and Yi R Fung. Xskill: Continual learning from experience and skills in multimodal agents. *arXiv preprint arXiv:2603.12056*, 2026a.

Ziyan Jiang, Li An, Yujian Liu, Jiabao Ji, Qiucheng Wu, Jacob Andreas, Yang Zhang, and Shiyu Chang. Visualskill: Multimodal skills for computer-use agents. *arXiv preprint arXiv:2606.18448*, 2026b.

So Kuroki, Taishi Nakamura, Takuya Akiba, and Yujin Tang. Agent skill acquisition for large language models via cycleqd. *arXiv preprint arXiv:2410.14735*, 2024.

Bingxuan Li, Yiwei Wang, Jiuxiang Gu, Kai-Wei Chang, and Nanyun Peng. Metal: A multi-agent framework for chart generation with test-time scaling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30054–30069, 2025.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in neural information processing systems*, 36:51991–52008, 2023.

Yuan Liang, Ruobin Zhong, Haoming Xu, Chen Jiang, Yi Zhong, Runnan Fang, Jia-Chen Gu, Shumin Deng, Yunzhi Yao, Mengru Wang, et al. Skillnet: Create, evaluate, and connect ai skills. *arXiv preprint arXiv:2603.04448*, 2026.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pp. 216–233. Springer, 2024.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1697–1706, 2022.

Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, et al. Scaling large language model-based multi-agent collaboration. *arXiv preprint arXiv:2406.07155*, 2024.

Yang Shi, Yifeng Xie, Minzhe Guo, Liangsi Lu, Mingxuan Huang, Jingchao Wang, Zhihong Zhu, Boyan Xu, and Zhiqi Huang. Mmerror: A benchmark for erroneous reasoning in vision-language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44973–44986, 2026.

Nithin Sivakumaran, Justin Chen, David Wan, Yue Zhang, Jaehong Yoon, Elias Stengel-Eskin, and Mohit Bansal. Dart: Leveraging multi-agent disagreement for tool recruitment in multimodal reasoning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5445–5464, 2026.

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. *arXiv preprint arXiv:2305.16291*, 2023.

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

Siting Wang, Minnan Pei, Luoyang Sun, Cheng Deng, Yuchen Li, Kun Shao, Zheng Tian, Haifeng Zhang, and Jun Wang. Spatialviz-bench: A cognitively-grounded benchmark for diagnosing spatial visualization in mllms. In *International Conference on Learning Representations*, 2026a.

Zeqing Wang, Wentao Wan, Qiqing Lao, Runmeng Chen, Minjie Lang, Xiao Wang, Feng Gao, Keze Wang, and Liang Lin. Towards top-down reasoning: An explainable multi-agent approach for visual question answering. *IEEE Transactions on Multimedia*, 2026b.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks. *arXiv preprint arXiv:2504.06821*, 2025b.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First conference on language modeling*, 2024.

Yutong Wu, Jie Zhang, Yiming Li, Chao Zhang, Qing Guo, Nils Lukas, and Tianwei Zhang. Cowpox: Towards the immunity of vlm-based multi-agent systems. 2025.

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

Binxiao Xu, Ruichuan An, Bocheng Zou, and Hang Hua. Agent skills should go beyond text: The case for visual skills. *arXiv preprint arXiv:2606.01414*, 2026a.

Zelai Xu, Zhexuan Xu, Xiangmin Yi, Huining Yuan, Mo Guang, Kaiwen Long, Xinlei Chen, Yi Wu, Chao Yu, and Yu Wang. Vs-bench: Evaluating vlms for strategic abilities in multi-agent environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21561–21572, 2026b.

Xiangyuan Xue, Yifan Zhou, Guibin Zhang, Zaibin Zhang, Yijiang Li, Chen Zhang, Zhenfei Yin, Philip Torr, Wanli Ouyang, and Lei Bai. Comas: Co-evolving multi-agent systems via interaction rewards. *arXiv preprint arXiv:2510.08529*, 2025.

Xinlei Yu, Chengming Xu, Zhangquan Chen, Yudong Zhang, Shilin Lu, Cheng Yang, Jiangning Zhang, Shuicheng Yan, and Xiaobin Hu. Visual document understanding and reasoning: A multi-agent collaboration framework with agent-wise adaptive test-time scaling. *arXiv preprint arXiv:2508.03404*, 2025a.

Xinlei Yu, Chengming Xu, Guibin Zhang, Yongbo He, Zhangquan Chen, Zhucun Xue, Jiangning Zhang, Yue Liao, Xiaobin Hu, Yu-Gang Jiang, et al. Visual multi-agent system: Mitigating hallucination snowballing via visual flow. *arXiv preprint arXiv:2509.21789*, 2025b.

Xinlei Yu, Chengming Xu, Zhangquan Chen, Bo Yin, Cheng Yang, Yongbo He, Yihao Hu, Jiangning Zhang, Cheng Tan, Xiaobin Hu, et al. Dual latent memory for visual multi-agent system. *arXiv preprint arXiv:2602.00471*, 2026.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*, 2024a.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024b.

Kangning Zhang, Shuai Shao, Qingyao Li, Jianghao Lin, Lingyue Fu, Shijian Wang, Wenxiang Jiao, Yuan Lu, Weiwen Liu, Weinan Zhang, et al. Mmskills: Towards multimodal skills for general visual agents. *arXiv preprint arXiv:2605.13527*, 2026a.

Zhehao Zhang, Ryan A Rossi, Tong Yu, Franck Dernoncourt, Ruiyi Zhang, Jiuxiang Gu, Sungchul Kim, Xiang Chen, Zichao Wang, and Nedim Lipka. Vipact: Visual-perception enhancement via specialized vlm agent collaboration and tool-use. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 36536–36546, 2026b.

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

YanZhao Zheng, ZhenTao Zhang, Chao Ma, YuanQiang Yu, JiHuan Zhu, Baohua Dong, and Hangcheng Zhu. Skillrouter: Retrieve-and-rerank skill selection for llm agents at scale. *arXiv preprint arXiv:2603.22455*, 2026.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*, 2024.

# A Supplementary Materials for Method

## A.1 Skill Structure and Node Feature Details

**Skill tuple definition.** A skill $s \in \mathcal{S}$ is a structured five-tuple $s = (c_{\text{trig}},\ \rho_{\text{strat}},\ \pi,\ \mathcal{F},\ \text{ver})$, where each field serves a distinct role in the co-evolution loop:

- $c_{\text{trig}}$: a natural-language description of the visual sub-task for which the skill is suited, used as the retrieval key during semantic matching.
- $\rho_{\text{strat}}$: step-by-step reasoning instructions that guide the agent's inference process on the matched sub-task.
- $\pi = n_{\text{succ}}/n_{\text{use}}$: a running accuracy estimate updated after each query on the *system-level* outcome, so that it tracks the same signal as the reward in eq. (11) rather than a separate per-agent criterion. It is written only during training and records the long-term effectiveness of a skill across versions, as reported in the case study of Section G.
- $\mathcal{F}$: a bounded failure buffer storing structured records $\big(q,\ \mathcal{I},\ \hat{a}_i,\ a^*,\ \delta_i,\ \text{type}\big)$, where $q$ is the query text, $\mathcal{I}$ is the input image, $\hat{a}_i$ is the agent's own incorrect prediction, $a^*$ is the ground-truth answer, $\delta_i$ is an LLM-generated diagnostic lesson summarising the failure cause, and $\text{type} \in \{\text{PERCEPTION},\ \text{REASONING},\ \text{AMBIGUOUS}\}$ is its failure category. The buffer retains the most recent records up to a fixed capacity, which is set larger than the evolution threshold $n_f$ so that the failure-type histogram is estimated from more than the minimum evidence. A skill becomes *hard* once $|\mathcal{F}(s)| \geq n_f$, and the buffer is cleared whenever the Skill Designer consumes it, so $|\mathcal{F}(s)|$ always counts failures accumulated since that skill was last revised. This structured representation enables the Skill Designer to perform evidence-based revisions rather than unconstrained free-form rewrites: each revision is grounded in concrete failure instances rather than abstract performance statistics.
- ver: a version counter incremented on every in-place modification, used to track the evolution history of a skill across training. The cumulative accuracy $\pi$ is *retained* across version increments so that long-term performance trends remain visible, while $\mathcal{F}$ is reset to $\emptyset$ after each modification to collect fresh diagnostic evidence under the updated specification.

**Functional-stage resolution.** Functional stage is not a stored element of the skill tuple. At graph construction, it is derived from the skill's resolved role through a fixed role-stage table. A newly created skill resolves through its `parent_id` to inherit the corresponding role mapping; unresolved skills receive the unconstrained fallback prior described in Appendix A.2.

**Node feature matrix.** The full agent feature matrix is $\mathbf{X}_{\text{agent}} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$, where $D$ is the sentence-encoder output dimension. Because both $c_{\text{trig}}$ and $\rho_{\text{strat}}$ are plain text, re-encoding after a skill update requires a single forward pass through the frozen sentence encoder and incurs negligible overhead regardless of system scale. No gradient flows through this re-encoding step; the updated embeddings are injected directly as new node features into the subsequent MMGT forward pass.

## A.2 MMGT Design Details

**Residual text shortcut in the query encoder.** When fusing visual patches and question text via cross-attention alone, high-entropy image features tend to dominate the output, suppressing question semantics in the fused representation. The residual term $\mathbf{W}_{\text{t}}\mathbf{q}_{\text{text}}$ in eq. (2) counteracts this by ensuring the question embedding is always additively present, regardless of image complexity.

**Per-agent gate behavior.** The gate $\mathbf{g}_i \in [0,1]^d$ in eq. (3) controls, per feature dimension, how much of the global task context $\mathbf{r}$ each agent injects into its own image query $\tilde{\mathbf{q}}_i$, allowing agents

to specialize their visual attention without explicit supervision. The residual connection on $\mathbf{h}_i^{\mathrm{proj}}$ in the same equation preserves the original skill-derived features while additively enriching them with role-relevant visual context, ensuring that agent-specific reasoning priors are not overwritten by the attended image representation.

**Connected-region extraction.** Given the heatmap $\mathbf{S}_i$ from eq. (9), we binarize it at a threshold set between its mean and maximum response and extract four-connected components from the resulting mask. Each component is scored by its response mass above the threshold and by how its peak and mass compare to the strongest component in the map; components that are both sufficiently peaked and sufficiently massive are retained, and if none qualifies we retain the component containing the global maximum, so a crop is always produced. A single retained component yields one local crop. When two components are retained, we compare the distance between their mass-weighted centroids, normalized by image size, against a fixed separation threshold: components far apart yield two separate crops, while nearby ones are merged into a single crop that covers both. When more than two substantial components are retained, we treat the attention as spatially distributed rather than pick an arbitrary local region, and fall back to broad contextual views, using overlapping full-width bands for portrait images and overlapping quadrant tiles otherwise. In all cases the bounding box in eq. (10) is expanded by a margin that scales with the component's own size and with how dominant it is relative to other retained components, clipped to a bounded range rather than a fixed fraction of the image, so small precise regions and large diffuse ones are not padded by the same amount; an optional local refinement may adjust each expanded boundary toward a nearby visual edge, while never moving it inside the attention component.

**Attention-guided views without additional training.** Attention maps are produced by the shared MMGT representation used for topology prediction. Skill revisions and topology-policy updates reshape these maps across successive updates, and each forward pass regenerates the corresponding agent-specific visual views. Crop construction is an execution-time readout that introduces no additional localization model or crop-specific objective. The original image remains available to preserve global context.

**Global relay node as non-bottleneck aggregator.** A virtual node that only broadcasts to agents acts as a passive bottleneck whose representation never updates to reflect the evolving agent state. The GRNL prevents this by making the upward direction an attention over all $N$ agent tokens: eq. (5) first aggregates the collective agent state into $\mathbf{r}^{(\ell)}$, and only then does eq. (6) broadcast the enriched context back, so the relay node accumulates progressively richer collaborative state across layers.

**Edge-score normalization.** Joint min–max normalization preserves the relative priority of candidate communication links for the current query before sampling. Accordingly, the resulting $p_{ij}$ values are used to construct a query-specific communication graph, rather than as cross-query calibrated confidence scores. The candidate set specifies the links available under a collaboration template, while the sampled DAG specifies which available links are activated for the current input.

**Sequential candidate-edge processing and DAG enforcement.** The candidate edge set $\mathcal{E}_{\mathrm{cand}}$ is constructed from the chosen base topology and specifies the communication links that may be activated. Within $\mathcal{E}_{\mathrm{cand}}$, edges are processed in a fixed row-major order. For each edge $(v_i, v_j)$, a DFS reachability check from $v_j$ over the previously committed edges determines whether adding $(v_i, v_j)$ would close a cycle. If so, the edge is skipped; otherwise, $a_{ij} \sim \mathrm{Bernoulli}(p_{ij})$ is sampled and committed to the partial graph. Cycle-forced exclusions carry no randomness and are therefore omitted from the log-probability in eq. (12). This construction guarantees that $\mathcal{G}_{\mathrm{com}}$ is a DAG and admits a well-defined topological execution order.

**Decision node.** After all agents in $\mathcal{V}$ have executed, a skill-free decision node reads the outputs $\{\hat{a}_i\}_{i=1}^{N}$ of all agents and produces the system answer $\hat{a}$ (the FinalRefer rule). The same decision node is used for SkillGraph and for every baseline topology.

### A.3 Adaptive Skill Evolution Details

**Failure-feedback protocol and information access.** Skill evolution uses two auxiliary modules during training: a Failure Diagnosis Module and a Skill Designer. Both modules are disabled with respect to Skill Bank updates during held-out evaluation: evaluation examples and their labels are not used to generate failure diagnoses, revise skills, or trigger skill evolution.

For a training instance on which an individual agent produces a parsable answer that differs from the dataset-provided ground-truth answer, the Failure Diagnosis Module receives the original task input, including the question, answer choices, and the original image when available; the agent's full response and predicted answer; and the description of the active skill. It additionally receives the ground-truth answer as an outcome-feedback signal. The module outputs one operational failure type, PERCEPTION, REASONING, or AMBIGUOUS, together with a short textual failure lesson.

The Failure Diagnosis Module does not generate, modify, or select the final answer for the current task. Ground-truth answers are used only during training to identify incorrect agent predictions, diagnose the associated failures, and support subsequent skill revision. They are not provided to the execution agents or MMGT when solving the current task, and they are not used to update the Skill Bank during held-out evaluation.

The Skill Designer receives the current skill specification and its associated diagnosed training failures, including the failure lessons, failure types, task inputs, agent predictions, and ground-truth answers. It may revise the skill's trigger condition and reasoning strategy, or create a new skill when the diagnosed failures reveal a recurring capability gap not covered by the current Skill Bank. Thus, the gains from skill evolution should be interpreted as gains from training-time, ground-truth-feedback-driven skill evolution.

The reliability of training-time failure attribution is evaluated independently in Appendix D; the human annotations and stronger-model diagnoses used in that audit are not available to the training or held-out evaluation pipeline.

**Modify action.** When $\mathcal{D}$ revises a hard skill in place, the cumulative running accuracy $\pi$ is retained across the version increment so that long-term performance trends remain visible. The failure buffer is reset to $\mathcal{F}(s) \leftarrow \emptyset$ to collect fresh diagnostic evidence under the improved specification, preventing stale failures from triggering a redundant second revision.

**Create action.** A newly created skill $s_{\text{new}}$ enters the retrieval pool as soon as the embedding cache is rebuilt, competing with existing skills on semantic similarity alone. It can therefore be exercised from the very next query onwards. Its accuracy estimate $\pi$ starts at zero and accumulates as queries are routed to it, recording the skill's subsequent effectiveness. The parent skill's failure buffer is cleared at the same time: those failures have been handed off to $s_{\text{new}}$, and leaving them in place would keep the parent above $n_f$ and re-trigger the identical proposal at every subsequent evolution period.

**Co-evolution.** The Skill Bank $\mathcal{S}$ and MMGT are connected through the feedback loop in Section 3.3. When any skill in $\mathcal{S}$ is modified or created, the embedding cache $\mathbf{M}$ is invalidated and reconstructed on the next forward pass. The node features in eq. (1) and their projections $\mathbf{h}_i^{\text{proj}}$ derive from the current skill text. Revisions can therefore affect the attention map computed by eq. (3), the crops extracted using eq. (10), and the edge predictions given by eq. (7), with MMGT parameters held fixed. MMGT serves as the shared adaptation module in this loop: the current skill representations and multimodal query jointly determine agent-specific visual attention and communication scores. The attention maps supply the auxiliary visual views used during execution, while the communication scores determine how information is exchanged among the active agents. The role prior $\mathbf{A}_{\text{role}}$ is rebuilt from the resolved role stages of the active skills; a newly created skill inherits its parent role mapping through `parent_id`. The prior changes only when the active stage configuration changes. The predicted topology determines which predecessor messages each agent receives during execution, shaping the traces and failures available for subsequent skill revision. The attention-guided crops serve as execution-time visual inputs in this process. This closes the feedback loop between skill content, team configuration, and failure-driven skill revision.

### A.4 Training Algorithm

At each MMGT update, let $\bar{\mathcal{C}}(\mathcal{Q})$ denote the agent-specific crops generated from the current attention maps and treated as fixed for differentiation. The topology-policy surrogate optimized by eq. (11) is

$$\mathcal{J}_{\text{topo}}(\Theta;\bar{\mathcal{C}}) = \mathbb{E}_{\mathcal{Q}}\,\mathbb{E}_{\mathcal{G}_{\text{com}}\sim P_\Theta(\cdot\mid\mathcal{Q})}\big[u\big(\hat{a}\big(\mathcal{G}_{\text{com}},\mathcal{Q},\bar{\mathcal{C}}(\mathcal{Q})\big)\big)\big]\,. \tag{13}$$

Here, $\bar{\mathcal{C}}(\mathcal{Q})$ is the stop-gradient execution context produced from the current attention maps, $\Theta$ denotes all MMGT parameters, and $u(\cdot)\in\{0,1\}$ is the binary correctness utility. The log-probability of a sampled graph is given by eq. (12); only that term produces gradients, which flow through the edge probabilities $p_{ij}^{(b)}$ and edge logits $e_{ij}^{(b)}$ in eq. (7) back to $\Theta$, while $u^{(b)}$ is detached from the computation graph. For fixed $\bar{\mathcal{C}}$,

$$\nabla_\Theta \mathcal{J}_{\text{topo}}(\Theta;\bar{\mathcal{C}}) = \mathbb{E}[u(\hat{a})\nabla_\Theta \log P_\Theta(\mathcal{G}_{\text{com}}\mid\mathcal{Q})]\,. \tag{14}$$

Thus, eq. (11) is the standard score-function estimator for the sampled topology policy under the current visual views. The views are fixed within each gradient update and regenerated from the current attention maps on subsequent forward passes after MMGT or Skill Bank updates. The update provides gradient-based optimization of the lightweight MMGT controller, while the VLM backbones and encoders are not fine-tuned. Skill Bank revisions follow the diagnosed training failures described in Appendix A.3. The complete training procedure is summarized in Algorithm 1.

**Algorithm 1** SkillGraph Training

**Require:** Agent pool $\mathcal{V}$, candidate edge set $\mathcal{E}_{\text{cand}}$, MMGT parameters $\Theta$, Skill Bank $\mathcal{S}$, Skill Designer $\mathcal{D}$, diagnosis LLM, dataset $\mathcal{T}$, $\text{Adam}(\Theta,\eta)$, total iterations $T$, batch size $B$, evolve period $K$
**Ensure:** Optimized MMGT parameters $\Theta$; evolved Skill Bank $\mathcal{S}$

Initialize $\Theta$, $\mathcal{S}$, and embedding cache $\mathbf{M}$
**for** $t = 1,\dots,T$ **do**
  Sample mini-batch $\{(q_b,\mathcal{I}_b,a_b^*)\}_{b=1}^{B}$ from $\mathcal{T}$
  **for** $b = 1,\dots,B$ **in parallel do** *// all instances share* $\Theta$
    Retrieve top-$N$ skills from $\mathbf{M}$ by cosine similarity; assign to $v_1,\dots,v_N$ by rank
    Build $\mathbf{X}_{\text{agent}}$ via eq. (1); build $\mathbf{A}_{\text{role}}$ from the stages of the active skills
    $\{p_{ij}^{(b)}\}$, $\{\boldsymbol{\alpha}_i^{(b)}\} \leftarrow \text{MMGT}(\mathbf{X}_{\text{agent}},\mathbf{A}_{\text{role}},\mathbf{q}_{\text{text}},\mathbf{Q}_{\text{img}})$ *// eq. (2)–eq. (7)*
    Form agent crops $\{\mathcal{I}_{i,j}^{(b)}\}$ from $\boldsymbol{\alpha}_i^{(b)}$ via eqs. (9) and (10)
    Process $\mathcal{E}_{\text{cand}}$ in row-major order; skip an edge if it would close a cycle (DFS check), else sample $a_{ij}^{(b)} \sim \text{Bernoulli}(p_{ij}^{(b)})$ and commit
    Obtain $\mathcal{G}_{\text{com}}^{(b)}$ via eq. (8); execute agents in topological order, each on $\big(\mathcal{I}_b,\{\mathcal{I}_{i,j}^{(b)}\}\big)$, giving $\hat{a}_{b,i}$
    Obtain the system answer $\hat{a}^{(b)}$ from the decision node; $u^{(b)} \leftarrow \mathbf{1}[\hat{a}^{(b)} = a_b^*]$
    **for** each agent $v_i$ with $\hat{a}_{b,i} \neq a_b^*$ **do**
      $(\delta_{b,i},\text{type}_{b,i}) \leftarrow \text{Diagnose}\big(\text{trace}_{b,i},\hat{a}_{b,i},a_b^*,s_i,\mathcal{I}_b,\{\mathcal{I}_{i,j}^{(b)}\}\big)$
      Append $\big(q_b,\mathcal{I}_b,\hat{a}_{b,i},a_b^*,\delta_{b,i},\text{type}_{b,i}\big)$ to $\mathcal{F}(s_i)$
    **end for**
  **end for**
  Compute $\mathcal{L}_{\text{MMGT}}$ via eqs. (11) and (12); $\Theta \leftarrow \text{Adam}\big(\Theta,\nabla_\Theta\mathcal{L}_{\text{MMGT}}\big)$
  **if** $t \bmod K = 0$ **then** *// skill evolution*
    $\mathcal{S} \leftarrow \mathcal{D}.\text{evolve}(\mathcal{S}_{\text{hard}})$ *// modify or create; clear consumed* $\mathcal{F}$
    Invalidate $\mathbf{M}$; rebuild on the next query
  **end if**
**end for**

## B Experimental Configuration

**Evaluation of the full SkillGraph framework.** For each conventional collaboration template, the "+ SkillGraph" condition uses the Skill Bank evolved on its training subset together with skill-conditioned visual attention, attention-guided auxiliary views, and MMGT-based query-adaptive

collaboration. The corresponding baseline retains the same agent pool and collaboration template. These paired evaluations measure the contribution of visual-feedback adaptation within each collaboration regime.

**Baselines.** We evaluate SkillGraph against both single-agent and multi-agent baselines to comprehensively assess its effectiveness. The single-agent baseline performs standard autoregressive decoding without any inter-agent communication. For multi-agent settings, we instantiate five widely adopted fixed-topology structures over the same agent pool, each using conventional text-based information passing and the same decision node as SkillGraph:

- **Linear** arranges agents in a sequential chain, where each agent receives the output of its immediate predecessor, forming a directed path graph.
- **Layered** partitions agents into two layers with feed-forward edges from every node in the first layer to every node in the second, forming a directed complete bipartite graph and enabling parallel processing within each layer.
- **Centralized** designates one hub agent whose output is broadcast to all other agents, following a star-out topology.
- **Random** constructs a random directed acyclic topology at graph initialization and keeps it fixed across all queries, providing a stochastic but query-independent topology baseline.
- **Complete** imposes a fixed ordering on the agents and connects every earlier agent to every later agent, forming a complete directed acyclic graph that maximizes feed-forward information flow at the cost of quadratic communication overhead.

Our primary experiments are conducted using Qwen3-VL-8B (Bai et al., 2025a) under both Instruct and Thinking inference modes. To validate structural generalizability across backbone families, we additionally evaluate on LLaVA-OneVision-Qwen2-7B (Li et al., 2024), which pairs a SigLIP vision encoder with a Qwen2 language model of 28 decoder layers; Qwen2.5-VL-7B-Instruct (Bai et al., 2025b), which uses a window-attention ViT with an MLP-based patch merger and extends multimodal rotary position embedding (MRoPE) to absolute time; and InternVL3-8B (Zhu et al., 2025), which follows the ViT–MLP–LLM paradigm and adopts native multimodal pre-training together with Variable Visual Position Encoding (V2PE). For model-scale analysis, we further extend experiments to Qwen3-VL-4B, Qwen3-VL-32B, InternVL3-2B, and InternVL3-38B.

**Benchmarks.** We evaluate on four complementary multimodal benchmarks spanning perception, compositional reasoning, and document understanding:

- **MMBench** (Liu et al., 2024) contains structured multiple-choice questions covering more than twenty ability dimensions organized in a three-level hierarchy.
- **MathVista** (Lu et al., 2023) requires fine-grained mathematical reasoning over diverse visual contexts, including geometry diagrams, function plots, and data charts.
- **RealWorldQA** focuses on practical scene understanding in authentic, unconstrained photographs, emphasizing spatial relations and physical reasoning that cannot be resolved from text alone.
- **InfoVQA** (Mathew et al., 2022) targets infographic comprehension, demanding precise text grounding, layout-aware parsing, and numerical reading from densely typeset documents. Following the standard protocol, we report Average Normalized Levenshtein Similarity (ANLS).

**Implementations.** Experiments are conducted on four NVIDIA A100 (80 GB) GPUs. Open-source VLM backbones are served via vLLM through an OpenAI-compatible endpoint. The image and text encoders are CLIP ViT-B/32 and all-MiniLM-L6-v2, both frozen. The number of agents is set to $N{=}4$, the initial edge probability is 0.5, the failure count threshold is $n_f{=}4$, and the skill evolution period is $K{=}1$.

Table 6: **Results on Qwen2.5-VL-7B-Instruct.** SkillGraph is evaluated under the same fixed-topology baselines as the main experiments.

| Method | MMBench | MathVista | RealWorldQA | InfoVQA | Average |
|---|---|---|---|---|---|
| DirectAnswer | 83.9 | 67.5 | 68.6 | 82.9 | 75.7 |
| Linear | 83.6 | 68.9 | 69.3 | 82.5 | 76.1 |
| + SkillGraph | **85.4** (+1.8) | **71.2** (+2.3) | **70.9** (+1.6) | **84.4** (+1.9) | **78.0** (+1.9) |
| Layered | 84.2 | 69.4 | 69.7 | 82.8 | 76.5 |
| + SkillGraph | **85.6** (+1.4) | **72.0** (+2.6) | **71.7** (+2.0) | **84.4** (+1.6) | **78.4** (+1.9) |
| Centralized | 83.8 | 69.5 | 69.2 | 82.6 | 76.3 |
| + SkillGraph | **85.8** (+2.0) | **71.4** (+1.9) | **71.3** (+2.1) | **84.5** (+1.9) | **78.3** (+2.0) |
| Random | 84.5 | 70.2 | 70.4 | 84.1 | 77.3 |
| + SkillGraph | **85.9** (+1.4) | **73.2** (+3.0) | **72.4** (+2.0) | **84.8** (+0.7) | **79.1** (+1.8) |
| Complete | 84.6 | 70.4 | 70.3 | 84.2 | 77.4 |
| + SkillGraph | **86.0** (+1.4) | **72.9** (+2.5) | **72.4** (+2.1) | **84.7** (+0.5) | **79.0** (+1.6) |

**Training Configuration.** SkillGraph does not fine-tune any vision or language backbone component. Gradient-based optimization updates only the lightweight MMGT parameters via REINFORCE. Separately, the textual Skill Bank is revised from diagnosed training failures by the Skill Designer; revised skills are re-encoded and serve as updated inputs to subsequent MMGT forward passes. MMGT is optimized with Adam at a learning rate of $\eta = 10^{-3}$ for $T$=20 training iterations.

## C Results on Additional Backbones

Table 6 reports results on Qwen2.5-VL-7B-Instruct under the same protocol as Table 2.

## D Reliability of Failure Diagnosis

Skill evolution relies on failure attribution that is sufficiently reliable to produce useful corrective lessons. In the main training pipeline, the Failure Diagnosis Module uses the same VLM backbone and inference configuration as the execution agents; no separately provisioned stronger teacher model is used to generate training-time failure lessons or to update the Skill Bank.

We evaluate the reliability of this training-time self-diagnosis mechanism in a separate offline audit. For each sampled failed trajectory, annotators assign a failure category based on the task input, the agent response, and the ground-truth answer. These human annotations are used only as an evaluation reference: they are not provided to the Failure Diagnosis Module, Skill Designer, MMGT, or execution agents, and they are never used for skill updates, model selection, or held-out evaluation.

To contextualize the reliability of self-diagnosis, we additionally run a stronger multimodal VLM under the same diagnosis task and compare its predictions against the same human annotations. This stronger model is an audit-only comparator rather than a component of SkillGraph's training or inference pipeline. In particular, its diagnoses are not used to generate failure lessons, revise skills, or select skills at evaluation time.

For this reliability study we further subdivide the three operational failure types of Section A.1 into six fine-grained categories, which map onto them as follows: VISUAL_MISREAD and VISUAL_OMISSION correspond to PERCEPTION; REASONING_ERROR and KNOWLEDGE_GAP correspond to REASONING; and INSTRUCTION_MISUNDERSTANDING and AMBIGUOUS_QUESTION correspond to AMBIGUOUS. This finer taxonomy is used only for the human-agreement evaluation below and does not change the failure buffer schema of Section A.1: the diagnosis LLM used during training still emits one of the three operational types described in Section 3.2, while teacher and human annotators for this study are instead asked to choose among the six fine-grained categories below, which gives a stricter test of diagnostic consistency than agreement on the coarser operational types alone.

We report both agreement accuracy and Cohen's kappa:

$$\kappa = \frac{p_o - p_e}{1 - p_e},$$

Table 7: **Reliability of failure diagnosis.** Acc. denotes agreement accuracy and $\kappa$ denotes Cohen's kappa.

| Dataset | Qwen3-VL-8B | | Qwen3-VL-32B | |
|---|---|---|---|---|
| | Acc. | $\kappa$ | Acc. | $\kappa$ |
| MMBench | 0.86 | 0.75 | 0.90 | 0.82 |
| MathVista | 0.88 | 0.78 | 0.92 | 0.85 |
| RealWorldQA | 0.82 | 0.69 | 0.86 | 0.76 |
| InfoVQA | 0.85 | 0.74 | 0.90 | 0.81 |
| Average | **0.853** | **0.74** | **0.895** | **0.81** |

Table 8: **Cross-dataset transfer of evolved skills.** Each row uses a Skill Bank evolved only on the indicated source benchmark and evaluates it on different target benchmarks. Diagonal entries are omitted because they correspond to in-domain rather than cross-dataset transfer.

| Training Source | MMBench | MathVista | RealWorldQA | InfoVQA |
|---|---|---|---|---|
| Base (no transferred skills) | 84.9 | 81.8 | 75.2 | 85.0 |
| MMBench | – | 82.4 | 76.0 | 85.2 |
| MathVista | 85.6 | – | 75.7 | 85.1 |
| RealWorldQA | 85.4 | 82.1 | – | 85.1 |
| InfoVQA | 85.2 | 81.9 | 75.4 | – |

where $p_o$ denotes observed agreement and $p_e$ denotes chance agreement.

Our diagnosis module achieves an average $\kappa = 0.74$ under Qwen3-VL-8B and $\kappa = 0.81$ under Qwen3-VL-32B, improving monotonically with backbone capability. As a reference point, MMErroR (Shi et al., 2026) reports $\kappa = 0.796$ from a human pilot study conducted during dataset construction, used there as the validation criterion for data reliability; SpatialViz-Bench (Wang et al., 2026a) reports a macro-averaged $\kappa = 0.885$ across fine-grained error categories using a binary-decomposition protocol. Both of these reference values are measured under conditions that ease agreement relative to our setting: they assess either synthetically constructed errors with a known ground-truth cause, or per-category binary decisions rather than a single six-way forced choice over naturally occurring, cross-domain model failures. Despite diagnosing open-ended failures across four heterogeneous benchmarks, our agreement remains within the range reported by these recent multimodal error-diagnosis and annotation reliability studies.

## E Cross-Dataset Skill Transfer

We evaluate whether skills evolved on one benchmark remain useful when transferred to a different multimodal task. For each transfer setting, the source Skill Bank is frozen and directly reused on the target benchmark, while MMGT computes target-specific visual attention, agent crops, and communication graphs from each target input. We exclude same-benchmark evaluation from this analysis and report only cross-dataset transfer.

The transferred Skill Banks consistently improve over the no-transfer baseline despite being evolved on different source distributions. Transfer is stronger between benchmarks that share underlying visual or reasoning requirements. In particular, MMBench skills transfer effectively to RealWorldQA, while reasoning-oriented skills evolved on MathVista provide larger gains on MMBench than on the document-centric InfoVQA. Conversely, InfoVQA yields smaller cross-dataset gains, consistent with its more specialized emphasis on text localization and layout understanding.

These results suggest that SkillGraph transfers reusable strategies rather than benchmark-specific solutions, while the magnitude of transfer depends on the compatibility between source and target task demands.

Table 9: **Training-time adaptation cost.** Results are measured using the two-A100-80GB profiling setup.

| Benchmark | Training tokens | Wall-clock time |
|---|---|---|
| MMBench | $5.69 \times 10^5$ | 3.07 h |
| MathVista | $7.11 \times 10^5$ | 4.79 h |
| RealWorldQA | $6.55 \times 10^5$ | 3.88 h |
| InfoVQA | $6.82 \times 10^5$ | 5.89 h |
| Total | $2.62 \times 10^6$ | 17.63 h |

## F Computational Cost

We report the computational cost of training-time adaptation. For each benchmark, MMGT optimization and Skill Bank evolution use 200 training cases that are disjoint from the benchmark test set used for inference-time evaluation. Failure diagnosis and ground-truth feedback are used only during training. At evaluation, the learned MMGT and evolved Skill Bank remain fixed, while agent-specific attention, auxiliary views, and communication graphs are inferred for each test query.

Under the two-A100-80GB profiling setup, one GPU serves the frozen Qwen3-VL-8B backbone through vLLM, while the other runs CLIP ViT-B/32, all-MiniLM-L6-v2, and MMGT. As summarized in Table 9, training across the four benchmarks generates approximately $2.62 \times 10^6$ tokens and requires 17.63 hours of wall-clock time in total. Most of this cost comes from multi-agent VLM rollouts used to obtain task rewards, whereas training the lightweight MMGT module itself introduces relatively limited computation.

During inference, the Qwen3-VL-8B vLLM process reserves approximately 68–72 GiB on the VLM-serving GPU, including model weights, runtime workspace, and the preallocated KV-cache pool. CLIP ViT-B/32, all-MiniLM-L6-v2, and MMGT together occupy approximately 1–2 GiB on the auxiliary GPU.

## G Detailed Analysis of Skill Bank Evolution

Figure 6 further characterizes how the Skill Bank evolves across benchmarks. Two patterns stand out. First, evolution actions (*Create* + *Modify*) decrease monotonically across iterations on every benchmark, indicating that the Skill Bank converges rather than expanding without bound. Second, the final bank size varies by task complexity: MathVista requires the largest inventory, reflecting the breadth of its mathematical sub-tasks (geometry, algebra, chart reading, etc.), whereas InfoVQA saturates earliest. This trend is consistent with the intuition that InfoVQA is an OCR-heavy document understanding benchmark that places stronger emphasis on text grounding and layout-aware perception than on diverse multi-step collaborative reasoning. Once these perception and grounding capabilities are established, the self-diagnosis loop encounters fewer novel failure modes, causing both skill creation and modification to taper off more rapidly.

To provide a more granular perspective on how these autonomous refinements translate into performance gains, Table 10 presents a qualitative case study tracing the evolution of a specific skill across versions. The initial seed strategy relies heavily on commonsense priors, making it vulnerable to visually misleading or atypical scenarios. Prompted by self-diagnosed failure cases, the framework iteratively refines both the trigger condition ($c_{\text{trig}}$) and the strategy description ($\rho_{\text{strat}}$). By version v5–v7, the skill has evolved from making naive stereotypic assumptions to enforcing a rigorous, evidence-based hypothesis-testing protocol. This qualitative shift illustrates the changes accompanying the quantitative improvements observed across iterations in Figure 4.

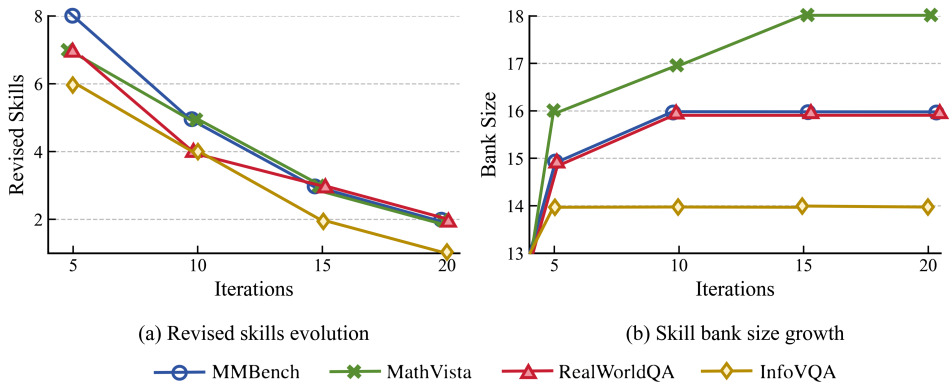



Figure 6: **Evolution of the Skill Bank across iterations.** The graph illustrates the frequency of evolution actions and the stabilization of the skill inventory across different multimodal benchmarks.

Table 10: Longitudinal case study of skill evolution for commonsense visual reasoning. Yellow highlights key iterative changes; Green highlights diagnostic failure patterns. $\pi$ represents the cumulative running accuracy of the skill.

| **Version** | **Trigger Condition** $c_{\text{trig}}$ | **Representative Failure** $\delta$ | **Strategy Description** $\rho_{\text{strat}}$ |
|---|---|---|---|
| Seed ($\pi = 0.51$) | Broad questions about real-world scenes and everyday situations. | *The skill answered "a chef" based on stereotype (white coat, kitchen setting) but the image showed a lab technician in a food-testing facility. Commonsense prior overrode available visual evidence.* | You are strong at commonsense reasoning about real-world scenes and everyday situations. Use what you know about how the world works to interpret the image and answer the question. Ask yourself: "What would normally happen in this scene?" Avoid letting commonsense assumptions override direct image evidence, since the image may depict an unusual or staged scenario. |
| v2 ($\pi = 0.58$) | Questions requiring inference of a *specific attribute, identity, or behavior* of a person or object from visual content. | *Given an image of two people laughing, the skill inferred "close friends" without checking attire or setting cues that indicated a formal business negotiation. Inference was plausible but unsupported.* | First identify what is directly visible and what is only inferred. Use commonsense knowledge to generate plausible interpretations. Cross-check candidate answers against explicit image evidence, and prefer conclusions that are strongly supported rather than merely typical. Be cautious with atypical, humorous, or staged scenes. |
| v5–v7 ($\pi = 0.67$) | Questions where the answer depends on evidence-grounded inference about likely attributes, roles, intentions, or actions not literally stated in the image, especially when multiple everyday interpretations are possible. | *The skill committed to "carrying groceries" after the first plausible hypothesis, ignoring a competing hypothesis ("delivering packages") that was better supported by the box shape and uniform in the image.* | First enumerate the key visual cues (appearance, pose, object interaction, scene context). Then generate 2–3 commonsense hypotheses and test each against the observed evidence. Prefer the hypothesis best supported by the image, not the one that is merely most stereotypical. If visual support is weak or ambiguous, answer conservatively and avoid overcommitting to fine-grained identity claims. |