CaSKG: Counterfactual-Causal Skill Graphs for Scalable Agent Skill Retrieval

Zhiyuan Li*
School of Artificial Intelligence, Jilin University
Ant Group
Changchun, China
zhiyuanl24@mails.jlu.edu.cn

Linyuan Gao*
School of Artificial Intelligence, Jilin University
Changchun, China
lygao25@mails.jlu.edu.cn

Xuechun Ding
Ant Group
Hangzhou, China
dingxuechun.dxc@antgroup.com

Hongwei Chen†
Ant Group
Hangzhou, China
wei.chenhw@antgroup.com

Yuan Wu†
School of Artificial Intelligence, Jilin University
Changchun, China
yuanwu@jlu.edu.cn

Yi Chang
School of Artificial Intelligence, Jilin University
Changchun, China
yichang@jlu.edu.cn

Abstract

Reusable skill libraries allow large language model (LLM) agents to reuse procedural knowledge across tasks, but they also turn memory access into a challenging retrieval problem. Full-library prompting preserves coverage at high context cost, vector retrieval returns compact neighborhoods but treats skills as independent text, and graph-based retrieval can recover workflow context only when the edges that carry relevance are reliable. We propose CaSKG, a counterfactual-causal skill graph framework that calibrates procedural relations before retrieval. CaSKG first builds a high-recall directed candidate graph from semantic, lexical, input/output, and structural evidence, with repair evidence and an optional LLM judge further refining candidate scores. It then applies direction-conditioned textual counterfactual probes that remove, substitute, and reorder skill pairs, aggregates the evidence with Bayesian smoothing, and publishes a state-filtered weighted graph for task-conditioned expansion. The graph is constructed offline and used without changing the downstream agent policy or task interface. Across six LLM backbones on ALFWorld ID-140 and ScienceWorld U211, CaSKG achieves the highest task score in all twelve combinations of model and benchmark. Relative to Graph-of-Skills (GoS), it improves the six-model macro-average ScienceWorld score from 72.62 to 80.50 and ALFWorld success from 80.01% to 86.79%, while reducing mean environment steps on both benchmarks. Qualitative and ablation analyses further show that calibrated edges help retrieval preserve prerequisites, state-changing actions, verification routines, and final completion steps. These results position edge-confidence calibration as an effective route to compact and executable skill retrieval at scale[1].

CCS Concepts

• **Computing methodologies** → **Planning and scheduling**; *Natural language processing*; • **Information systems** → **Retrieval models and ranking**.

Keywords

LLM agents, skill retrieval, graph retrieval, causal validation, counterfactual reasoning

ACM Reference Format:
Zhiyuan Li, Linyuan Gao, Xuechun Ding, Hongwei Chen, Yuan Wu, and Yi Chang. 2026. CaSKG: Counterfactual-Causal Skill Graphs for Scalable Agent Skill Retrieval. In . ACM, New York, NY, USA, 11 pages.

1 Introduction

Large language model (LLM) agents increasingly solve tasks by combining generation with external tools and application programming interfaces (APIs), and interacting with environments. Toolformer shows that language models can learn to invoke APIs during generation [22], and ReAct interleaves reasoning with actions in an environment [33]. Beyond one-time tool calls, agents can also accumulate reusable procedures: Voyager, for example, stores executable skills that can be retrieved and reused across tasks [27]. Augmented language models and systems built for large tool repositories further enlarge the action space available to an agent [14, 17, 19]. As this procedural memory grows, the central question is no longer whether an agent can use a tool, but how it can expose the right subset of skills for the current task without overwhelming the context.

Skill retrieval is difficult because useful task context is rarely a single textually matching procedure. A household or science task may require prerequisites, object-location routines, state-changing actions, verification steps, and recovery procedures whose descriptions do not all overlap with the task instruction. Full-library exposure preserves recall by making every skill visible, but it shifts the selection burden to the agent and introduces many irrelevant alternatives. Dense retrieval and retrieval-augmented generation (RAG) reduce the context size by ranking items according to semantic similarity [7, 9], yet they usually treat skills as independent

*Equal contribution; work done while Zhiyuan Li was an intern at Ant Group.
†Co-corresponding authors.
[1]Code is available at: https://github.com/ZhiyuanLi218/Caskg

text units. This is a poor fit for procedural memory, where the utility of one skill often depends on another skill that prepares, checks, repairs, or follows it. Recent tool-retrieval evidence reinforces this mismatch: textual retrieval scores alone do not reliably predict whether a returned capability will be useful for the current task [24].

Graph-based skill retrieval is a natural response to this limitation. Instead of retrieving isolated skills, GoS constructs an offline skill graph and propagates query relevance from lexical and semantic seeds so that workflow-related skills can enter the retrieved bundle [12]. This relational view addresses an important weakness of vector-only retrieval: a skill can be operationally necessary even when it is not the closest textual neighbor of the task instruction. However, graph retrieval introduces a different failure mode. Once relevance starts to propagate, the quality of the retrieved bundle depends on the edges that carry that relevance. An edge induced by topical similarity, co-occurrence, or loose interface compatibility may look plausible during construction but still connect skills that are alternatives, unordered neighbors, or operationally unsuitable for the current procedure.

The bottleneck is therefore not simply how to build a larger skill graph, but how to decide which candidate relations should influence retrieval. Publishing every plausible association increases coverage but can spread relevance through weak edges and pollute the skill context. Pruning aggressively can remove useful paths and leave parts of the library unreachable. Exhaustively assessing all ordered skill pairs is also impractical because the number of possible relations grows quadratically with the library size. This paper formulates skill-graph construction as a budgeted edge-confidence calibration problem: given a high-recall pool of candidate relations, the system must decide which edges deserve assessment, how much confidence each assessed relation should receive, and how that confidence should control graph propagation.

We propose CaSKG, a counterfactual-causal skill graph framework that separates association discovery from edge reliability assessment. CaSKG first induces a high-recall directed candidate graph from skill-level evidence, including semantic, lexical, input/output, and structural signals, with repair evidence and an optional LLM judge refining candidate scores. The framework also reserves trace co-occurrence and existing-relation channels for self-evolving skill libraries; in the static construction used in this study, these channels remain extension points rather than active inputs. The active construction signals are used to discover plausible source-to-target hypotheses rather than to directly publish every relation at full strength. CaSKG then allocates a limited validation budget to selected candidate edges and evaluates each directed hypothesis with three textual counterfactual probes: removing the source skill, replacing it with a dissimilar skill, and reversing the proposed order. These probes estimate whether the target skill depends on the source, whether the source is specific rather than interchangeable, and whether the relation has a meaningful workflow direction [10, 34].

The probe outputs are aggregated with a Beta-smoothed posterior that yields an edge-level reliability estimate. CaSKG converts this estimate into a publication state: confirmed relations retain full support, uncertain relations are downweighted, rejected relations are removed, and a bounded set of unvalidated candidates is kept only as a low-weight scaffold for coverage. The resulting graph is frozen before evaluation. At runtime, lexical and semantic matches initialize a personalized PageRank-style expansion over this state-filtered graph, so the downstream agent receives a compact skill bundle shaped by calibrated procedural structure rather than by raw association strength alone. CaSKG therefore changes the graph over which retrieval propagates without adding a new online action policy or changing the task interface.

We evaluate CaSKG against full-library exposure, vector retrieval, and GoS on ALFWorld ID-140 and ScienceWorld U211 with six LLM backbones [25, 29]. In the complete archived comparison, CaSKG obtains the highest reported task score in all 12 backbone and benchmark groups. Relative to GoS, the six-model macro-average ScienceWorld score rises from 72.62 to 80.50, and ALFWorld success rises from 80.01% to 86.79%. Additional task-type, trajectory, scale, and component analyses show that the gains are consistent with the intended mechanism: improving the reliability of graph edges helps retrieve skill bundles that preserve prerequisites, intermediate state changes, verification routines, and final task-completion actions.

The main contributions of this work are:

- We formulate scalable skill-graph construction as a budgeted edge-confidence calibration problem, where retrieval must balance relational coverage against the risk that weak associations distort graph propagation.
- We develop CaSKG, an offline graph-construction method that combines multi-signal candidate induction with LLM-simulated counterfactual probes and Bayesian state-gated publication before runtime retrieval.
- We evaluate CaSKG on two interactive benchmarks with six LLM backbones and analyze its behavior through aggregate results, task-type gains, trajectory examples, library-scale records, and graph-construction ablations.

2 Related Work

Tool retrieval. Tool-augmented language models have evolved from local invocation decisions toward retrieval over large and reusable capability libraries. Early work framed tool use as a control problem inside generation: Toolformer learns to insert API calls into model outputs, ReAct couples reasoning traces with environment actions, and augmented-language-model surveys place these behaviors in a broader family of models that call external modules and environments [14, 22, 33]. As tool repositories grow, the challenge is no longer limited to deciding whether a call should be made. Gorilla connects language models to large API collections, while API-Bank, ToolBench, and ToolLLM turn large-scale tool selection, planning, and composition into explicit evaluation settings [11, 17, 19, 32]. A further step is to treat reusable action sequences as skills rather than isolated interfaces: Voyager stores executable skills across tasks, and ToolRet shows that generic retrieval scores are not reliable indicators of whether a returned tool will solve the current task [24, 27]. This line of work establishes the need for retrieval, but it leaves open how an agent should retrieve a coherent set of interdependent procedures rather than a single callable tool.

Graph retrieval. The need to retrieve interdependent capabilities connects skill retrieval to a broader movement from independent

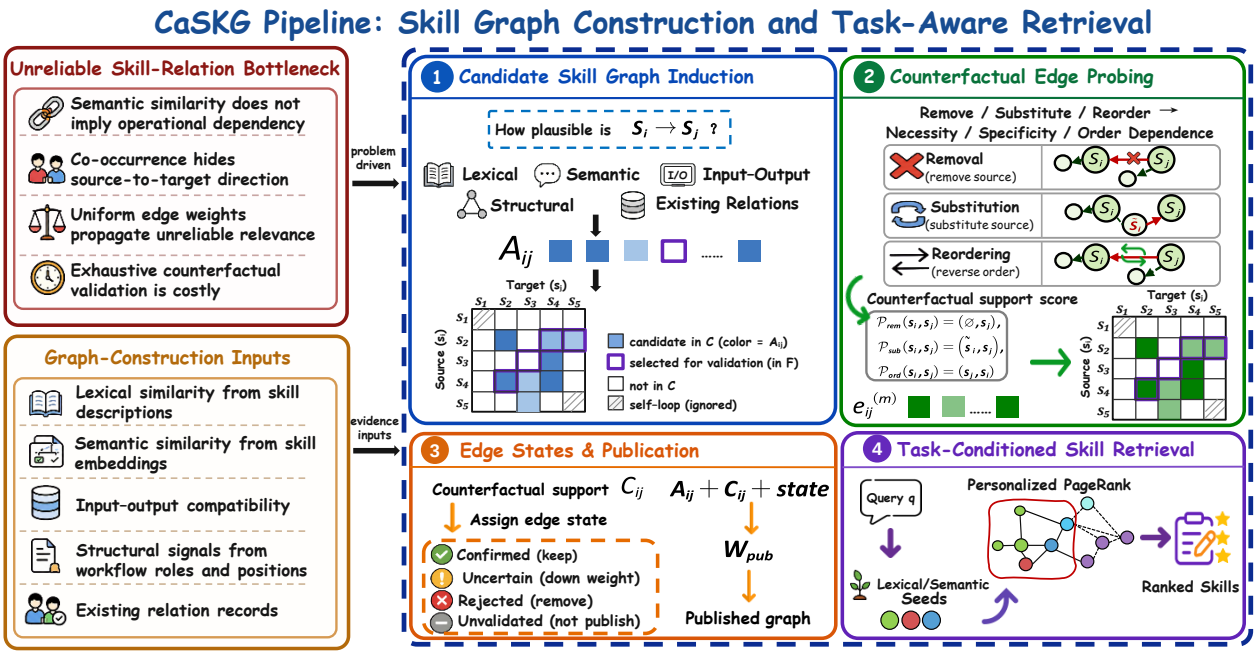

Figure 1: Overview of CaSKG. Multiple skill-level signals induce a directed candidate set. A budgeted subset is examined with direction-conditioned removal, substitution, and reordering probes. The resulting evidence assigns edge states and publication weights, producing the published skill graph. Task-conditioned retrieval then propagates query relevance over this graph to return a compact skill context.

matching toward structured memory. Dense Passage Retrieval and RAG provide scalable query-to-memory retrieval, but their standard use largely scores each unit independently [7, 9]. Graph-based retrieval adds a second source of evidence: relations among memory units. Topic-Sensitive PageRank shows that query-biased propagation can use links among candidates, and recent systems such as GraphRAG and HippoRAG adapt graph-structured memory to corpus and knowledge retrieval [3–5]. This idea has also moved into tool and skill settings. ToolNet represents relations among tools, Graph-of-Skills builds a dependency-aware graph for executable skills and retrieves bounded skill bundles through graph propagation, GraSP models skill composition with typed preconditions and effects, and SkillReranker uses task states to refine the candidate order [2, 12, 13, 30]. These methods show that relational structure can recover useful capabilities missed by one-shot semantic similarity. Their effectiveness, however, depends on the reliability of the relations over which relevance propagates. Edges induced from semantic similarity, co-use, interface compatibility, or workflow order may be plausible associations without being valid procedural dependencies. Once such edges are published, multi-hop propagation can carry relevance toward unrelated or operationally unsuitable skills. The assessment of candidate relations before they influence skill retrieval therefore remains a central unresolved issue.

Causal graph retrieval. Causal graphs provide a complementary perspective on relation quality because they distinguish directional dependence from statistical association. The potential-outcomes framework and structural causal models formalize interventions and counterfactuals, while causal representation learning extends this view to learned variables and mechanisms [18, 20, 23]. Work on language models has further examined whether causal information expressed in text can support associational, interventional, and counterfactual reasoning [6, 8]. Retrieval research has started to use this structure: Causal Graph RAG retrieves causal graphs as structured context for language-model reasoning, and CausalRAG incorporates causal graph construction and tracing into RAG to improve contextual continuity, retrieval precision, and interpretability [21, 28]. These studies show that retrieval can benefit from directional and explanatory structure, but they mainly target knowledge corpora and question answering. In contrast, large procedural memories require relation assessment over executable or reusable skills, where an edge should indicate whether one procedure supports, orders, verifies, or repairs another. Existing graph-based tool and skill retrieval still relies mostly on transitions, dependency labels, similarity, task states, and precondition-effect relations [2, 12, 13, 30]. CaSKG addresses this gap by using counterfactual edge evidence to calibrate which candidate skill relations are allowed to shape graph-based retrieval.

3 Method

3.1 Framework Overview

CaSKG is an offline graph-construction and retrieval framework for providing compact procedural context from a reusable skill library. Given a skill library $\mathcal{S} = \{s_1, \ldots, s_n\}$ of n skills, CaSKG constructs a directed candidate graph over ordered skill pairs, calibrates selected edges with direction-conditioned textual counterfactual evidence, and publishes a state-filtered weighted graph. At inference time, CaSKG uses a task query q to identify seed skills, propagates their relevance over the published graph, and retrieves a compact skill bundle for the downstream agent. The agent policy, task interface, and environment interaction loop remain unchanged.

The framework separates relation discovery, edge assessment, graph publication, and task-time retrieval. Candidate induction prioritizes coverage, bringing plausible prerequisite, workflow, and recovery relations into the candidate graph. Counterfactual edge probing estimates whether a candidate relation reflects an operational dependency rather than only topical similarity or unordered co-occurrence. Graph publication determines which relations can transmit query relevance and with what strength. The retrieval stage then expands from task-relevant seeds over the calibrated graph. This design keeps procedural coverage broad while limiting the influence of weak associations. Figure 1 summarizes the four stages of CaSKG: candidate skill graph induction, counterfactual edge probing, edge-state assignment and graph publication, and task-conditioned skill retrieval. The following subsections describe these stages in the same order.

3.2 Candidate Skill Graph Induction

CaSKG first induces a sparse set of directed candidate relations $C \subseteq \mathcal{S} \times \mathcal{S}$. Each $(s_i, s_j) \in C$ denotes the ordered hypothesis $s_i \rightarrow s_j$, where the source skill s_i may provide operational support for the target skill s_j. This stage prioritizes coverage: it collects plausible relations for later calibration while leaving reliability to subsequent assessment.

Multi-source evidence. CaSKG is organized around heterogeneous evidence channels from skill descriptions, semantic representations, input/output interfaces, and workflow roles. Lexical and semantic signals capture textual and conceptual affinity between skills, while input/output and structural signals capture interface compatibility and workflow position. Repair evidence, when available, indicates whether one skill can help recover from or complete another procedure. The framework also defines two extension channels for self-evolving skill libraries: future execution logs can contribute trace co-occurrence evidence, and previously recorded relations can preserve continuity across graph updates. The static construction in this paper instantiates the active skill-level channels above, while the trace and relation-history channels remain reserved interfaces for future self-evolving updates. The union of the active recall channels, followed by deduplication and local truncation of low-priority neighbors, yields the candidate set C. For edges already in C, repair evidence and an optional LLM judge further refine the association score [36].

Initial association score. For each candidate edge $(s_i, s_j) \in C$, let $\mathcal{A}_{ij}$ be the nonempty set of active scoring signals for the pair, with positive total signal weight. Each $\phi_k(i,j) \in [0,1]$ is the normalized value of signal k, and $\lambda_k \geq 0$ is the corresponding signal weight. The value $\phi_{\text{struct}}(i,j)$ denotes the normalized structural signal for the ordered pair. The parameters τ_{str} and η_{str} denote the activation threshold and retention coefficient for this structural signal. CaSKG first computes the weighted active-signal support $\widetilde{A}_{ij}$ and then applies a structural floor to obtain the initial association score A_{ij}:

$$\begin{aligned} \widetilde{A}_{ij} &= \text{clip}_{[0,1]}\left(\frac{\sum_{k\in\mathcal{A}_{ij}} \lambda_k \phi_k(i,j)}{\sum_{k\in\mathcal{A}_{ij}} \lambda_k}\right), \\ A_{ij} &= \begin{cases} \max\left(\widetilde{A}_{ij}, \eta_{\text{str}}\phi_{\text{struct}}(i,j)\right), & \phi_{\text{struct}}(i,j) > \tau_{\text{str}}, \\ \widetilde{A}_{ij}, & \text{otherwise.} \end{cases} \end{aligned} \tag{1}$$

For any interval $[a,b]$, the operator $\text{clip}_{[a,b]}$ truncates its input to that interval; here, $\text{clip}_{[0,1]}$ bounds the score to the unit interval. The weighted average uses only active signals, so unavailable signals do not become negative evidence. The structural floor preserves strong workflow evidence when $\phi_{\text{struct}}(i,j)$ exceeds τ_{str}. Thus, $A_{ij} \in [0,1]$ is the initial edge weight on the weighted candidate graph (C, A); pairs outside C have no candidate edge.

The association score serves two purposes. It ranks candidate edges for the limited counterfactual validation pass, and it remains available as discovery support during graph publication. CaSKG selects a budgeted validation frontier $F \subseteq C$ from the candidate graph. Edges in F are sent to the counterfactual probes, while edges in $C \setminus F$ remain unvalidated rather than being treated as negative evidence.

3.3 Counterfactual Edge Probing

An association score can indicate that two skills are related, but it cannot by itself determine whether the ordered edge $s_i \rightarrow s_j$ captures an operational dependency. A high-scoring association may still connect skills that are interchangeable alternatives, unordered neighbors, or only topically similar. CaSKG therefore probes each selected candidate edge in F with textual counterfactual tests conditioned on the proposed source-to-target direction.

Direction-consistent probes. Building on prior studies of language-model counterfactual reasoning and evaluation [10, 34], CaSKG applies three complementary probes to the skill descriptions of each $(s_i, s_j) \in F$. Let $\mathcal{P}_m(s_i, s_j)$ denote the modified relation context for probe type m. The removal probe makes the source skill unavailable and tests whether the target skill is impaired, measuring necessity. The substitution probe replaces the source with a low-overlap skill $\widetilde{s}_i$ and tests whether the target skill degrades, measuring source specificity. The reordering probe reverses the proposed relation and tests whether the workflow remains coherent, measuring directionality:

$$\begin{aligned} \mathcal{P}_{\text{rem}}(s_i, s_j) &= (\varnothing, s_j), \\ \mathcal{P}_{\text{sub}}(s_i, s_j) &= (\widetilde{s}_i, s_j), \\ \mathcal{P}_{\text{ord}}(s_i, s_j) &= (s_j, s_i), \end{aligned} \tag{2}$$

where $\varnothing$ denotes removing the source skill from the relation context and $\widetilde{s}_i$ denotes the substitute skill used in the substitution test.

For each probe type $m \in \mathcal{M} = \{\text{rem}, \text{sub}, \text{ord}\}$, the LLM returns an oriented counterfactual support score $e_{ij}^{(m)} \in [0,1]$. A larger score means that removing the source, replacing it with the

substitute, or reversing the proposed order would more strongly undermine the hypothesized dependency. The three probe scores share the same support direction and can therefore be aggregated directly: $e_{ij}^{(\mathrm{rem})}$ captures impairment under source removal, $e_{ij}^{(\mathrm{sub})}$ captures degradation under substitution, and $e_{ij}^{(\mathrm{ord})}$ captures loss of workflow coherence under reversal. These scores provide text-level evidence for necessity, specificity, and order dependence of the candidate edge.

3.4 Bayesian Edge Calibration and Graph Publication

The probe scores are converted into a smoothed relation-reliability estimate rather than being used directly as edge weights. CaSKG then maps each candidate edge to a publication state, which determines whether the edge is published, attenuated, removed, or retained as a low-weight scaffold for coverage.

Reliability estimation. CaSKG uses a Beta-form accumulator initialized at $\mathrm{Beta}(1,1)$ to combine the three counterfactual views. For a probe score $e_{ij}^{(m)}$, the binary variable $z_{ij}^{(m)}$ records whether the probe supports the directed relation, and $\delta_{ij}^{(m)}$ records the evidence mass. The parameter $\epsilon_e > 0$ is a minimum evidence-mass floor that prevents near-midpoint probe judgments from being ignored entirely:

$$\begin{aligned} z_{ij}^{(m)} &= \mathbb{I}\left[e_{ij}^{(m)} > 0.5\right], \\ \delta_{ij}^{(m)} &= \max\left(2\left|e_{ij}^{(m)} - 0.5\right|, \epsilon_e\right). \end{aligned} \tag{3}$$

Here, $\mathbb{I}[\cdot]$ is the indicator function. The midpoint 0.5 determines evidence polarity, while the distance from the midpoint determines how much evidence the probe contributes. The aggregated Beta parameters are

$$\begin{aligned} \alpha_{ij} &= 1 + \sum_{m \in \mathcal{M}} z_{ij}^{(m)} \delta_{ij}^{(m)}, \\ \beta_{ij} &= 1 + \sum_{m \in \mathcal{M}} \left(1 - z_{ij}^{(m)}\right) \delta_{ij}^{(m)}. \end{aligned} \tag{4}$$

where α_{ij} accumulates positive evidence for the directed relation and β_{ij} accumulates evidence against it. The normalized Beta mean gives the smoothed relation-reliability score:

$$c_{ij} = \frac{\alpha_{ij}}{\alpha_{ij} + \beta_{ij}}. \tag{5}$$

The score $c_{ij} \in [0,1]$ summarizes probe-derived support for the ordered relation $s_i \to s_j$.

State-gated publication. The association score A_{ij} represents discovery support, and the reliability score c_{ij} represents counterfactual support. CaSKG uses a symmetric confirmation threshold $\tau_c \in (0.5, 1)$ to assign an assessment state σ_{ij}:

$$\sigma_{ij} = \begin{cases} \text{confirmed}, & (s_i,s_j) \in F \ \wedge \ c_{ij} > \tau_c, \\ \text{rejected}, & (s_i,s_j) \in F \ \wedge \ c_{ij} < 1 - \tau_c, \\ \text{uncertain}, & (s_i,s_j) \in F \ \wedge \ 1 - \tau_c \le c_{ij} \le \tau_c, \\ \text{unvalidated}, & (s_i,s_j) \in C \setminus F, \end{cases} \qquad (s_i,s_j) \in C. \tag{6}$$

Confirmed edges receive strong evidence from the probes, rejected edges receive counter-evidence, uncertain edges remain within the middle band, and unvalidated edges are candidates that did not enter the validation frontier. A bounded subset $E_{\mathrm{scaf}} \subseteq C \setminus F$ is retained as scaffold edges for coverage; their weights are governed by the scaffold attenuation rather than by probe-derived reliability.

Define $\widehat{c}_{ij} = c_{ij}$ for $(s_i,s_j) \in F$ and $\widehat{c}_{ij} = 0$ otherwise. Let $\epsilon_w > 0$ be the floor for positive published edge weights. Let ρ_{unc} and ρ_{scaf} be the attenuation coefficients for uncertain and scaffold edges, with $1 > \rho_{\mathrm{unc}} > \rho_{\mathrm{scaf}} > 0$. CaSKG combines discovery support and probe-derived reliability into b_{ij} and then applies the state-dependent publication gate ρ_{ij}:

$$\begin{aligned} b_{ij} &= \max\left(A_{ij}, \widehat{c}_{ij}, \epsilon_w\right), \\ \rho_{ij} &= \begin{cases} 1, & \sigma_{ij} = \text{confirmed}, \\ \rho_{\mathrm{unc}}, & \sigma_{ij} = \text{uncertain}, \\ \rho_{\mathrm{scaf}}, & \sigma_{ij} = \text{unvalidated} \ \wedge \ (s_i,s_j) \in E_{\mathrm{scaf}}, \\ 0, & \text{otherwise}, \end{cases} \\ w_{ij}^{\mathrm{pub}} &= \begin{cases} \mathrm{clip}_{[\epsilon_w,1]}(\rho_{ij} b_{ij}), & \rho_{ij} > 0, \\ 0, & \rho_{ij} = 0. \end{cases} \end{aligned} \tag{7}$$

Here, b_{ij} keeps the stronger available support source, ρ_{ij} controls how the edge state affects propagation, and w_{ij}^{pub} is the final published edge weight. Confirmed relations retain full support, uncertain relations are downweighted, rejected relations are removed, and selected unvalidated candidates serve as attenuated scaffold links.

The publication step produces the graph

$$\begin{aligned} G_{\mathrm{pub}} &= (\mathcal{S}, E_{\mathrm{pub}}, W, \Sigma), \\ E_{\mathrm{pub}} &= \{(s_i,s_j) \in C : w_{ij}^{\mathrm{pub}} > 0\}, \end{aligned} \tag{8}$$

where E_{pub} is the published edge set, W contains the published weights, and Σ contains the assessment states. This graph is the frozen structure used by the retrieval stage.

3.5 Task-Conditioned Skill Retrieval

At inference time, CaSKG uses G_{pub} as the structural substrate for skill retrieval. Given a task query q, lexical and semantic rankings provide initial task-relevant seeds. These seeds are reranked and converted into an inverse-rank-weighted seed distribution π_q, where higher-ranked skills receive larger probability mass. In the personalized PageRank update below, π_q anchors the restart term to the current task, while graph propagation allows related skills to enter the retrieved context even when they are not direct semantic neighbors of the query.

Query-conditioned diffusion. Let T be the row-normalized transition matrix derived from the published graph G_{pub} and its edge weights W. Let $\gamma \in (0,1)$ be the restart coefficient, which controls how strongly each update returns to π_q, and let $p^{(t)}$ be the skill relevance distribution after t propagation steps. Starting from $p^{(0)} = \pi_q$, personalized PageRank [5] iterates

$$p^{(t+1)} = \gamma \pi_q + (1-\gamma) T^\top p^{(t)}. \tag{9}$$

After convergence, writing p for the limiting relevance distribution, CaSKG ranks skills by p and returns the highest-ranked skill summaries or procedures as the task context. The restart term keeps the expansion tied to the query, while the state-gated edge weights determine which procedural relations transmit relevance and how strongly they do so. In this way, CaSKG couples task-local semantic seeds with reliability-weighted relational expansion, allowing procedurally related skills to enter the context while reducing the influence of weak relations.

4 Experiments

The experiments examine whether calibrating skill-graph edges before retrieval improves both task outcomes and interaction behavior. The comparison is designed around three questions: whether CaSKG improves over full-library access, independent semantic retrieval, and an existing graph-retrieval baseline; whether any task-score gain comes with additional environment interaction cost; and whether the observed gains can be explained by more coherent procedural skill bundles. We evaluate all methods under a frozen Skill1000 library on two interactive benchmarks, report the common protocol and main results, and then analyze task-type and trajectory-level behavior.

4.1 Experimental Setup

Benchmarks. We use two interactive benchmarks with complete evaluated cohorts. ALFWorld ID-140 contains 140 in-distribution household-task episodes and measures whether the agent completes the goal. ScienceWorld U211 contains 211 evaluated science-task episodes and reports the official best score for each episode. The main comparison uses a Skill1000 library. For every method, the library or retrieval structure is built offline and kept fixed during evaluation, so performance differences reflect how skill context is exposed rather than online adaptation of the skill store.

Baselines. We compare four retrieval settings that correspond to different ways of managing the tradeoff between coverage and noise. Vanilla Skills exposes the complete Skill1000 catalog without graph retrieval, maximizing recall but leaving filtering to the agent. Vector Skills retrieves skills independently by embedding similarity, reducing context size but ignoring procedural links among skills. GoS uses dependency-aware graph construction and graph propagation to recover skills connected to the query. CaSKG uses the counterfactual-causal, state-weighted graph constructed by the procedure in Section 3.

Models and evaluation. The main comparison covers MiniMax-M2.7 [1], GLM-5.2 [35], Kimi-K2.6 [15], Qwen3.5-397B-A17B [26], DeepSeek-V4-Flash [31], and GPT-5.6-Luna [16]. Within each benchmark, methods use the same task cohort, prompt, evaluator, episode limits, and environment interaction loop. CaSKG publishes its graph before evaluation and does not update the graph or add a separate online planner during an episode. Each accepted record stores per-episode outcomes, environment steps, retrieved context, and infrastructure diagnostics.

Metrics and reporting rules. On ALFWorld, R is the success rate over 140 tasks and is reported as a percentage. On ScienceWorld, R is the mean best official score over 211 episodes and is not a percentage.

Table 1: Reward and mean interaction steps at Skill1000 on ALFWorld ID-140 and ScienceWorld U211.

Model	Method	ALFWorld ID-140		ScienceWorld U211	
		R (%) ↑	Steps↓	R ↑	Steps↓
MiniMax-M2.7	Vanilla	42.90	22.54	45.90	21.73
	Vector	45.70	22.84	43.21	21.45
	GoS	<u>63.60</u>	<u>19.69</u>	<u>55.85</u>	<u>18.91</u>
	CaSKG	**73.57**	**18.44**	**68.33**	**17.45**
GLM-5.2	Vanilla	95.00	11.05	75.50	17.03
	Vector	<u>96.43</u>	10.12	77.07	16.65
	GoS	95.71	<u>9.91</u>	<u>80.33</u>	<u>15.75</u>
	CaSKG	**97.86**	**9.69**	**85.11**	**14.52**
Kimi-K2.6	Vanilla	77.90	16.07	72.23	18.91
	Vector	90.00	13.49	72.58	17.55
	GoS	<u>93.60</u>	<u>13.08</u>	<u>76.82</u>	<u>16.15</u>
	CaSKG	**95.00**	**12.34**	**83.88**	**15.43**
Qwen3.5-397B-A17B	Vanilla	79.30	15.60	<u>63.72</u>	18.34
	Vector	78.60	15.49	62.60	18.51
	GoS	<u>88.60</u>	<u>14.15</u>	63.18	<u>17.08</u>
	CaSKG	**92.14**	**11.60**	**74.97**	**15.56**
DeepSeek-V4-Flash	Vanilla	72.86	16.91	64.84	18.49
	Vector	<u>78.57</u>	<u>16.89</u>	68.65	18.39
	GoS	77.86	17.09	<u>73.45</u>	<u>16.20</u>
	CaSKG	**86.43**	**14.41**	**83.40**	**15.61**
GPT-5.6-Luna	Vanilla	<u>72.86</u>	**17.74**	84.09	14.99
	Vector	55.00	22.06	84.09	14.40
	GoS	60.71	21.86	<u>86.08</u>	<u>14.22</u>
	CaSKG	**75.71**	<u>17.79</u>	**87.33**	**13.18**

Bold and <u>underlined</u> values are the best and second-best reported results within each model and metric; higher R and lower Steps are better. Rankings require at least two reported values. CaSKG rows are shaded blue.

We also report *Steps*, the arithmetic mean number of environment interactions per episode over the complete valid cohort; higher R and fewer steps are better. Steps measures observed interaction consumption during task execution rather than token usage, wall-clock latency, retrieval latency, or graph-construction cost.

4.2 Main Results

Table 1 shows that CaSKG provides the strongest overall task-performance–interaction-cost profile. It obtains the highest task score for every backbone on both benchmarks and uses fewer mean steps than GoS in all twelve model–benchmark settings. Averaged across the six backbones, the ScienceWorld score rises from 72.62 with GoS to 80.50 with CaSKG, while ALFWorld success rises from 80.01% to 86.79%. This consistency across weaker and stronger backbones is important: the gain is not confined to a single model family or to cases where the base model is weak. Instead, the results indicate that improving the retrieval graph changes the usefulness of the context supplied to the same downstream agent loop.

Overall comparison. The four retrieval settings expose different failure modes. Vanilla Skills keeps the entire library available, so it can include useful procedures but also forces the model to search through irrelevant alternatives during task execution. Vector Skills reduces this burden, but it retrieves each skill as an independent text item and therefore misses procedures whose value comes from workflow position rather than lexical similarity. GoS improves over vector retrieval by allowing relevance to move through a skill graph, which is especially helpful when the required skill bundle contains prerequisites or follow-up actions. CaSKG

keeps this relational benefit while filtering the graph through counterfactual edge evidence. The result is a retrieval context that is neither an unfiltered catalog nor a set of isolated nearest neighbors; it is a compact bundle shaped by validated procedural relations.

ScienceWorld. ScienceWorld is the benchmark where the benefit of calibrated graph retrieval is most visible. Many tasks require a sequence of preparation, operation, observation, and classification steps rather than a single action named in the task instruction. Under this condition, direct semantic matches often recover only part of the procedure. GoS already improves over Vector Skills for all six backbones, confirming that graph propagation helps retrieve supporting skills beyond the nearest textual neighbors. CaSKG adds a second layer of improvement by making the propagated edges more selective. The largest gaps appear for MiniMax-M2.7 and DeepSeek-V4-Flash, where CaSKG improves over GoS by 12.48 and 9.95 points, respectively. The advantage narrows for stronger backbones, but it remains positive: CaSKG is still ahead of GoS by 4.78 points for GLM-5.2 and 1.25 points for GPT-5.6-Luna. This pattern suggests that CaSKG is most useful when a model needs retrieval to supply missing procedural structure, while still providing incremental benefit when the model already solves many episodes.

ALFWorld. ALFWorld shows the same advantage in a more stateful household-control setting. Success requires not only identifying the target object or receptacle, but also preserving the order of navigation, pickup, state change, and placement. The largest separation again occurs for MiniMax-M2.7: CaSKG improves on GoS by 9.97 percentage points and on both non-graph settings by more than 27 points. Kimi-K2.6 and Qwen3.5-397B-A17B follow the same ranking pattern, while GLM-5.2 is already near the success ceiling and leaves less room for improvement. GPT-5.6-Luna reveals why retrieval design still matters for strong models: Vector Skills and GoS fall sharply below Vanilla Skills, whereas CaSKG recovers the best success rate among all settings. The result indicates that structural retrieval is beneficial only when the structure suppresses misleading links; otherwise, graph propagation can be worse than exposing the full catalog.

Environment-interaction cost. The task-score gains are not obtained by spending more environment interactions. Under the shared episode limit, CaSKG reduces the ScienceWorld six-model mean to 15.29 steps, compared with 16.39 for GoS and roughly 18 steps for the two non-graph settings. It is the shortest method for every ScienceWorld backbone. On ALFWorld, CaSKG averages 14.05 steps, below GoS at 15.96 and the two non-graph settings at roughly 16.7–16.8 steps. It also uses the fewest steps in five of six ALFWorld backbones; the only exception is GPT-5.6-Luna, where Vanilla Skills is marginally shorter (17.74 versus 17.79) but less successful. Across both benchmarks, CaSKG uses fewer steps than GoS in all twelve model–benchmark settings and has the lowest mean step count among all methods in eleven. The more plausible interpretation is that calibrated graph retrieval reduces exploratory and corrective actions by giving the agent a more executable procedure, rather than simply allowing it to interact longer. This is still an aggregate interaction-cost pattern, not a claim about token usage, latency, graph-construction cost, or success-conditioned efficiency.

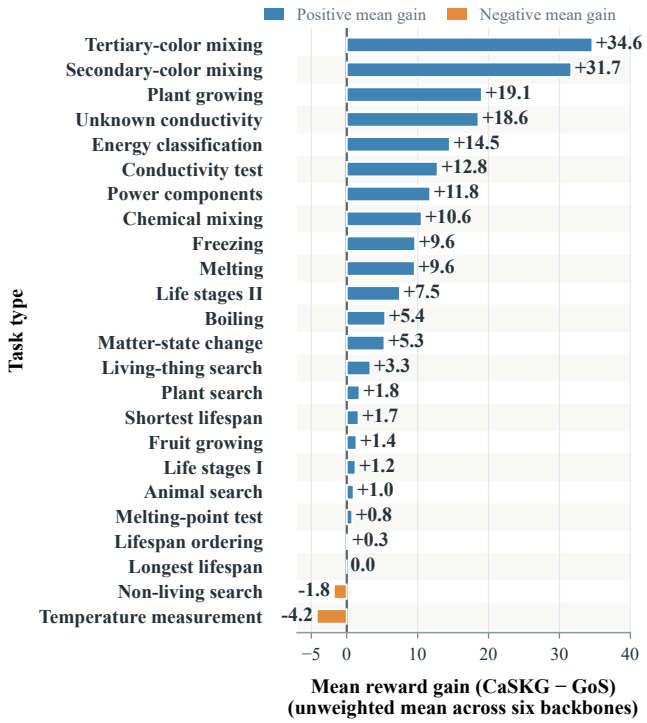

Figure 2: Task-type reward gains of CaSKG over GoS on ScienceWorld U211. All 24 task types are shown individually and ordered by gain; each bar is the unweighted mean reward difference across the six evaluated backbones. Blue and orange indicate positive and negative gains, respectively.

Overall, the results support the intended advantage of CaSKG: it improves the structure of retrieved context rather than merely changing the amount of context. The following qualitative analysis examines this explanation at the task-type and trajectory levels.

4.3 Qualitative Analysis

We use qualitative analysis to connect the aggregate improvements in Table 1 to the retrieval mechanism. The first view is a task-type breakdown on ScienceWorld. Figure 2 reports the unweighted mean gain of CaSKG over GoS across the six evaluated backbones for all 24 U211 task types, ordered by gain. CaSKG improves on 21 task types, ties on one, and trails GoS on two. The largest gains appear in tertiary- and secondary-color mixing, plant growing, unknown conductivity, and energy classification. These categories typically require multiple dependent operations: setting up materials, applying a transformation, observing the result, and mapping the observation to a final answer. The two negative cases, non-living-thing search and temperature measurement, are more direct retrieval or measurement tasks, where additional graph expansion can offer less advantage and may occasionally introduce distraction.

The task-type pattern suggests that CaSKG is most useful when success depends on preserving a chain of operations rather than retrieving one obviously named skill. We therefore examine one representative task from each benchmark and compare all four

methods. The examples are not intended as independent proof of the aggregate results; instead, they illustrate the failure modes behind the table: full-library exposure can leave the agent to choose among noisy alternatives, vector retrieval can miss non-lexical dependencies, and graph retrieval can still be misled if weak edges carry relevance into the retrieved context.

ScienceWorld: conductivity testing. The conductivity example shows how CaSKG turns a partially matched task into an executable procedure. Determining whether sodium chloride conducts electricity requires focusing on the material, assembling a valid circuit, observing the result, classifying the material, and placing it in the corresponding box. In a MiniMax-M2.7 episode of `test-conductivity`, CaSKG retrieved guidance covering conductivity testing, circuit construction and connection, material classification, and final placement. The agent completed the circuit, identified sodium chloride as nonconductive, and placed it in the green box, scoring 100 in 24 steps. The baselines fail in different ways. GoS reached 55 after 30 steps because unrelated utilities entered the context and the agent spent its remaining interactions repairing connection commands. Vanilla Skills also scored 55 in 29 steps; although the full catalog contained relevant skills, it did not prevent repeated trials of alternative endpoint descriptions. Vector Skills scored 5 in 30 steps because semantic retrieval supplied mostly unrelated technical utilities and no workable circuit plan. The advantage of CaSKG in this case is not simply that it retrieved a skill about conductivity. It retrieved a bundle that preserved the preparation–operation–verification–placement dependency chain needed to finish the task.

ALFWorld: cooling and placement. The ALFWorld example highlights the same mechanism in a stateful household workflow. Cooling an apple and placing it on a countertop requires the agent to search, pick up the object, locate or use the cooling appliance, track the changed object state, and complete the final placement. In the GLM-5.2 episode `pick_cool_then_place_in_recep-Apple-None-CounterTop-14`, CaSKG retrieved complementary guidance for locating the appliance, cooling the object, tracking its state, and operating the destination receptacle. It found the apple, cooled it, and placed it on the countertop in 27 steps. Vanilla Skills, Vector Skills, and GoS all received zero reward at the 30-step limit, but their failures differ. Vanilla Skills reached cooling but did not complete the final move, showing that recall alone does not guarantee procedural completion. Vector Skills retrieved mostly unrelated simulation and parallelization skills, so its compact context omitted key household dependencies. GoS found the apple late, completed cooling, and exhausted its budget before placement despite retrieving a temperature-regulation skill. The case illustrates why relevance to a state-changing action is insufficient: successful retrieval must also preserve the search, object-state tracking, and final placement relations that define the task.

The two examples lead to the same conclusion as the aggregate results. CaSKG succeeds not because it names the target operation more often, but because it returns a usable chain of adjacent skills: preparation, state-changing action, verification, and final completion. The baselines expose the complementary failures behind this result. Full-library access has recall but weak ordering pressure, vector retrieval is compact but can omit non-lexical dependencies, and uncalibrated graph propagation can include relevant-looking edges that do not support the next executable step. This explains why CaSKG is most advantageous when the task objective hides several dependent subgoals. At the same time, the negative task types in Figure 2 show that graph structure is not universally beneficial; when the task is close to a direct search or measurement operation, the additional relational context has less room to help. The main result of the qualitative analysis is therefore selective procedural expansion: CaSKG keeps retrieval broad enough to include prerequisites and follow-up actions, but constrains propagation so that weak associations are less likely to dominate the agent's context.

Table 2: Archived skill-library scale comparison on ALFWorld ID-140 for MiniMax-M2.7 and Qwen3.5-397B-A17B.

Model	Skills	R (%) ↑			Steps↓	
		CaSKG	GoS	ΔR	CaSKG	GoS
MiniMax-M2.7	200	**57.14**	50.00	+7.14	**20.73**	22.21
	500	**67.86**	45.00	+22.86	**19.95**	23.07
	1,000	**73.57**	63.60	+9.97	**18.44**	19.69
	2,000	**70.00**	54.29	+15.71	**18.71**	21.32
Qwen3.5-397B-A17B	200	**85.00**	76.43	+8.57	**14.50**	16.25
	500	**94.29**	72.86	+21.43	**12.48**	16.94
	1,000	**92.14**	88.60	+3.54	**11.60**	14.15
	2,000	**91.43**	77.86	+13.57	**12.31**	16.47

R is success rate, $\Delta R = R_{\text{CaSKG}} - R_{\text{GoS}}$ is measured in percentage points, and Steps is the mean number of environment interactions. Values reproduce archived aggregate records; Qwen GoS values are archived aggregate counterparts rather than episode directories. In the MiniMax runs, CaSKG assesses 500 candidate relations at 200–1,000 skills and 2,000 relations at 2,000 skills, so this is a descriptive system-level scale comparison. Bold denotes the better observed result in each method pair.

5 Ablation Study

The ablation study further attributes the gains in Section 4 to the main design choices in CaSKG. We organize the analysis around two diagnostic questions. The scale study examines whether calibrated graph retrieval remains effective as the skill library becomes larger and more diverse. The component study examines how broad candidate induction, judge-assisted scoring, and counterfactual correction with state-gated publication contribute to the final retrieval graph. Throughout this section, R denotes ALFWorld success rate and *Steps* denotes the mean number of environment interactions; higher R and fewer steps are better.

5.1 Sensitivity to Skill Library Size

Larger skill libraries provide richer procedural coverage and create a stronger test of relation calibration, since retrieval must select useful procedural neighborhoods from a broader set of candidate relations. We therefore use the ALFWorld ID-140 scale records to compare CaSKG with GoS at 200, 500, 1,000, and 2,000 skills. The comparison covers MiniMax-M2.7 and Qwen3.5-397B-A17B. For MiniMax-M2.7, CaSKG assesses 500 candidate relations at 200–1,000 skills and expands the validation frontier to 2,000 relations at the 2,000-skill scale, matching the larger construction setting used at that scale.

Figure 3 visualizes the same scale comparison and highlights CaSKG's stable advantage under different library sizes. On MiniMax-M2.7, CaSKG improves success over GoS at every scale, with gains

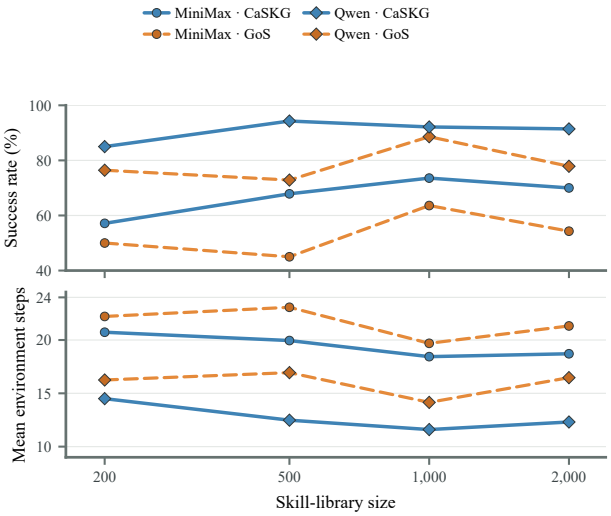

Figure 3: Skill-library-size sensitivity on ALFWorld ID-140 for MiniMax-M2.7 and Qwen3.5-397B-A17B. The upper panel reports task success rate and the lower panel reports mean environment steps for CaSKG and GoS at four library sizes.

of 7.14, 22.86, 9.97, and 15.71 percentage points from 200 to 2,000 skills. On Qwen3.5-397B-A17B, the corresponding gains are 8.57, 21.43, 3.54, and 13.57 points. The improvement is largest at 500 skills for both backbones, and CaSKG still adds 3.54 points at the strong 1,000-skill Qwen setting where GoS already reaches 88.60% success. This pattern indicates that CaSKG benefits from calibrated procedural structure across both moderate and large libraries. Its relative advantage is most visible when the graph contains rich procedural neighbors and the retrieval process can use edge calibration to focus propagation.

The step results provide a second view of the same mechanism. CaSKG uses fewer mean environment interactions than GoS at every tested scale for both backbones. For MiniMax-M2.7, the reductions are 1.48, 3.12, 1.25, and 2.61 steps; for Qwen3.5-397B-A17B, they are 1.75, 4.46, 2.55, and 4.16 steps. The success gains are therefore accompanied by lower interaction cost. This pattern is consistent with the role of the calibrated graph: it gives the agent a more focused procedural neighborhood, reducing exploratory and corrective actions after retrieval. MiniMax reaches its best observed CaSKG success at 1,000 skills, while Qwen reaches its best observed CaSKG success at 500 skills, showing that the most useful library size can depend on the backbone. Across these settings, CaSKG maintains the stronger success–step profile.

5.2 Component Analysis of Graph Construction

The component ablation examines the two design pressures behind CaSKG's graph construction. The first stage should recover a sufficiently broad set of candidate relations, including procedural dependencies that are difficult to capture from surface similarity alone. The second stage should select the relations that are reliable enough to influence retrieval. We evaluate four pipeline configurations

Table 3: Component ablation of CaSKG's graph-construction pipeline with MiniMax-M2.7 and the Skill1000 library on ALFWorld ID-140.

Variant	$\|C\|$	$\|F\|$	$\|E_{\text{pub}}\|$	R (%) ↑	Steps↓
Full CaSKG	9,937	500	3,292	**73.57**	**18.44**
Semantic-only	3,982	500	2,698	67.14	19.21
w/o LLM judge	9,753	500	3,188	71.43	18.79
Publish all candidates	9,937	0	9,937	71.43	18.74

$|C|$, $|F|$, and $|E_{\text{pub}}|$ denote the numbers of candidate, counterfactually assessed, and published relations, respectively. R is success rate (%), and Steps is the mean number of environment interactions; higher R and lower Steps are better.

under the Skill1000 setting with MiniMax-M2.7 on ALFWorld ID-140. All variants use Qwen3-Embedding-8B with 4,096-dimensional embeddings, share the same 140-task cohort, retrieval mode, 30-step limit, and downstream execution protocol. Full CaSKG and the two first-stage ablations each assess 500 candidate relations. The full-candidate variant publishes the complete candidate graph, providing a direct comparison between selective publication and maximum graph density.

Table 3 shows selective publication gives the strongest configuration. Full CaSKG reaches 73.57% success with 18.44 mean steps while publishing 3,292 of 9,937 candidate relations. The full-candidate variant publishes all 9,937 relations and reaches 71.43% success with 18.74 mean steps. Because both configurations start from the same candidate set, the comparison highlights the value of counterfactual correction and state-gated publication. The selected graph provides enough relational coverage for retrieval while keeping propagation concentrated on higher-confidence procedural relations.

The semantic-only variant further highlights the role of multi-signal candidate induction. It constructs 3,982 candidate relations and publishes 2,698 relations, reaching 67.14% success with 19.21 mean steps. Full CaSKG raises success by 6.43 percentage points and lowers the mean step count by 0.77 steps. The improvement shows that non-semantic evidence supplies useful procedural neighbors whose value depends on workflow role, interface compatibility, co-occurrence, or repair evidence beyond lexical similarity alone. This ablation evaluates the multi-signal candidate stage as an integrated module and supports its contribution to downstream execution.

The no-judge variant shows that the optional judge signal provides additional calibration on top of deterministic construction signals. The candidate and published graph sizes remain close to the full configuration, while full CaSKG adds 2.14 percentage points of success and reduces the mean step count from 18.79 to 18.44. This pattern suggests that the judge signal refines borderline relation scores that affect which edges survive publication and how strongly they influence propagation. The comparison treats the judge as part of the complete graph-construction pipeline and shows its incremental contribution under the reported configuration.

The component results give a sharper explanation of the main comparison. CaSKG benefits from both sides of the pipeline: multi-signal construction supplies a candidate graph broad enough to include non-lexical procedural dependencies, and counterfactual state-gated publication turns that broad candidate set into a confidence-weighted retrieval graph. The LLM judge further refines relation scoring under this configuration. Together, the ablations support

the design principle of CaSKG: retrieve from a graph that is broad at the candidate stage and selective at the publication stage.

6 Conclusion

This paper studies skill retrieval for LLM agents as the problem of exposing compact and executable procedural context from a large reusable skill library. CaSKG addresses this problem by separating high-recall association discovery from edge-confidence calibration. It constructs a directed candidate graph from heterogeneous skill signals, evaluates selected relations with direction-conditioned textual counterfactual probes, and publishes a state-filtered weighted graph for task-conditioned retrieval. Across ALFWorld and ScienceWorld, CaSKG achieves the best reported task score in all twelve model–benchmark settings. Relative to GoS, the six-model macro-average score increases from 72.62 to 80.50 on ScienceWorld and from 80.01% to 86.79% on ALFWorld, while using fewer observed environment interactions. Task-type, trajectory, library-scale, and ablation analyses are consistent with calibrated relations helping preserve prerequisites, state-changing actions, verification routines, and completion steps within the retrieved skill bundle.

Overall, these findings suggest that edge-confidence calibration is a useful design principle for scalable agent memory. By combining broad candidate recall with selective graph publication, CaSKG retrieves compact skill context while preserving the operational structure needed to execute complex tasks.

Ethics and Privacy Statement

This work studies skill retrieval for LLM agents in simulated ALFWorld and ScienceWorld environments using reusable skill libraries and aggregate task outcomes; it does not collect personal data, involve human subjects, or infer sensitive attributes. The main broader-impact consideration is that more reliable procedural retrieval can make autonomous agents more capable, which benefits reproducible tool use and controlled task execution but should still be deployed with task-appropriate safeguards when connected to external tools or real-world environments. CaSKG is an offline retrieval-graph construction method that does not introduce a new action policy or expand the agent's permissions, and the experiments are conducted within benchmark-defined environments and evaluation protocols.

References

[1] Aili Chen, Aonian Li, Baichuan Zhou, Bangwei Gong, Binyang Jiang, Boji Dan, Changqing Yu, Chao Wang, Cheng Ma, Cheng Zhong, et al. 2026. The minimax-m2 series: Mini activations unleashing max real-world intelligence. *arXiv preprint arXiv:2605.26494* (2026).

[2] Yanping Chen, Weijie Shi, Wen Yang, and Jiajie Xu. 2026. Task Decomposition-Guided Reranking for Adaptive Agent Skill Retrieval. *arXiv preprint arXiv:2607.06283* (2026).

[3] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[4] Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems* 37 (2024), 59532–59569.

[5] Taher H Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th international conference on World Wide Web*. 517–526.

[6] Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng Lyu, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, et al. 2023. Cladder: Assessing causal reasoning in language models. *Advances in Neural Information Processing Systems* 36 (2023), 31038–31065.

[7] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*. 6769–6781.

[8] Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. 2023. Causal reasoning and large language models: Opening a new frontier for causality. *arXiv preprint arXiv:2305.00050* (2023).

[9] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33 (2020), 9459–9474.

[10] Jiaxuan Li, Lang Yu, and Allyson Ettinger. 2023. Counterfactual reasoning: Testing language models' understanding of hypothetical scenarios. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 804–815.

[11] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A comprehensive benchmark for tool-augmented llms. In *Proceedings of the 2023 conference on empirical methods in natural language processing*. 3102–3116.

[12] Dawei Liu, Zongxia Li, Hongyang Du, Xiyang Wu, Shihang Gui, Yongbei Kuang, and Lichao Sun. 2026. Graph-of-Skills: Dependency-Aware Structural Retrieval for Massive Agent Skills. *arXiv preprint arXiv:2604.05333* (2026).

[13] Xukun Liu, Zhiyuan Peng, Xiaoyuan Yi, Xing Xie, Lirong Xiang, Yuchen Liu, and Dongkuan Xu. 2024. Toolnet: Connecting large language models with massive tools via tool graph. *arXiv preprint arXiv:2403.00839* (2024).

[14] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842* (2023).

[15] Moonshot AI. 2026. Kimi K2.6: Advancing Open-Source Coding. Accessed: 2026-08-25. https://www.kimi.ai/blog/kimi-k2-6

[16] OpenAI. 2026. GPT-5.6 System Card. Accessed: 2026-08-25. https://deploymentsafety.openai.com/gpt-5-6/change-log

[17] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2024. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems* 37 (2024), 126544–126565.

[18] Judea Pearl et al. 2003. Causality: models, reasoning, and inference. *Econometric Theory* 19, 675-685 (2003), 46.

[19] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2024. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, Vol. 2024. 9695–9717.

[20] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.

[21] Chamod Samarajeewa, Daswin De Silva, Evgeny Osipov, Damminda Alahakoon, and Milos Manic. 2024. Causal reasoning in large language models using causal graph retrieval augmented generation. In *2024 16th International Conference on Human System Interaction (HSI)*. IEEE, 1–6.

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

[23] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. 2021. Toward causal representation learning. *Proc. IEEE* 109, 5 (2021), 612–634.

[24] Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. 2025. Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*. 24497–24524.

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

[26] Qwen Team. 2026. Qwen3.5: Towards Native Multimodal Agents. https://qwen.ai/blog?id=qwen3.5

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

[28] Nengbo Wang, Xiaotian Han, Jagdip Singh, Jing Ma, and Vipin Chaudhary. 2025. Causalrag: Integrating causal graphs into retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*. 22680–22693.

[29] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader?. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 11279–11298.

[30] Tianle Xia, Lingxiang Hu, Yiding Sun, Ming Xu, Lan Xu, Siying Wang, Wei Xu, and Jie Jiang. 2026. Grasp: Graph-structured skill compositions for llm agents. *arXiv preprint arXiv:2604.17870* (2026).

[31] Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348* (2026).

[32] Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, and Jian Zhang. 2023. On the tool manipulation capability of open-source large language models. *arXiv preprint arXiv:2305.16504* (2023).

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

[34] Paul Youssef, Christin Seifert, Jörg Schlötterer, et al. 2024. LLMs for generating and evaluating counterfactuals: A comprehensive study. In *Findings of the association for computational linguistics: EMNLP 2024*. 14809–14824.

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

[36] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems* 36 (2023), 46595–46623.