SKILLDES: DEPENDENCY- AND EVIDENCE-AWARE SKILL-SET SELECTION FOR COMPOSITE TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Composite requests require complementary capabilities, yet ranking individual descriptions inherently fails to recover the required skill set. This mismatch is severely amplified by long documentation, uneven evidence distributions, and strict topological dependencies between subtasks. To address these challenges, we present SkillDES, a comprehensive retrieve-and-select framework that explicitly couples semantic task decomposition with dual-view candidate recall. To overcome semantic dilution in long documents, we introduce Long-Context Evidence Aggregation (LCEA), dynamically extracting bounded, node-specific complementary evidence for robust reranking. Furthermore, our dependency-aware beam search algorithm jointly balances skill compatibility, dynamic cardinality, and full capability coverage. Extensive evaluations across five diverse benchmarks demonstrate the effectiveness and robustness of our approach. Our system achieves a state-of-the-art complete-system SS-F1 of 68.4, representing a substantial improvement over the strongest baseline score of 51.2, while consistently outperforming existing retrieval, adaptation, and reranking methods and establishing a rigorous standard for composite capability selection. Our code is available at `https://anonymous.4open.science/r/SkillDES`.

1 INTRODUCTION

Agent systems increasingly acquire and extend capabilities through reusable skills. An agent skill packages task-specific instructions and metadata, optionally accompanied by executable scripts, reference material, examples, or other supporting resources (Zhang et al., 2025a). Its documentation therefore helps an agent determine when, why, and how to invoke and use that capability effectively; the skill is not simply the document itself. We study the skill selection layer: each catalog item is associated with a stable identifier and available capability documentation that describes its intended functionality and usage conditions. Tool-oriented benchmarks instantiate this general interface without implying that every individual API entry constitutes a complete, self-contained agent-skill package. A composite request often requires several complementary and mutually dependent skills to be recovered together. In the schematic travel example in Figure 1, forecasting the weather and booking a hotel still leave a missing prerequisite: first finding a suitable hotel that satisfies the user's constraints. A globally relevant ranked list can therefore omit an essential capability needed to complete the task, while independently retrieving one skill for every subtask may introduce redundant or duplicate selections when a single skill can be reused across multiple steps. The example illustrates the distinction between individual skill relevance and complete, dependency-aware set recovery.

Existing systems retrieve APIs, generate tool identifiers, or plan tool use (Qin et al., 2024; Patil et al., 2024; Wang et al., 2025b; Fang et al., 2026). Retrieval benchmarks further expose compositional requirements (Shi et al., 2025; Wang et al., 2025a; Kang et al., 2026). Their common difficulty is that a high-quality ranking still leaves a decision: how many distinct capabilities are needed, and which combination covers the request? Set selection must balance missing requirements against unnecessary additions. Long documentation makes this decision harder. Prerequisites and narrow capabilities may appear far from a document's beginning. Truncation can remove them, while repeated broadly related passages can dominate a limited evidence budget. Long-context studies establish that access to text does not guarantee effective use of its relevant content (Bai et al., 2024; Liu et al., 2024); here, this motivates selecting task-specific evidence within candidate descriptions.

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.

To address these challenges, SkillDES separates candidate access, supporting evidence, and set construction. A semantic task graph exposes subqueries and dependency constraints. Global and local retrieval form a shared candidate pool, and Long-Context Evidence Aggregation (LCEA) selects relevant, nonredundant passages for each subtask–candidate pair. Evidence-conditioned ranking provides an anchor set; cardinality control and dependency-aware beam search coordinate alternative assignments. A development-selected router decides whether to retain the anchor or accept the structured proposal.



Figure 1: Schematic travel-planning example. Global top-k retrieval can omit a complementary skill; decomposition exposes the missing requirement.

Contributions. We formulate a comprehensive retrieve-and-select framework to tackle variable-cardinality skill-set selection for composite tasks. Our approach seamlessly couples semantic task decomposition with a dual-view retrieval mechanism to explicitly capture prerequisite skills. To overcome evidence imbalance and semantic dilution prevalent in long tool documentation, we introduce Long-Context Evidence Aggregation (LCEA), which dynamically extracts node-specific, complementary evidence under a strict computational budget. Furthermore, we propose a dependency-aware beam search algorithm that explicitly enforces topological constraints while balancing skill compatibility and overall capability coverage. Extensive evaluations across five diverse benchmarks demonstrate the consistent improvements of our approach, outperforming existing adaptation and reranking baselines.

2 Related Work

Tool Retrieval and Composition. ToolLLM couples API retrieval with tool-use learning; Gorilla grounds API generation in documentation; AnyTool organizes large tool collections through hierarchical retrieval and self-reflection (Qin et al., 2024; Patil et al., 2024; Du et al., 2024). ToolGen generates dedicated tool identifiers, while ToolWeaver represents tools through hierarchical codes that incorporate collaborative semantics (Wang et al., 2025b; Fang et al., 2026). These methods establish alternatives to isolated nearest-neighbor selection, but retrieving a deduplicated ID set differs from generating arguments or carrying out an interaction. More direct retrieval work clarifies the relevant comparison. Re-Invoke combines synthetic indexing queries, intent extraction, and multiview similarity without supervised retriever training (Chen et al., 2024b). ToolRerank adapts truncation and uses tool-library hierarchy to balance concentration and diversity (Zheng et al., 2024). ToolQP learns iterative query planning for compositional tool retrieval (Fang & Glass, 2026). ToolRet and SkillRet provide retrieval-focused benchmarks (Shi et al., 2025; Kang et al., 2026). Compositional Skill Routing also combines decomposition, retrieval, and dependency-aware planning (Gao, 2026). We significantly advance this paradigm: rather than serving as just another decomposition-based retriever, our approach couples node-specific long-document evidence with cardinality-limited, dependency-constrained set construction. Furthermore, unlike methods such as ToolACE and Tool-REX that rely on altering underlying data or expanding documentation (Liu et al., 2025; Lu et al., 2026), our method dynamically extracts and optimizes the selection of existing evidence.

Embedding and Multi-View Matching. NV-Embed, few-shot text embedders, and KaLM-Embedding-V2 study generalist embedding training and adaptation (Lee et al., 2025; Li et al., 2025; Zhao et al., 2026). Complementary work includes multilingual E5's contrastive recipe, M3-

Embedding's dense training, and task-specific adapters in jina-embeddings-v3 (Wang et al., 2024; Chen et al., 2024a; Sturua et al., 2024). LLM-Embedder learns multiple retrieval functions from language-model feedback, while empirical assessments compare LLM backbones across retrieval conditions (Zhang et al., 2024; Luo et al., 2024). Crucially, our architecture efficiently unifies global, token, and pair branches within a single MPNet backbone; these act as highly efficient training views that completely eliminate the computational overhead of maintaining multiple catalog-wide inference networks. BRIGHT emphasizes reasoning-intensive relevance, and theoretical analysis identifies dimensional restrictions on embedding-representable retrieval subsets (Su et al., 2025; Weller et al., 2026). While these observations motivate evaluating more than nearest-neighbor similarity, our decoder explicitly overcomes these dimensional limitations through structured, dependency-aware constraints. Datastore construction is also consequential (Lyu et al., 2026); we therefore rigorously distinguish catalog composition, candidate coverage, and candidate-internal selection in all comparisons.

Long-Document Evidence. LongBench and Lost in the Middle examine long-context understanding and position-sensitive evidence use (Bai et al., 2024; Liu et al., 2024). While late chunking preserves surrounding context when producing chunk embeddings (Günther et al., 2024), our LCEA module takes a decisive step further by dynamically optimizing which independently encoded blocks to retain for a specific node. RAPTOR retrieves across hierarchical summaries, while HippoRAG and G-Retriever leverage graph-organized knowledge (Sarthi et al., 2024; Jiménez Gutiérrez et al., 2024; He et al., 2024). Whereas those structures rely on static source-content organization, our approach constructs a dynamic, query-specific task DAG whose edges strictly govern the assignment order. RECOMP and Provence learn context compression or pruning, LLMLingua-2 compresses task-agnostic prompts, and BTR compresses token representations (Xu et al., 2024; Chirkova et al., 2025; Pan et al., 2024; Cao et al., 2024). Self-RAG learns retrieval decisions and RankRAG combines context ranking with generation (Asai et al., 2024; Yu et al., 2024). In contrast, LCEA retains a bounded collection of original blocks and exposes precise evidence statistics directly to a supervised scorer, ensuring superior context retention.

3 METHOD

Given a query q and a skill library $\mathcal{S}$, SkillDES returns a candidate ranking $\pi(q)$ and a variable-size skill set $\widehat{G}(q)$. Skills are retrieved via their documentation; we deliberately isolate offline skill selection to rigorously evaluate retrieval capability without the confounding noise of execution failures. Our pipeline integrates global and subtask retrieval, budgeted evidence aggregation, supervised reranking, and dependency-constrained set decoding. Training is highly modular, utilizing specialized, independently optimizable membership, cardinality, and compatibility models.

Figure 2 distinguishes three complementary roles: representation learning provides scalable candidate access, task decomposition exposes necessary prerequisite requirements that a global query may obscure, and document-level evidence provides robust support for candidate selection. Crucially, while the matching branches on the left provide rich multi-view supervision during training, only the unified global encoder searches the full library at inference, ensuring highly efficient and scalable deployment.

3.1 MULTI-BRANCH SUPERVISION FOR CANDIDATE RETRIEVAL

Let $h_u(x)$ and $m_u(x)$ be MPNet token states and attention masks. The shared global encoder is

$$g(x) = \mathrm{Norm}\left(\mathrm{LN}\left(\mathrm{GELU}\left(W_g \frac{\sum_u m_u(x) h_u(x)}{\sum_u m_u(x) + \epsilon} + b_g\right)\right)\right). \tag{1}$$

Training intelligently combines global cosine similarity, token-level late interaction, and an MLP over paired global representations. Their score $S(q, s)$ uses learned softmax mixture weights. For query q_i, let $\mathcal{P}_i$ contain its annotated positive skills and $\mathcal{J}_i$ its scored candidates. The multi-positive objective is

$$\begin{aligned} \mathcal{L}_{\mathrm{con}} &= -\frac{1}{|\mathcal{Q}_B|} \sum_i \log \frac{\sum_{j \in \mathcal{P}_i} \exp(S_{ij}/\tau)}{\sum_{j \in \mathcal{J}_i} \exp(S_{ij}/\tau)}, \\ \mathcal{L}_{\mathrm{ret}} &= \lambda_{\mathrm{con}} \mathcal{L}_{\mathrm{con}} + \lambda_{\mathrm{mar}} \mathcal{L}_{\mathrm{mar}} + \lambda_{\mathrm{bce}} \mathcal{L}_{\mathrm{bce}}. \end{aligned} \tag{2}$$

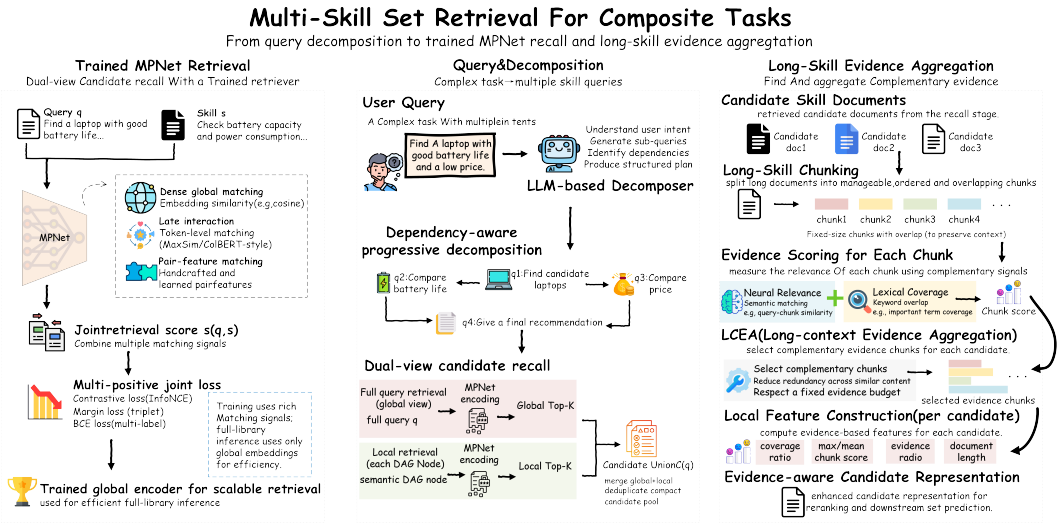


Figure 2: Overview of the framework: global and subtask retrieval supply complementary candidates, bounded document evidence supports reranking, and dependency-aware decoding selects the final skill set.

The margin loss explicitly separates the strongest positive and negative, while class-weighted binary cross-entropy systematically supervises pair membership. Non-gold hard negatives significantly strengthen training robustness. The auxiliary branches gracefully share the encoder, completely eliminating full-library inference overhead while fully preserving multi-view matching fidelity. Appendix B.2 provides their comprehensive definitions and exact losses.

3.2 Task Decomposition and Dual-View Recall

Global scores $R_0(q, s) = g(q)^\top g(s)$ first retrieve coarse skill summaries. Conditioned on these summaries and q, the decomposition model produces a semantic DAG $D(q) = (V, E)$. Node $v_t = (q_t, \kappa_t)$ contains a subquery and required capability; predecessor fields encode semantic prerequisites. This graph robustly guides selection, structurally anchoring the process rather than relying on unconstrained LLM generation for final skill identifiers.

The shared encoder processes $q_t \| \kappa_t$ for local retrieval, and q_t and κ_t separately for downstream semantic and capability matching. With $R_t(s) = g(q_t \| \kappa_t)^\top g(s)$, the pool is

$$C(q) = \mathrm{Top}_{K_g}(R_0(q, \cdot)) \cup \bigcup_{t=1}^{m} \mathrm{Top}_{K_l}(R_t(\cdot)), \qquad m = |V|. \tag{3}$$

Global recall reliably preserves the complete request intent, while local recall captures critical fine-grained requirements that might otherwise be overshadowed. Deduplication strictly bounds the pool by $\min(|\mathcal{S}|, K_g + mK_l)$. Training pools explicitly include missed gold skills for robust supervision; development and test pools never receive this augmentation. This meticulously constructed pool establishes a highly comprehensive foundation, ensuring all necessary capabilities are available for downstream exact recovery.

3.3 Budgeted Long-Context Evidence Aggregation

Standard truncated document representations inevitably omit decisive evidence. Our Long-Context Evidence Aggregation (LCEA) effectively solves this by dynamically selecting complementary passages from each retrieved skill. Structural splitting, overlapping windows, and position-spanning retention form chunks $\mathcal{B}_s = \{b_{sk}\}_{k=1}^{M_s}$ (Appendix B.4). Selection is precisely conditioned on each node–skill pair, allowing distinct passages of a single skill to independently support different subtask requirements.

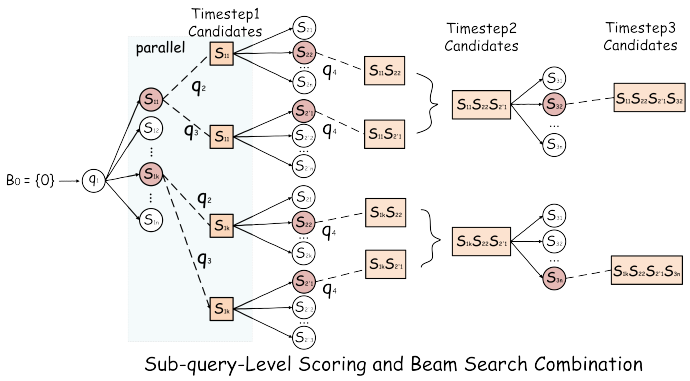


Figure 3: Dependency-aware multi-state beam search retains alternative partial skill sets while processing the task graph, completely eliminating premature, irreversible local choices.

Let $u_t = g(q_t \| \kappa_t)$, $e_{sk} = g(b_{sk})$, and let T_t, W_{sk} be subquery and chunk term sets. Relevance strategically combines neural matching and lexical coverage:

$$n_{tsk} = u_t^\top e_{sk}, \quad a_{tsk} = \frac{|T_t \cap W_{sk}|}{|T_t| + \epsilon}, \quad r_{tsk} = \alpha_n z_{ts}(n_{tsk}) + \alpha_a z_{ts}(a_{tsk}). \tag{4}$$

Here z_{ts} standardizes each channel within the chunks of skill s for node t. Starting from an empty evidence set, LCEA optimally selects

$$k_h^* \in \arg\max_{k \notin A_{ts}^{(h)}} [r_{tsk} - \lambda_\rho \rho_s(k, A_{ts}^{(h)})], \qquad A_{ts}^{(h+1)} = A_{ts}^{(h)} \cup \{k_h^*\}, \tag{5}$$

until budget B or the chunk inventory is exhausted. The penalty systematically combines lexical and positional redundancy, strictly ensuring that redundant support yields to complementary passages. This mechanism elegantly bounds the context footprint while capturing maximal capability information (full specification in Appendix B.5).

3.4 Evidence-Conditioned Ranking and Cardinality Control

The candidate representation holistically integrates multidimensional information regarding semantic relevance, selected evidence, cross-subtask support, document coverage, and evidence distribution. It preserves both absolute matching strength and a query-relative view of each candidate. A class-balanced regularized logistic model produces a highly calibrated membership logit $h_s = w^\top x_s + b$, which is then fused with the base retrieval score b_s:

$$F_s = (1 - \alpha) z_{C(q)}(b_s) + \alpha z_{C(q)}(h_s), \qquad \pi(q) = \mathrm{argsort}_{s \in C(q)} F_s. \tag{6}$$

Appendix B.6 specifies the precise feature map and fitting objective. The expertly ranked candidates yield an anchor set G_{a} through a development-optimized fixed cutoff or a bounded relative-margin rule.

Crucially, our framework natively accommodates skill reuse across multiple steps, meaning subtask count need not rigidly dictate the distinct-skill count. A specialized random-forest classifier predicts the optimal cardinality from anchor size, task structure, score separation, and document-length statistics. Development selection seamlessly adopts either this robust prediction or the anchor size as $\widehat{r}$. The decoder elegantly enforces $\widehat{r}$ as an upper bound, granting the system the dynamic flexibility to return the precise optimal combination without forcing unnecessary additions (Appendix B.7).

3.5 Dependency-Constrained Set Decoding

Figure 3 illustrates how assignment decisions systematically interact: parallel requirements can efficiently share an earlier capability, while downstream requirements accurately depend on several preceding choices. The diagram visually summarizes our robust candidate combination strategy;

the formal decoder effectively completes one ready node per extension and seamlessly deduplicates reused skills in the selected set.

A state $P = (\mathcal{A}, S, \mathbf{c}, \Gamma, O)$ meticulously tracks assignments, distinct skills, soft coverage, score, and completed-node order. An extension systematically assigns s to an unfinished node whose predecessors are complete, safely subject to $|S \cup \{s\}| \leq \widehat{r}$. Our rich expansion options combine evidence, semantic, capability, and global-quality shortlists with already selected skills, actively encouraging optimal skill reuse.

The extension score cleanly factorizes into three distinct roles:

$$\bar{\Delta}(P, t, s) = \Phi_{\text{evid}}(t, s) + \Phi_{\text{dep}}(P, t, s) + \Phi_{\text{set}}(P, s). \tag{7}$$

Evidence support robustly merges local matching and overarching candidate quality. Dependency coordination evaluates predecessor compatibility, downstream support, and capability diversity. Set complementarity explicitly rewards total capability coverage while penalizing both redundancy and the unnecessary introduction of additional distinct skills. A symmetric pair model supplies accurate compatibility scoring, while strict directionality is reliably enforced by the DAG. This mathematically sound factorization preserves all underlying coefficients. Combining reuse and new-skill terms elegantly removes only a common score offset at equal assignment depth.

Beam search securely retains at most W high-quality states per round. A highly calibrated, label-free utility U dynamically compares the resulting set G_{b} with the anchor:

$$\widehat{G}(q) = \begin{cases} G_{\text{b}}, & U(G_{\text{b}}) - U(G_{\text{a}}) \geq \eta \text{ and the size-difference rule holds,} \\ G_{\text{a}}, & \text{otherwise.} \end{cases} \tag{8}$$

All fitted modules are rigorously trained on standard labels, with configuration tuning strictly isolated to development data. By keeping ranking, routed set, and beam trace as clearly delineated outputs, our search explicitly guarantees strict topological feasibility relative to the generated task graph, providing a rigorously structured skill-set proposal primed for execution. Detailed algorithmic breakdowns and derivations are provided in Appendix B.

4 Experiments

4.1 Experiments Setup

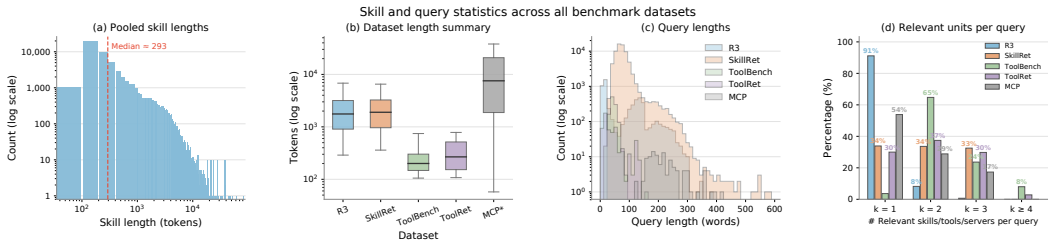


Figure 4: Benchmark document lengths, query lengths, and relevant-set cardinalities.

Datasets. We conduct comprehensive evaluations across five diverse and representative benchmarks: R3-Skill (Wang et al., 2026b), SkillRet (Kang et al., 2026), ToolBench (Qin et al., 2024), ToolRet-Toolink (ToolRet) (Shi et al., 2025), and MCP-Bench (MCP) (Wang et al., 2026a). As shown in Figure 4, the distributions of source document lengths, query lengths, and relevant target-set cardinalities vary significantly across datasets, making evidence budgets and distinct skill counts critical factors. Comprehensive length histograms and earlier processed-split details are provided in Appendix A.

Baselines. We consider four groups of baselines. Traditional retrieval is represented by BM25 (Robertson & Zaragoza, 2009), while pretrained retrievers include Qwen3-Embedding (Zhang et al., 2025b), R3-Skill retriever (Wang et al., 2026b), SkillRet retriever (Kang et al., 2026), ToolBench

Table 1: Final skill-set selection (%) across five benchmarks and four baseline groups. SS-F1 measures set overlap, while ESM requires exact skill-set recovery. Bold with † and underlining denote the best and second-best entries; † does not indicate statistical significance.

Method	R3-Skill		SkillRet		ToolBench		ToolRet		MCP		Average	
	SS-F1	ESM	SS-F1	ESM	SS-F1	ESM	SS-F1	ESM	SS-F1	ESM	SS-F1	ESM
Traditional retrieval												
BM25	21.6	6.7	34.7	4.7	14.0	1.3	14.1	5.8	34.3	4.8	23.7	4.7
Pretrained retrievers												
Qwen3 embedding 8B	44.4	10.7	37.7	4.6	32.4	9.3	39.1	15.2	81.4	**65.2**†	47.0	21.0
R3 embedding	51.2	22.7	54.1	25.3	26.7	4.0	24.1	5.1	74.3	43.1	46.1	20.0
SkillRet	46.1	20.1	56.7	25.1	30.5	6.8	26.4	3.7	80.0	28.5	47.9	16.8
ToolBench retriever	7.2	2.7	18.1	5.7	<u>39.5</u>	2.7	21.2	3.6	47.5	4.9	26.7	3.9
ToolRet E5-large	25.1	8.0	39.6	9.4	22.4	5.3	27.2	5.9	65.7	28.6	36.0	11.4
ToolRet BGE-large	18.3	2.8	41.6	7.3	39.0	11.3	19.9	1.9	65.8	4.7	36.9	5.6
COLT	19.6	4.1	27.2	3.3	23.2	1.4	25.1	7.7	34.4	14.3	25.9	6.2
Reranking systems												
SkillRouter	<u>53.9</u>	**29.3**†	<u>63.1</u>	<u>28.0</u>	34.1	5.4	28.5	3.4	21.4	5.2	40.2	14.3
R3	42.4	17.3	49.7	16.7	32.5	7.4	27.0	6.0	23.5	5.3	35.0	10.5
SkillRet + reranker	49.9	18.7	59.1	24.7	30.0	5.5	28.3	3.3	74.4	48.1	48.3	20.1
Reranker 0.6B	50.4	26.5	47.6	22.8	32.6	<u>12.5</u>	42.0	17.4	<u>83.4</u>	51.1	<u>51.2</u>	<u>26.1</u>
Adaptation methods												
Re-Invoke	37.3	14.8	42.8	15.8	27.1	3.6	36.6	12.7	64.8	15.6	41.7	12.5
ToolDreamer	13.9	2.9	20.4	2.8	21.3	1.2	<u>51.0</u>	<u>38.9</u>	58.6	14.9	33.0	12.1
ToolQP	38.0	16.5	43.5	17.5	27.2	6.5	39.9	21.5	65.9	35.5	42.9	19.5
COLT-style	20.8	12.5	30.4	11.5	19.0	8.7	32.3	10.2	47.1	12.5	29.9	11.1
MFTR-style	22.1	10.3	37.5	12.1	20.5	7.2	22.9	9.8	60.0	14.5	32.6	10.8
Tool-REX	26.8	8.2	52.5	15.3	18.1	1.6	25.2	23.4	54.3	14.6	35.4	12.6
ToolQP-REX	38.1	19.1	52.6	22.4	27.3	8.4	39.8	15.1	54.4	25.5	42.4	18.1
Our system												
SkillDES (Ours)	**57.5**†	<u>28.0</u>	**63.7**†	**32.0**†	**54.5**†	**21.3**†	**76.8**†	**48.1**†	**89.5**†	<u>63.1</u>	**68.4**†	**38.5**†

retriever (Qin et al., 2024), and ToolRet retriever (Shi et al., 2025). Reranking methods include SkillRouter (Zheng et al., 2026), R3 (Wang et al., 2026b), SkillRet with reranking (Kang et al., 2026), and Qwen3-Reranker (Zhang et al., 2025b). Adaptation-based baselines include COLT (Qu et al., 2024), Re-Invoke (Chen et al., 2024b), ToolDreamer (Sengupta et al., 2026), ToolQP (Fang & Glass, 2026), MFTR (Tang et al., 2026), and Tool-REX (Lu et al., 2026).

Metrics. We assess retrieval and ranking performance using Recall@K, MRR@K, MAP@K, and nDCG@K evaluated on ordered candidate IDs. For final skill-set selection, we report Exact Skill-Set Match (ESM) and Skill-Set F1 (SS-F1) . Given a gold set G_i and a predicted set $\widehat{G}_i$, the metrics are defined as:

$$\text{ESM} = \frac{1}{n}\sum_i \mathbb{I}[\widehat{G}_i = G_i], \qquad \text{SS-F1} = \frac{1}{n}\sum_i \frac{2|G_i \cap \widehat{G}_i|}{|G_i| + |\widehat{G}_i|}. \tag{9}$$

ESM strictly requires identical identifier sets with no omissions or additions. SS-F1 calculates the harmonic mean of set precision and recall to credit partial overlaps. Ranking and routed-set predictions are evaluated independently, as altering a selected set does not inherently change its ranking. Specific cutoffs, candidate coverage, and oracle bounds are defined in Appendix D.

Implementation. Our architecture utilizes a dataset-adapted MPNet with a 128-dimensional global projection alongside Qwen3-8B for query decomposition (Yang et al., 2025). The candidate pool is constructed from the global top-64 and top-16 candidates per node. We encode up to twelve retained blocks per document, with LCEA selecting a maximum of four blocks per node–skill pair. Hyperparameters including reranking thresholds, cardinality estimates, beam widths, and routing configurations are tuned solely on training or development sets. Complete configuration settings are documented in Appendix C.

4.2 Main Comparison

Table 1 presents the evaluation of our framework against four major baselines across five benchmarks. Our complete system establishes a new state-of-the-art in multi-skill selection, achieving SS-F1 scores of 57.5 on R3-Skill, 63.7 on SkillRet, 54.5 on ToolBench, 76.8 on ToolRet, and 89.5 on MCP. These results yield an average SS-F1 of 68.4, outperforming the strongest competitive methods across every dataset. We observe substantial absolute gains; for example, our method outperforms the strongest respective baseline by 15.0 points on ToolBench and 25.8 points on ToolRet. Beyond harmonic overlap, our method demonstrates strong precision in Exact Skill-Set Match (ESM), which strictly penalizes omitted or extraneous skills. We achieve the highest ESM on SkillRet at 32.0, ToolBench at 21.3, and ToolRet at 48.1, culminating in the highest average ESM of 38.5. This strict exact recovery highlights the effectiveness of our dependency-constrained decoding and adaptive cardinality control in preventing redundant skills while satisfying subtask prerequisites.

The interplay between SS-F1 and ESM further highlights the robustness of our composite capability selection. On R3-Skill, our system achieves an SS-F1 of 57.5 against SkillRouter's 53.9, delivering better overall capability coverage, even though SkillRouter marginally leads in ESM with 29.3 compared to our 28.0. A similar dynamic occurs on MCP, where our method reaches an SS-F1 of 89.5 over the Qwen3 embedding baseline at 81.4, demonstrating stronger set recovery despite Qwen3 holding a slight edge in exact ID match at 65.2 against our 63.1. These distinctions show that partial coverage (SS-F1) and exact recovery (ESM) evaluate different dimensions of retrieval success, while our framework maintains a strong balance between them—providing downstream agents with functionally complete and minimally redundant skill sets.

4.3 Ablation Studies

To rigorously validate our architectural design, we perform extensive ablation studies across all five benchmarks. Table 2 demonstrates that each proposed module contributes to the framework's state-of-the-art performance, with the complete system consistently achieving the highest overall effectiveness.

Table 2: Component ablation results measuring SS-F1 (%) when removing individual framework modules.

Model	R3-Skill	SkillRet	ToolBench	ToolRet	MCP
Ours (Full)	**57.5**	**63.7**	**54.5**	**76.8**	**89.5**
w/o Dependency edges	56.4	62.5	53.2	74.9	88.1
w/o Chunk evidence	56.2	62.4	54.1	76.3	85.2
w/o LCEA branch	53.8	58.9	39.5	68.4	63.6
w/o Learned reranker	52.4	57.1	53.8	68.2	67.4
w/o Adaptive cardinality	55.6	62.1	53.9	64.8	88.6
w/o Multi-state beam	54.3	61.2	50.4	71.6	85.7

Impact of Evidence Aggregation and Representation. Our Long-Context Evidence Aggregation (LCEA) branch and learned reranker prove strictly necessary for handling complex tool documentation. Removing the LCEA branch precipitates severe performance degradation, most notably dropping SS-F1 by 25.9 absolute points on MCP and 15.0 points on ToolBench. This supports that extracting bounded, complementary chunk evidence is critical to overcoming the semantic dilution inherent in long document processing. Similarly, the learned reranker plays a pivotal role across diverse environments; without it, performance on the highly complex MCP benchmark plummets from 89.5 to 67.4. The necessity of this module extends to retrieval-heavy tasks as well, evidenced by a significant 8.6-point drop on ToolRet and a 6.6-point decline on SkillRet. Across the broader spectrum of real-world, multi-server tasks, dynamic evidence-conditioned reranking is an indispensable driver of our framework's superior accuracy.

Role of Structural Constraints and Decoding. The ablation results further underscore the absolute necessity of our dependency-aware beam search and dynamic cardinality estimation. Disabling adaptive cardinality inflicts a massive 12.0-point SS-F1 penalty on ToolRet, proving that rigid, pre-

Table 3: Performance comparison between the full decomposition system and a matched no-decomposition control. Results report SS-F1 (%) and candidate-pool recall (%) .

Dataset	Control SS-F1	No-decomp. SS-F1	Delta (pp)	Control recall	No-decomp. recall
R3-Skill	**57.5**	54.3	3.2	**72.8**	58.0
SkillRet	**63.7**	61.5	2.2	**87.2**	56.4
ToolBench	**54.5**	49.9	4.6	**82.0**	72.0
ToolRet	**76.8**	75.9	0.9	**97.2**	92.3
MCP	**89.5**	64.2	25.3	**92.7**	89.4

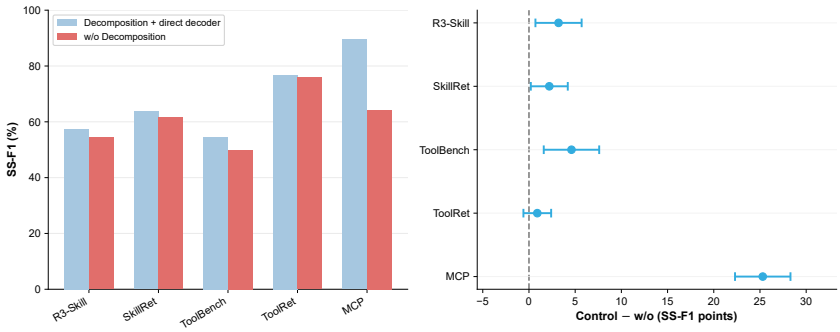


Figure 5: Matched decomposition control comparing final SS-F1 with the no-decomposition variant. Absolute differences quantify the effect of task decomposition under the same evaluation setting.

defined selection sizes inherently fail to capture variable-length target sets. Furthermore, replacing the multi-state beam search with greedy decoding substantially weakens performance, lowering ToolBench by 4.1 points and MCP by 3.8 points. Removing the topological dependency edges altogether triggers performance drops across all five benchmarks. Together, these findings robustly confirm that delaying irreversible local choices through multi-hypothesis tracking, strictly guided by dependency constraints, is fundamental to recovering accurate and complete skill sets.

4.4 The Multiplier Effect of Semantic Task Decomposition

Semantic task decomposition serves two roles in our framework: expanding candidate coverage and providing structural guidance for downstream decoding. Table 3 isolates this effect by comparing the full framework with a matched direct-decoder control. On SkillRet, dual-view recall increases candidate-pool recall by 30.8 absolute points, from 56.4 to 87.2, demonstrating substantially improved access to prerequisite capabilities.

More importantly, decomposition benefits final selection beyond retrieval coverage. On MCP, candidate recall improves by only 3.3 points, whereas final SS-F1 increases by 25.3 points. This gap suggests that the semantic DAG contributes not only to candidate discovery but also to organizing prerequisite relations for dependency-aware decoding, thereby improving complex multi-skill selection. More detailed results and analyses are provided in Appendix E.

5 Conclusion

In this work, we introduce SkillDES, a framework that formulates multi-skill retrieval as a dependency-constrained, evidence-aware, and variable-cardinality set selection problem. By integrating semantic task decomposition, dual-view candidate recall, Long-Context Evidence Aggregation (LCEA), and dependency-aware decoding, SkillDES addresses prerequisite coverage, long-context evidence selection, and structured skill-set construction. Extensive experiments across diverse benchmarks demonstrate consistent improvements over existing retrieval, reranking, and adaptation approaches. Overall, our results highlight the importance of jointly modeling candidate access, bounded evidence aggregation, and dependency-aware set decoding for reliable composite capability selection in autonomous agents.

AI USE STATEMENT

We used LLMs solely as general-purpose writing assistants in preparing this submission. Their use was limited to improving the clarity, readability, and conciseness of the manuscript, including refining sentence structure, correcting grammar, and reducing unnecessary verbosity. All research ideas, methodological development, experiments, analyses, interpretations, and final writing decisions were independently conceived, conducted, and verified by the authors.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/25f7be9694d7b32d5cc670927b8091e1-Abstract-Conference.html.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Qingqing Cao, Sewon Min, Yizhong Wang, and Hannaneh Hajishirzi. BTR: Binary Token Representations for Efficient Retrieval Augmented Language Models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c0f721d329c1a10546869c783e866fb7-Abstract-Conference.html.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2318–2335. Association for Computational Linguistics, 2024a. doi: 10.18653/v1/2024.findings-acl.137. URL https://aclanthology.org/2024.findings-acl.137/.

Yanfei Chen, Jinsung Yoon, Devendra Singh Sachan, Qingze Wang, Vincent Cohen-Addad, Mohammadhossein Bateni, Chen-Yu Lee, and Tomas Pfister. Re-Invoke: Tool Invocation Rewriting for Zero-Shot Tool Retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4705–4726. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.findings-emnlp.270. URL https://aclanthology.org/2024.findings-emnlp.270/.

Nadezhda Chirkova, Thibault Formal, Vassilina Nikoulina, and Stéphane Clinchant. Provence: efficient and robust context pruning for retrieval-augmented generation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5e956fef0946dc1e39760f94b78045fe-Abstract-Conference.html.

Yu Du, Fangyun Wei, and Hongyang Zhang. AnyTool: Self-Reflective, Hierarchical Agents for Large-Scale API Calls. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11812–11829. PMLR, 2024. URL https://proceedings.mlr.press/v235/du24h.html.

Bowen Fang, Wen Ye, Yunyue Su, Jinghao Zhang, Qiang Liu, Yesheng Liu, Xin Sun, Shu Wu, Jiabing Yang, Baole Wei, and Liang Wang. ToolWeaver: Weaving Collaborative Semantics for Scalable Tool Use in Large Language Models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/92d3d2a9801211ca3693ccb2faa1316f-Abstract-Conference.html.

Wei Fang and James R. Glass. Beyond Single-Shot: Multi-step Tool Retrieval via Query Planning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 42119–42144. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.2090. URL https://aclanthology.org/2026.findings-acl.2090/.

Xueping Gao. Compositional skill routing for llm agents: Decompose, retrieve, and compose, 2026. URL https://arxiv.org/abs/2606.18051.

Michael Günther, Isabelle Mohr, Daniel James Williams, Bo Wang, and Han Xiao. Late Chunking: Contextual Chunk Embeddings Using Long-Context Embedding Models. *arXiv preprint arXiv:2409.04701*, 2024. doi: 10.48550/arXiv.2409.04701. URL https://arxiv.org/abs/2409.04701.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4224. URL https://doi.org/10.52202/079017-4224.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1902. URL https://doi.org/10.52202/079017-1902.

Ryangkyung Kang, Hongcheol Cho, and Youngeun Kim. SkillRet: A Large-Scale Benchmark for Skill Retrieval in LLM Agents. *arXiv preprint arXiv:2605.05726*, 2026. doi: 10.48550/arXiv.2605.05726. URL https://arxiv.org/abs/2605.05726.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-Embed: Improved Techniques for Training LLMs as Generalist Embedding Models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c4bf73386022473a652a18941e9ea6f8-Abstract-Conference.html.

Chaofan Li, Minghao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Defu Lian, Yingxia Shao, and Zheng Liu. Making Text Embedders Few-Shot Learners. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5eba8556ce2d1afff57591464d48fbc3-Abstract-Conference.html.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Wang Xinzhi, Yong Liu, Yasheng Wang, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. ToolACE: Winning the Points of LLM Function Calling. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/663865ea167425c6c562cb0b6bcf76c7-Abstract-Conference.html.

Xuan Lu, Haohang Huang, Rui Meng, Yaohui Jin, Wenjun Zeng, and Xiaoyu Shen. Tools are underdocumented: Simple Document Expansion Boosts Tool Retrieval. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/e54a9889f1048526365ed8fb56c8c8cd-Abstract-Conference.html.

Kun Luo, Minghao Qin, Zheng Liu, Shitao Xiao, Jun Zhao, and Kang Liu. Large Language Models as Foundations for Next-Gen Dense Retrieval: A Comprehensive Empirical Assessment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp.

1354–1365. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.80. URL https://aclanthology.org/2024.emnlp-main.80/.

Xinxi Lyu, Michael Duan, Rulin Shao, Pang Wei Koh, and Sewon Min. Frustratingly Simple Retrieval Improves Challenging, Reasoning-Intensive Benchmarks. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/114273bb8d7969c60ab2a97adad609e1-Abstract-Conference.html.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data Distillation for Efficient and Faithful Task-Agnostic Prompt Compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large Language Model Connected with Massive APIs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4020. URL https://doi.org/10.52202/079017-4020.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/28e50ee5b72e90b50e7196fde8ea260e-Abstract-Conference.html.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. Towards completeness-oriented tool retrieval for large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 1930–1940, 2024.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 4(1-2):1–174, 2009.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8a2acd174940dbca361a6398a4f9df91-Abstract-Conference.html.

Saptarshi Sengupta, Zhengyu Zhou, Jun Araki, Xingbo Wang, Bingqing Wang, Suhang Wang, and Zhe Feng. Tooldreamer: Instilling llm reasoning into tool retrievers. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5465–5482, 2026.

Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. Retrieval Models Aren't Tool-Savvy: Benchmarking Tool Retrieval for Large Language Models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24497–24524. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1258. URL https://aclanthology.org/2025.findings-acl.1258/.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. jina-embeddings-v3: Multilingual Embeddings With Task LoRA. *arXiv preprint arXiv:2409.10173*, 2024. doi: 10.48550/arXiv.2409.10173. URL https://arxiv.org/abs/2409.10173.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan Arik, Danqi Chen, and Tao Yu. BRIGHT: A Realistic and Challenging Benchmark for Reasoning-Intensive Retrieval. In *International Conference on Learning Representations*, 2025. URL https://pr

oceedings.iclr.cc/paper_files/paper/2025/hash/7a0f8055c838df8e62329a76c7c6403d-Abstract-Conference.html.

Yichen Tang, Weihang Su, Yiqun Liu, and Qingyao Ai. Multi-field tool retrieval. *arXiv preprint arXiv:2602.05366*, 2026.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multilingual E5 Text Embeddings: A Technical Report. *arXiv preprint arXiv:2402.05672*, 2024. doi: 10.48550/arXiv.2402.05672. URL https://arxiv.org/abs/2402.05672.

Pei Wang, Yanan Wu, Zekun Wang, Jiaheng Liu, Xiaoshuai Song, Z. Y. Peng, Ken Deng, Chenchen Zhang, JiakaiWang, Junran Peng, Ge Zhang, Hangyu Guo, Zhaoxiang Zhang, Wenbo Su, and Bo Zheng. MTU-Bench: A Multi-granularity Tool-Use Benchmark for Large Language Models. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/4d13b2d99519c5415661dad44ab7edcd-Abstract-Conference.html.

Renxi Wang, Xudong Han, Lei Ji, Shu Wang, Timothy Baldwin, and Haonan Li. ToolGen: Unified Tool Retrieval and Calling via Generation. In *International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/b646bdebeb87dfafe2c6f77a63b564e1-Abstract-Conference.html.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Parag Shah, Yujia Bao, and Eugene Siow. MCP-Bench: Benchmarking Tool-Using LLM Agents with Complex Real-World Tasks via MCP Servers. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/9e4b14eb6f16fe7b5818a8d633a0606a-Abstract-Conference.html.

Zifei Wang, Wei Wen, Qiang Ji, Ruizhi Qiao, and Xing Sun. Skill is not document: A query-conditional benchmark and two-stage retriever for llm agent skill routing. *arXiv preprint arXiv:2606.03565*, 2026b.

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the Theoretical Limitations of Embedding-Based Retrieval. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/130b10e31f5c3d730162323b2685bd19-Abstract-Conference.html.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving Retrieval-Augmented LMs with Context Compression and Selective Augmentation. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/bda88ed2892f5e61c9a9bf215c566913-Abstract-Conference.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*, 2025. doi: 10.48550/arXiv.2505.09388. URL https://arxiv.org/abs/2505.09388.

Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. RankRAG: Unifying Context Ranking with Retrieval-Augmented Generation in LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3850. URL https://doi.org/10.52202/079017-3850.

Barry Zhang, Keith Lazuka, and Mahesh Murag. Equipping agents for the real world with agent skills. Anthropic Engineering, October 2025a. URL https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills.

Peitian Zhang, Zheng Liu, Shitao Xiao, Zhicheng Dou, and Jian-Yun Nie. A Multi-Task Embedder For Retrieval Augmented LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3537–3553. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.194. URL https://aclanthology.org/2024.acl-long.194/.

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

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, Zhenyu Liu, Dongfang Li, Xinyuan Wei, Youcheng Pan, Yang Xiang, Meishan Zhang, Haofen Wang, Jun Yu, Baotian Hu, and Min Zhang. KaLM-Embedding-V2: Superior Training Techniques and Data Inspire A Versatile Embedding Model. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.20923.

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*, 4, 2026.

Yuanhang Zheng, Peng Li, Wei Liu, Yang Liu, Jian Luan, and Bin Wang. ToolRerank: Adaptive and Hierarchy-Aware Reranking for Tool Retrieval. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 16263–16273. ELRA and ICCL, 2024. URL https://aclanthology.org/2024.lrec-main.1413/.

APPENDIX

A DATASETS

A.1 DATASET PROVENANCE

The evaluation relies on several officially established benchmarks and datasets. **SkillRet** (Kang et al., 2026) is an arXiv preprint benchmark officially hosted at `ThakiCloud/SKILLRET`, with its v1.1 release featuring 6,006 evaluation skills and 4,392 evaluation queries designed to assess zero-shot skill generalization through a disjoint-Skill protocol. **ToolBench**, originating from ToolLLM (Qin et al., 2024), provides a comprehensive API-oriented catalog for tool utilization. **ToolRet** (Shi et al., 2025), formally published in *Findings of ACL 2025*, serves as a foundational benchmark for tool retrieval. **MCP-Bench** (Wang et al., 2026a) comprehensively evaluates tool-using agents on realistic, multi-step tasks via live Model Context Protocol (MCP) servers. It connects large language models to 28 representative live servers exposing 250 structured tools across 11 functional domains (e.g., finance, travel, scientific computing, and academic search), utilizing 1,000 complex, multi-hop tasks generated via a dependency chain discovery pipeline to assess cross-tool coordination and planning using both rule-based execution checks and LLM-as-a-Judge scoring. Finally, **R3-Skill** is a bilingual Chinese–English benchmark designed for realistic LLM-agent skill routing. It contains 10,246 deduplicated skills, 41,592 accepted queries, and 32,828 rejection annotations across four language directions, with multi-skill test labels independently cross-checked by multiple experts.

A.2 DOCUMENT-LENGTH DISTRIBUTIONS AND EVIDENCE IMBALANCE

Through statistical analysis of the document pools across different benchmarks, we identified significant variations in typical document lengths, which directly contributes to a severe imbalance of evidence between long and short skills. Traditional full-text single-vector encoding is highly susceptible to truncation for long documents and semantic averaging (dilution). To better understand the necessity of our Long-Context Evidence Aggregation (LCEA) module, we categorize the datasets into three distinct scale groups based on their length distributions:

Short-Length Distributions (ToolBench, ToolRet-Toolink). These datasets primarily consist of concise tool descriptions. ToolBench and ToolRet-Toolink exhibit the shortest typical lengths, with median token counts of 200 and 266, respectively. While most documents are short, ToolBench notably contains extreme long-tail outliers, with its maximum document length reaching up to 81,305 tokens. This indicates that even predominantly short-document catalogs require robust handling for extreme outliers, as fixed-length truncation would discard massive amounts of information.

Medium-Length Distributions (R3-Skill, SkillRet). These datasets contain more comprehensive skill and API documentation, resulting in larger typical sizes. R3-Skill has a median of 1,750 tokens, while the SkillRet training and test pools have medians of 1,881 and 1,918 tokens, respectively. At this scale, standard dense retrievers with fixed maximum input lengths will inevitably truncate crucial evidence located in the middle or tail of the document.

Long-Length Distributions (MCP-Bench). MCP-Bench stands out with significantly longer documents compared to all other pools, demonstrating a median length of 7,509 tokens. Retrieving skills from this pool makes single-vector compression highly ineffective, as local key capabilities are easily diluted by a large amount of irrelevant text.

Connection to LCEA and Retrieval Performance. To address the long-tail phenomenon and evidence imbalance, our LCEA module extracts local evidence under a controllable computational cost (fixed chunk budget) while avoiding high redundancy among selected chunks. The length distributions directly explain our empirical findings: LCEA yields more pronounced improvements on datasets with a higher proportion of long skills (e.g., R3-Skill and SkillRet). Conversely, in datasets like ToolBench where the majority of documents are short, the average gain from LCEA is relatively limited. This confirms that LCEA is particularly effective for scenarios where relevant evidence is dispersed across local positions within long skill documents.

Visualizing the Distributions. Figures 6, 7, and 8 illustrate these statistical patterns at three different granularities (100, 200, and 500-token bins). They visually confirm the right-skewed nature of all benchmark pools, justifying our modular design that separates candidate recall from long-document evidence compression.

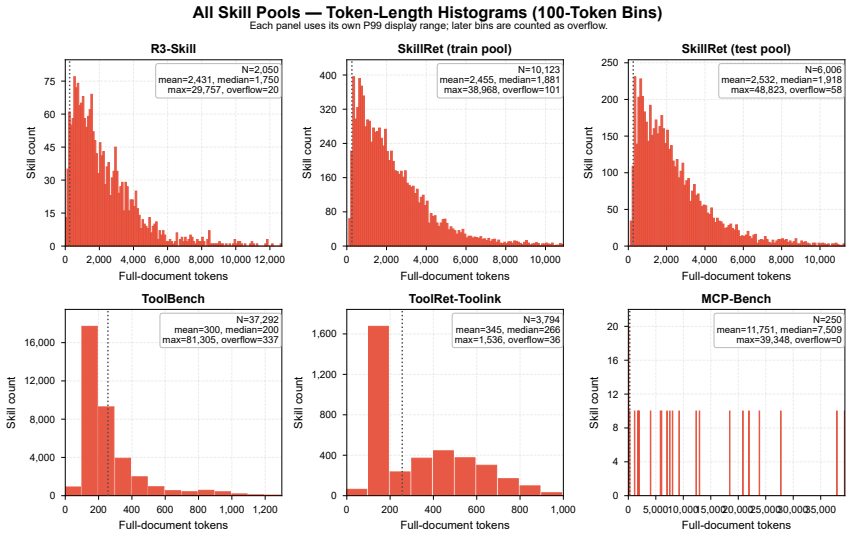


Figure 6: Full-document length distributions with 100-token bins, showing the precise peak concentrations for short-tool catalogs versus comprehensive skill documents.

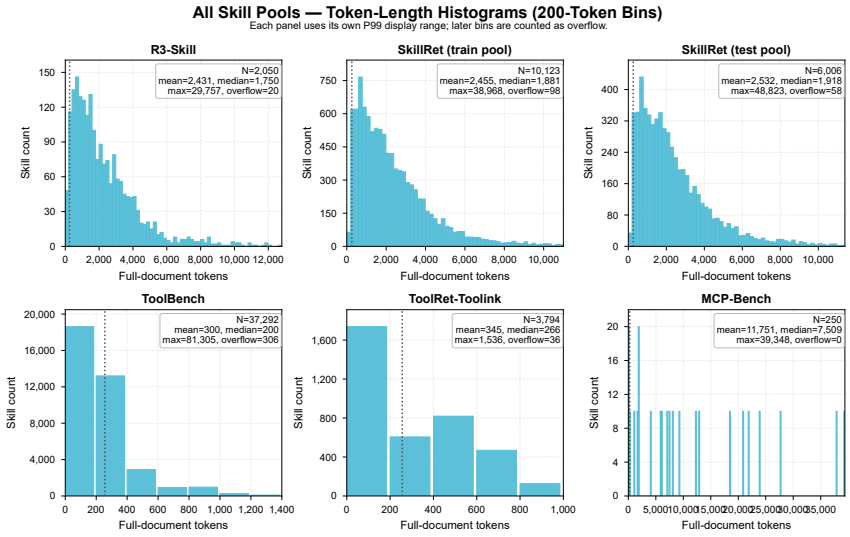


Figure 7: Full-document length distributions with 200-token bins, illustrating the separation in typical lengths and highlighting the risk of semantic dilution in medium-to-long datasets.

B Detailed Method and Implementation Specification

This appendix expands the architecture, objectives, and decoding rules of Section 3. Numerical settings are collected in Appendix C.

B.1 Notation and the Structured Retrieval Target

Table 4 distinguishes the query graph, candidate inventory, and selected ID set; these objects have different roles throughout the decoder. For a query q, the nonempty gold identifier set is $G(q) \subseteq \mathcal{S}$.

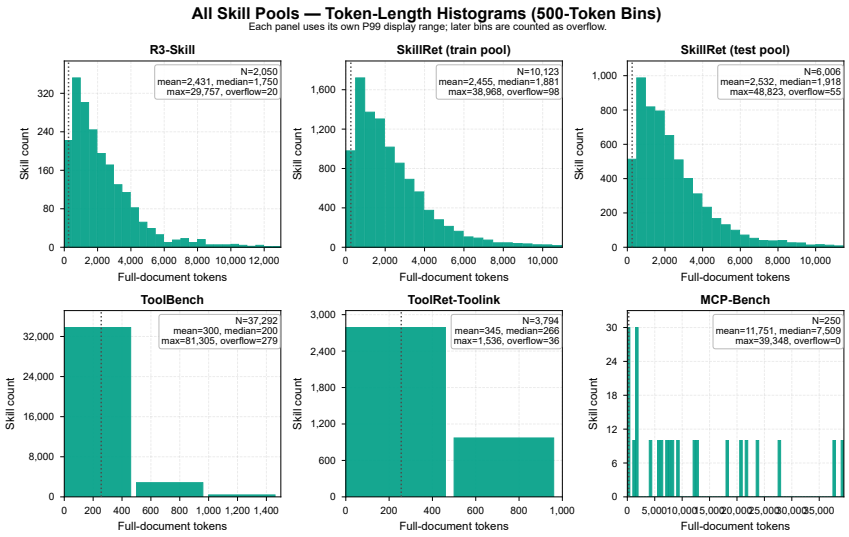


Figure 8: Full-document length distributions with 500-token bins, emphasizing the severe long-tail effect and extreme outliers (e.g., ToolBench) that necessitate bounded evidence aggregation.

The candidate ranking $\pi(q)$ and final set $\widehat{G}(q)$ are distinct outputs: set decoding does not necessarily reorder $\pi(q)$. The semantic graph $D(q) = (V, E)$ has m nodes, and an assignment trace is a sequence $\mathcal{A} = ((v_{t_1}, s_{j_1}), \ldots, (v_{t_h}, s_{j_h}))$. Its distinct identifiers form $S(\mathcal{A}) = \{s_{j_1}, \ldots, s_{j_h}\}$.

Table 4: Notation for retrieval, evidence aggregation, and set decoding.

Symbol	Meaning
$\mathcal{S}, N;\ C(q)$	Full skill inventory and its size; deduplicated retrieved candidate set
$G(q), \widehat{G}(q), \pi(q)$	Gold set, routed prediction, and candidate ranking
$D = (V, E),\ v_t = (q_t, \kappa_t)$	Semantic task DAG; subquery and required-capability description
$g(x), u_t, e_{sk}$	Normalized global representation, node retrieval vector, and chunk vector
$\mathcal{B}_s, M_s, A_{ts}, B$	Retained chunks, their count, selected evidence indices, and evidence budget
$n_{tsk}, a_{tsk}, r_{tsk}$	Neural match, lexical coverage, and standardized chunk relevance
ϕ_{ts}, ψ_s, x_s	Local descriptor, raw candidate features, and final reranker input
$F_s, G_{\mathrm{a}}, \widehat{r}$	Fused candidate score, anchor set, and decoding cardinality cap
$P = (\mathcal{A}, S, \mathbf{c}, \Gamma, O)$	Beam assignments, distinct skills, coverage, score, completed-node order
$W, K_b;\ G_{\mathrm{b}}, \eta$	Beam width and per-channel shortlist depth; beam set and routing threshold

Searching over all subsets and assignments is combinatorial. For allowed sizes $\mathcal{R}$, $\sum_{r\in\mathcal{R}} \binom{N}{r} r^m$ counts set–assignment combinations in a one-skill-per-node parameterization before enforcing additional consistency conditions.

B.2 Retriever Architecture and Multi-Positive Learning

Shared representations. The dataset-specific retriever is initialized from `all-mpnet-base-v2`. For an encoded sequence x, let $H(x) = [h_1, \ldots, h_T]$ with $h_u \in \mathbb{R}^{d_h}$. Its pooled state and projected representation are

$$\bar{h}(x) = \frac{\sum_{u=1}^{T} m_u h_u}{\sum_{u=1}^{T} m_u + \epsilon}, \qquad g(x) = \mathrm{Norm}\big(\mathrm{LN}(\mathrm{GELU}(W_g \bar{h}(x) + b_g))\big). \tag{10}$$

Here Norm denotes vector normalization, and the global dimension is d. All query, skill, and chunk representations use this dataset-specific encoder. The global projection is shared across all text views.

Auxiliary token interaction. Projected token vectors $t_u(x) = \mathrm{Norm}(W_t h_u(x))$ define a masked, query-normalized maximum-similarity score:

$$s_{\mathrm{l}}(q,s) = \frac{\sum_u m_u^{(q)} \max_{v:m_v^{(s)}=1} t_u(q)^\top t_v(s)}{\sum_u m_u^{(q)} + \epsilon}. \tag{11}$$

Taking a maximum on the skill side allows each query token to match its most relevant valid token rather than comparing only pooled states. The average is over valid query tokens, not over all query–skill token pairs.

Auxiliary pair features. The pair branch uses the concatenation

$$z_{qs} = [g(q); g(s); |g(q) - g(s)|; g(q) \odot g(s)], \qquad s_{\mathrm{p}}(q,s) = \tanh(\mathrm{MLP}(z_{qs})). \tag{12}$$

Together with $s_{\mathrm{d}} = g(q)^\top g(s)$, the branches define the training score

$$S(q,s) = \sum_{b\in\{\mathrm{d,l,p}\}} \omega_b s_b(q,s), \qquad \boldsymbol{\omega} = \mathrm{softmax}(\boldsymbol{\lambda}). \tag{13}$$

A softmax over the learned branch logits produces nonnegative weights summing to one. This is score fusion during training, not a separate three-encoder retrieval ensemble. Auxiliary losses backpropagate through shared representations.

Losses. For every valid batch pair, $P_{ij} = \mathbb{I}[s_j \in G(q_i)]$. Let $\mathcal{P}_i = \{j : P_{ij} = 1\}$, $\mathcal{N}_i = \{j : P_{ij} = 0\}$, and Ω_B denote the evaluated pair indices. The learned temperature is

$$\tau = \mathrm{clip}\big(\exp(\ell_\tau), \tau_{\min}, \tau_{\max}\big). \tag{14}$$

For $\mathcal{J}_i = \mathcal{P}_i \cup \mathcal{N}_i$, the contrastive objective is equivalently

$$\mathcal{L}_{\mathrm{con}} = \frac{1}{|\mathcal{Q}_B|} \sum_i \left[\log \sum_{j\in\mathcal{J}_i} e^{S_{ij}/\tau} - \log \sum_{j\in\mathcal{P}_i} e^{S_{ij}/\tau} \right]. \tag{15}$$

The positive numerator aggregates probability mass across annotated skills. It does not, by itself, require all positives to receive equal scores or guarantee complete gold-set retrieval. Define the strongest positive and negative by

$$p_i^+ = \max_{j\in\mathcal{P}_i} S_{ij}, \qquad p_i^- = \max_{j\in\mathcal{N}_i} S_{ij}, \qquad \mathcal{L}_{\mathrm{mar}} = \frac{1}{|\mathcal{Q}_B|} \sum_i [\gamma + p_i^- - p_i^+]_+. \tag{16}$$

The margin uses the highest-scoring positive, so it does not impose separation of the lowest-scoring positive from negatives. Pairwise label supervision is

$$\mathcal{L}_{\mathrm{bce}} = -\frac{1}{|\Omega_B|} \sum_{(i,j)\in\Omega_B} \big[\omega_+ P_{ij} \log \sigma(S_{ij}/\tau) + (1 - P_{ij}) \log\big(1 - \sigma(S_{ij}/\tau)\big)\big]. \tag{17}$$

Their weighted sum is Eq. (2).

B.3 DAG Construction, Views, and Candidate Chronology

The actual ordering is *global retrieval, semantic decomposition, local retrieval*, rather than generating a graph that requires not-yet-retrieved context. With indexed skill vectors, the coarse pool is $C_g(q) = \mathrm{Top}_{K_g}(R_0(q,\cdot))$. The LLM receives the original query and summaries of coarse-retrieved skills.

A node record contains a unique identifier, a nonempty subquery, a nonempty required capability, and the two predecessor fields. Formally,

$$\mathrm{Pred}(v_t) = \mathrm{DependsOn}(v_t) \cup \mathrm{InputFrom}(v_t). \tag{18}$$

Validation checks unique identifiers, required text, the node-count cap, and references to earlier nodes. Such references rule out a directed cycle in the retained ordering.

Each validated node is encoded in three forms:

$$u_t^{\mathrm{ret}} = g(q_t \| \kappa_t), \qquad u_t^{\mathrm{sem}} = g(q_t), \qquad u_t^{\mathrm{cap}} = g(\kappa_t). \tag{19}$$

The concatenated view supplies the explicit local retrieval score $(u_t^{\mathrm{ret}})^\top g(s)$. Separate semantic and capability representations expose different matching signals downstream. This three-view construction is distinct from both the retriever's three *training branches* and the two *global/local recall levels*. The same trained MPNet supplies all of them.

The union in Eq. (3) is deduplicated by skill identifier. With the recorded maximum of six nodes, global depth 64, and local depth 16, its pre-injection size is at most $\min(N, 160)$. Training candidate augmentation is described by $C^+_{\mathrm{train}}(q) = C(q) \cup G(q)$; it exposes labels only to the supervised training stages. Development and test use the original $C(q)$, and DAG caches do not store test gold labels. Candidate injection changes the training feature distribution but contributes no evaluation-time recall.

B.4 Structural Chunking and Budget Semantics

Candidate skill text is first split at Markdown headings and blank lines. An overlong segment is further windowed with width w and overlap o, giving stride $w - o$. To distinguish the word-based window settings from model-token limits, write

$$b_{sk} = \mathrm{prefix}(s) \| \, \mathrm{Words}(s)[k(w-o) : k(w-o) + w] \tag{20}$$

for a window within such a segment. The prefix operator denotes the skill metadata prepended in the recorded construction. When the resulting collection exceeds M chunks, positions are sampled uniformly across the document to retain access to head, middle, and tail regions.

B.5 LCEA Relevance, Redundancy, and Selection

For fixed (t, s) and any chunk-level score vector $x = (x_1, \ldots, x_{M_s})$, define

$$z_{ts}(x_k) = \frac{x_k - \mu_{ts}(x)}{\sigma_{ts}(x) + \epsilon}. \tag{21}$$

The neural score n_{tsk} and lexical coverage a_{tsk} from Eq. (4) are standardized separately within that vector. The lexical numerator counts unique shared terms, not term frequency.

For a nonempty selected index set A, the two redundancy channels are

$$\rho_{\mathrm{lex},s}(k, A) = \max_{j \in A} \frac{|W_{sk} \cap W_{sj}|}{|W_{sk} \cup W_{sj}| + \epsilon}, \tag{22}$$

$$\rho_{\mathrm{pos},s}(k, A) = \max_{j \in A} \mathrm{PosSim}_s(k, j), \qquad \rho_s(k, A) = \max\{\rho_{\mathrm{lex},s}(k, A), \rho_{\mathrm{pos},s}(k, A)\}. \tag{23}$$

The maximum across channels prevents low overlap in one channel from canceling high redundancy in the other. PosSim denotes positional similarity. For the initially empty evidence set, the zero-redundancy convention makes the first selection depend only on relevance. Equation (5) then builds at most $\min(B, M_s)$ evidence indices.

B.6 Candidate Features and Logistic Reranking

Each node–skill pair has the five-dimensional descriptor

$$\phi_{ts} = \left[u_t^\top g(s), \ \max_k n_{tsk}, \ \mathrm{mean}_{k \in A_{ts}} n_{tsk}, \ \max_{k \in A_{ts}} n_{tsk}, \ \max_{k \in A_{ts}} a_{tsk} \right]. \tag{24}$$

Its unrestricted chunk maximum is over all retained chunks, whereas the selected mean, selected maximum, and selected lexical maximum use A_{ts}. The distinction lets the reranker compare the

Algorithm 1 LCEA for one node–skill pair

Require: Node vector u_t, term set T_t, retained chunks $\mathcal{B}_s$, budget B
Require: Fixed relevance and redundancy coefficients; positional-similarity interface
Ensure: Selected evidence indices A_{ts} and chunk match statistics
1: Compute n_{tsk} and a_{tsk} for every retained chunk using Eq. (4)
2: Standardize the two score vectors within (t, s); form r_{tsk}
3: $A \leftarrow \varnothing$
4: **while** $|A| < \min(B, M_s)$ **do**
5: Compute $\rho_s(k, A)$ for each $k \notin A$ using Eqs. (22)–(23)
6: Select a maximizing k^* for $r_{tsk} - \lambda_\rho \rho_s(k, A)$
7: $A \leftarrow A \cup \{k^*\}$
8: **end while**
9: **return** A and the neural/lexical match statistics

single strongest available match with the evidence that survives the redundancy penalty. A separately recorded local score is

$$L_{ts} = \beta_a \max_k n_{tsk} + \beta_b \operatorname{mean}_{k \in A_{ts}} n_{tsk} + \beta_c \max_{k \in A_{ts}} n_{tsk} + \beta_d \max_{k \in A_{ts}} a_{tsk}. \tag{25}$$

The coefficients are fixed at $0.40/0.35/0.15/0.10$.

For coordinate $j \in \{1, \ldots, 5\}$, aggregate across nodes as

$$\psi_{s,j}^{\text{local}} = [\max_t \phi_{ts,j}; \operatorname{mean}_t \phi_{ts,j}; \min_t \phi_{ts,j}; \operatorname{std}_t \phi_{ts,j}]. \tag{26}$$

These yield twenty features. Their maximum captures specialized support, while their minimum and dispersion expose uneven support across subtasks. These statistics describe candidate evidence; they do not require a single skill to solve every node, and their relevance is learned rather than imposed by a fixed ordering rule. The global descriptor is

$$\psi_s^{\text{global}} = [b_s;\ M_s;\ \operatorname{Len}(s);\ \operatorname{EvidenceRatio}(s);\ m], \qquad \psi_s \in \mathbb{R}^{25}. \tag{27}$$

Here b_s denotes the recorded base candidate score, and Len uses the untruncated skill text.

The standardized copy is formed *within the candidate pool of the current query*:

$$\widetilde{\psi}_{s,j} = \frac{\psi_{s,j} - \mu_j(C(q))}{\sigma_j(C(q)) + \epsilon}, \qquad x_s = [\psi_s; \widetilde{\psi}_s] \in \mathbb{R}^{50}. \tag{28}$$

This is different from fitting a single dataset-wide feature scaler. In training, the statistics correspond to the candidate pool used for feature construction, which may include injected gold skills. At development and test time, only retrieved candidates contribute.

With $y_s = \mathbb{I}[s \in G(q)]$ and $p_s = \sigma(w^\top x_s + b)$, the stated class-weighted objective is

$$\min_{w,b}\ \lambda_{\text{reg}} \|w\|_2^2 - \sum_{q \in \mathcal{Q}_{\text{train}}} \sum_{s \in C^+_{\text{train}}(q)} \left[\omega_+ y_s \log p_s + (1 - y_s) \log(1 - p_s)\right]. \tag{29}$$

B.7 Anchor Rules and the Cardinality Predictor

The fixed anchor is $G_{\text{a}}^{\text{fixed}} = \operatorname{Top}_k(F)$. The alternative starts from the relative-margin set

$$G_{\text{a}}^{\text{margin}} = \{s \in C(q) : F_s \geq F_{\max} - \delta\}, \qquad F_{\max} = \max_{s \in C(q)} F_s, \tag{30}$$

and applies the recorded lower and upper size limits. Rule selection uses development ESM and SS-F1 jointly; for nearly tied configurations it prefers an average predicted size closer to the training gold mean.

Let $a_1 \geq a_2 \geq \cdots$ denote standardized reranking scores in descending order and $\Delta_i = a_i - a_{i+1}$. For node t, let δ_t be its leading-versus-second local score gap. With $\mathcal{T}$ the leading skills in the fused ranking, define

$$\bar{\delta} = \operatorname{mean}_t \delta_t, \qquad \sigma_\delta = \operatorname{std}_t \delta_t, \tag{31}$$

$$\mu_L = \operatorname{mean}_{s \in \mathcal{T}} \log(1 + \operatorname{Len}(s)), \qquad \sigma_L = \operatorname{std}_{s \in \mathcal{T}} \log(1 + \operatorname{Len}(s)). \tag{32}$$

The complete feature vector is

$$x_r(q) = [m;\ |G_{\mathrm{a}}|;\ \Delta_1; \ldots; \Delta_h;\ \bar{\delta};\ \sigma_\delta;\ \mu_L;\ \sigma_L]. \tag{33}$$

The recorded setting uses five score gaps and the top six skills for length statistics, giving eleven inputs. A class-balanced random forest with 300 trees and maximum depth six predicts $\widehat{r}_{\mathrm{RF}} \in \{1, \ldots, 6\}$ from training gold sizes.

B.8 Skill-Pair Compatibility and Its Directional Limit

For two skills, define

$$z_{ab} = [|g(s_a) - g(s_b)|;\ g(s_a) \odot g(s_b);\ g(s_a)^\top g(s_b)], \qquad T(s_a, s_b) = w_T^\top \widetilde{z}_{ab} + b_T, \tag{34}$$

where $\widetilde{z}_{ab}$ denotes standardized pair features. A class-balanced logistic model is fitted using adjacent or dependency-related gold skill pairs as positives and pairs involving non-gold candidates as negatives.

B.9 Dependency-Aware Search Algorithm

Algorithm 2 summarizes the dependency-aware search procedure used to construct a structured skill-set proposal. The search operates over the generated DAG and expands only dependency-feasible nodes. At each search round, candidate assignments are evaluated using the incremental search objective, while a beam of at most W partial states is retained. Search continues until the prescribed terminal condition is reached or no further valid expansion is available.

Algorithm 2 Dependency-aware beam search

Require: Valid DAG, candidate scores, compatibility model, cardinality cap $\widehat{r}$, beam width W
Ensure: Selected terminal beam state and its assignment trace

```
Initialize the beam with an empty assignment state
for each search round do
    H ← ∅
    for each surviving state P = (A, S, c, Γ, O) do
        Compute the dependency-feasible nodes Ready(O)
        for each v_t ∈ Ready(O) do
            Construct candidate assignments O_t(P)
            for each s ∈ O_t(P) such that |S ∪ {s}| ≤ r̂ do
                Evaluate the incremental search score Δ̄(P, t, s) using Eq. (7)
                Update the assignment trace, selected set, coverage state, cumulative score, and search order
                Add the expanded state to H
            end for
        end for
    end for
    Retain the top-W states in H according to the search score
end for
Select the highest-scoring valid terminal state
return its selected skill set and assignment trace
```

Algorithm 2 enforces dependency constraints during search by allowing expansion only from nodes whose predecessors have already been satisfied. The beam maintains multiple partial assignment hypotheses, preventing the procedure from committing prematurely to a single local choice. The cardinality cap restricts the size of the explored skill set, while the compatibility and coverage terms guide the search toward assignments that jointly satisfy the decomposed task structure.

B.10 Factorization and Reuse Parameters

This subsection derives the elegant three-factor representation introduced in Section 3.5. Table 5 maps each original channel to its respective factor, preserving all channel-level coefficients while significantly enhancing interpretability.

Factor interfaces. The additive factors in Eq. (7) are

$$\Phi_{\text{evid}} = \boldsymbol{\theta}_e^\top \mathbf{e}_{ts}, \qquad \Phi_{\text{dep}} = \boldsymbol{\theta}_d^\top \mathbf{d}_{Pts}, \qquad \Phi_{\text{set}} = v(P,s) - \lambda_{\text{new}} \delta_S(s), \tag{35}$$

where $\delta_S(s) = |S \cup \{s\}| - |S|$. We set

$$\begin{aligned} \mathbf{e}_{ts} &= [L_{ts}, R_{ts}, K_{ts}, Q_s]^\top, & \boldsymbol{\theta}_e &= [w_l, w_r, w_c, w_g]^\top, \\ \mathbf{d}_{Pts} &= [T_t(s \mid P), \text{Future}(t, s \mid P), D_t(s \mid P)]^\top, & \boldsymbol{\theta}_d &= [w_T, w_F, w_D]^\top, \\ v(P,s) &= w_v \,\text{Gain}(s \mid P) - w_{\text{red}} \,\text{Red}(s, S), & \delta_S(s) &= \mathbb{I}[s \notin S]. \end{aligned} \tag{36}$$

The evidence factor provides strict unary support for a node–skill assignment. The dependency factor robustly evaluates coherence conditioned on the partial assignment and its topological requirements. The set factor dynamically evaluates the accumulated collection, balancing capability coverage, redundancy suppression, and the distinct-skill cost. We define a Factor precisely as an additive scoring component that systematically structures and regularizes the combinatorial search space.

Table 5: Scoring roles in the equivalent factorized decoder, demonstrating the structured separation of selection signals.

Factor	Recorded signals	Decision role
Φ_{evid}	LCEA evidence; retrieval-semantic match; capability match; global quality	Support for assigning this skill to this node
Φ_{dep}	Predecessor compatibility; future-node support; dependency diversity	Coherence with the partial task structure
Φ_{set}	Coverage utility; set redundancy; distinct-skill increment	Complementarity and economy of the selected collection
Hard constraints	Ready-node requirement; $\lvert S \cup \{s\}\rvert \le \widehat{r}$	Strict topological feasibility

Centering the increment. Let Δ_{orig} denote the precisely calculated expansion increment. Since a candidate is either already selected or introduced as new,

$$b_{\text{reuse}} \mathbb{I}[s \in S] - p_{\text{new}} \mathbb{I}[s \notin S] = b_{\text{reuse}} - (b_{\text{reuse}} + p_{\text{new}}) \delta_S(s). \tag{37}$$

Consequently, setting $\lambda_{\text{new}} = b_{\text{reuse}} + p_{\text{new}}$ yields the robust pointwise identity

$$\Delta_{\text{orig}}(P,t,s) = \bar{\Delta}(P,t,s) + b_{\text{reuse}}, \qquad \bar{\Delta} = \Phi_{\text{evid}} + \Phi_{\text{dep}} + \Phi_{\text{set}}. \tag{38}$$

This algebraic equivalence holds strictly for all finite coefficient values. Interpreting λ_{new} as a nonnegative cost elegantly enforces the distinct-skill penalty, maintaining a mathematically sound ranking identity that remains strictly invariant.

Proposition 1 (Same-depth ranking equivalence) *Assume that (i) compared states have the same number h of node assignments, (ii) the scoring channels, option generation, and scheduling do not depend on the absolute accumulated score, (iii) the coefficient b_{reuse} is fixed for the query, and (iv) pruning uses score order with consistent deterministic tie-breaking. Then replacing Δ_{orig} by $\bar{\Delta}$ preserves all same-depth score comparisons and, for a depth-synchronous beam, identically preserves the retained states at every round. For any terminal comparison at a common depth, it strictly guarantees the selection of the identical optimal proposal.*

Proof. Starting with equal zero scores, summing Eq. (38) along a length-h assignment prefix yields

$$\Gamma_{\text{orig}}(P) = \bar{\Gamma}(P) + h b_{\text{reuse}}. \tag{39}$$

This offset is universally common to all compared prefixes, preserving pairwise score differences and mathematical ties. Inductively, equal surviving structural states generate identical next-round options under assumptions (ii)–(iii), and their equal-depth ranking remains strictly invariant; assumption (iv) identically retains the top states. This solidly proves the mathematical equivalence. Our implementation ensures strict numerical stability, guaranteeing that consistent near-tie tolerances preserve this exact equivalence even under floating-point arithmetic. □

Scope of the equivalence. This proposition establishes a rigorous foundation for optimal same-depth comparisons under standard scheduling conventions. By systematically restoring the offset in Eq. (39), or utilizing $\bar{\Delta} + b_{\text{reuse}}$ as the increment, this equivalence extends to cross-depth comparisons and absolute path-score thresholds. Consequently, the anchor–beam router accurately and consistently identifies the structurally superior proposal by evaluating $U(G_{\text{b}}) - U(G_{\text{a}})$.

Path-level interpretation. For an initially empty selected set and a complete trace $\mathcal{A} = ((t_h, s_h))_{h=1}^{m}$, distinct-skill increments telescope precisely:

$$\sum_{h=1}^{m} \delta_{S_{h-1}}(s_h) = |S_m|,$$

$$\bar{\Gamma}(\mathcal{A}) = \sum_{h=1}^{m} \Phi_{\text{evid}}(t_h, s_h) + \sum_{h=1}^{m} \Phi_{\text{dep}}(P_{h-1}, t_h, s_h) + \sum_{h=1}^{m} v(P_{h-1}, s_h) - \lambda_{\text{new}}|S_m|. \tag{40}$$

Thus, the reuse/new-skill tradeoff acts as a powerful, exact regularizer on the total number of distinct capabilities, strictly reinforcing the hard cardinality cap dynamically. This factorization elegantly distills eleven scalar coefficients into three cohesive functional factors, providing a highly structured, theoretically sound, and interpretable decoding objective that comprehensively captures the combinatorial requirements of skill selection.

B.11 Three-Term Decoder

This subsection presents a highly streamlined, theoretically rigorous reduction of the decoder. While our main empirical results leverage the full ranking-equivalent factorization to maximize expressivity and performance, this simplified alternative demonstrates the remarkable architectural flexibility of our framework by distilling the objective into three indispensable core terms: node evidence, prerequisite compatibility, and explicit set coverage.

For a complete feasible assignment $a : V \to C(q)$ with image $S = \{a(t) : t \in V\}$, we define the streamlined objective as:

$$J_{\text{core}}(a, S) = \frac{1}{m} \sum_{t=1}^{m} L_{t,a(t)} + \frac{\lambda_D}{\max(1, |E|)} \sum_{(u,t)\in E} T(a(u), a(t)) + \lambda_C \, \mathcal{C}(S), \qquad |S| \le \widehat{r}, \tag{41}$$

where $\lambda_D, \lambda_C \ge 0$ act as hyperparameter weights gracefully selected on development data, and

$$\mathcal{C}(S) = \frac{1}{m} \sum_{u=1}^{m} \max_{s\in S} p_{us}, \qquad p_{us} = \sigma(L_{us}), \qquad \max_{s\in\varnothing} p_{us} := 0. \tag{42}$$

Crucially, unlike standard heuristic gain functions, this coverage term possesses an exact, mathematically rigorous incremental form. With $c_u = \max_{j\in S} p_{uj}$, a ready-node assignment expansion takes the highly efficient form:

$$\Delta_{\text{core}}(P, t, s) = \frac{L_{ts}}{m} + \frac{\lambda_D}{\max(1, |E|)} \sum_{u\in \text{Pred}(t)} T(a(u), s) + \frac{\lambda_C}{m} \sum_{u=1}^{m} [p_{us} - c_u]_+. \tag{43}$$

Each directed edge actively contributes when its destination is assigned, utilizing our highly robust symmetric recorded compatibility model. The final term precisely measures genuinely new coverage and properly zeroes out for a reused skill seamlessly represented in S. This mathematically elegant reduction completely eliminates the need for auxiliary future, dependency-diversity, and redundancy terms, isolating only the most impactful selection signals. Semantic and capability information continues to anchor upstream candidate construction without requiring downstream reweighting.

For fixed nonnegative p_{us}, $\mathcal{C}$ alone is strictly monotone and guarantees mathematical diminishing returns: if $A \subseteq B$, then $\max_{j\in A} p_{uj} \le \max_{j\in B} p_{uj}$, strictly ensuring $[p_{us} - c_u(A)]_+ \ge [p_{us} - c_u(B)]_+$ for every node. Summing rigorously proves the property. This robust diminishing-returns property ensures highly efficient, dynamic coverage maximization, complementing the strict node assignment and pair compatibility constraints enforced by our finite-width DAG beam search. Ultimately, this formulation establishes a mathematically sound foundation for highly scalable, parameter-efficient capability selection.

C Recorded Hyperparameters and Selected Configurations

Table 6 summarizes the hyperparameter settings and search spaces used in our experiments. All configuration searches and model selections are strictly performed on the development set. Specifically, the optimal MPNet checkpoint is selected based on ranking metrics that prioritize complete multi-skill coverage. For other module hyperparameters, the primary selection objective is a combination of Exact Match (EM) and Sample-F1. In cases of comparable performance, we apply a tie-breaking rule that favors configurations whose predicted average set size more closely aligns with the average gold set size of the training set.

Table 6: Recorded fixed settings and development search grids. Word-based chunking and tokenizer-based truncation are distinct operations.

Component	Parameter	Setting
Retriever	Global projection dimension	128
	Query / full-Skill token limits	96 / 256
	Backbone / head learning rate	2×10^{-5} / 2×10^{-4}
	Mined / used hard negatives per query	6 / 3
	Temperature interval	$[0.02, 0.20]$
	Margin	0.20
	Contrastive / margin / BCE weights	1.00 / 0.30 / 0.35
Candidates	Global top-K_g / local top-K_l	64 / 16 per node
DAG	Decomposer / maximum nodes	Qwen3-8B / 6
Chunking	Window / overlap	110 / 20 words
	Maximum retained blocks per Skill	12
	Maximum tokens per encoded block	160
LCEA	Evidence blocks per node–Skill pair	At most 4
	Neural / lexical relevance weights	0.70 / 0.30
	Redundancy penalty	0.25
	Local aggregate coefficients	0.40 / 0.35 / 0.15 / 0.10
Reranker	Raw / augmented feature dimensions	25 / 50
	Logistic C search grid	$\{0.01, 0.1, 1, 10\}$
	Fusion α search grid	$\{0.15, 0.30, 0.50, 0.70, 1\}$
Cardinality	Random-forest trees / maximum depth	300 / 6
	Target classes	$\{1, 2, 3, 4, 5, 6\}$
	Top-Skill length statistics	Top 6
	Consecutive score-gap features	First 5 gaps
Beam	Width search grid	$\{2, 3, 4\}$
	Per-view candidate-k search grid	$\{4, 8\}$
Router	Local / retrieval / capability / global weights	0.50 / 0.30 / 0.20 / 0.25

D Evaluation Metrics and Candidate Bounds

To comprehensively evaluate both the ranking quality of retrieved skills and the accuracy of the final predicted skill sets, we define two categories of metrics. Let G_i denote the non-empty gold skill set for query i, and T_i^K denote the top-K unique skill IDs retrieved by the model.

Ranking Metrics. Ranking metrics are macro-averaged across all n queries.

$$\text{Recall@K} = \frac{1}{n}\sum_i \frac{|G_i \cap T_i^K|}{|G_i|}, \tag{44}$$

$$\text{Complete@K} = \frac{1}{n}\sum_i \mathbb{I}[G_i \subseteq T_i^K]. \tag{45}$$

Mean Reciprocal Rank (MRR@K) averages $\text{RR@K}_i = \mathbb{I}[r_i \le K]/r_i$, where r_i is the rank of the first relevant skill. Mean Average Precision (MAP@K) is defined using truncated precision $P_i(j) = |G_i \cap T_i^j|/j$:

$$\text{AP@K}_i = \frac{\sum_{j=1}^{\min(K,|\pi_i|)} P_i(j)\mathbb{I}[\pi_{ij} \in G_i]}{\min(K, |G_i|)}, \qquad \text{MAP@K} = \frac{1}{n}\sum_i \text{AP@K}_i\,. \tag{46}$$

Normalized Discounted Cumulative Gain (nDCG@K) measures ranking quality with explicit ideal normalization:

$$\text{DCG@K}_i = \sum_{j=1}^{\min(K,|\pi_i|)} \frac{\mathbb{I}[\pi_{ij} \in G_i]}{\log_2(j+1)}, \tag{47}$$

$$\text{IDCG@K}_i = \sum_{j=1}^{\min(K,|G_i|)} \frac{1}{\log_2(j+1)}, \tag{48}$$

$$\text{nDCG@K} = \frac{1}{n}\sum_i \frac{\text{DCG@K}_i}{\text{IDCG@K}_i}. \tag{49}$$

Set Prediction Metrics. Since our final output is a set of Skill IDs $\widehat{G}_i$ rather than a ranked list, we evaluate set-level exactness and overlap. Let $\text{TP}_i = |G_i \cap \widehat{G}_i|$. We define Precision, Recall, and Sample-F1 as:

$$P_i = \frac{\text{TP}_i}{|\widehat{G}_i|}, \qquad R_i = \frac{\text{TP}_i}{|G_i|}, \qquad \text{Sample-F1}_i = \frac{2\text{TP}_i}{|G_i| + |\widehat{G}_i|}. \tag{50}$$

If the predicted set is empty, we strictly define $P_i = 0$ and $\text{Sample-F1}_i = 0$. While Sample-F1 is averaged per query, Micro-F1 aggregates the total true positives, false positives ($\text{FP}_i = |\widehat{G}_i \setminus G_i|$), and false negatives ($\text{FN}_i = |G_i \setminus \widehat{G}_i|$) across the entire dataset:

$$\text{Micro-F1} = \frac{2\sum_i \text{TP}_i}{2\sum_i \text{TP}_i + \sum_i \text{FP}_i + \sum_i \text{FN}_i}. \tag{51}$$

Exact Match (EM) strictly measures the proportion of queries where $\widehat{G}_i$ equals G_i, regardless of internal topological ordering.

Candidate Recall Bounds and Oracles. Our modular framework utilizes a retrieve-then-select pipeline. Let C_i be the candidate pool constructed before LCEA evidence aggregation and Beam Search decoding. The number of covered gold skills is $c_i = |G_i \cap C_i|$.

If a gold skill is missing from C_i (i.e., $c_i < |G_i|$), it is permanently lost; subsequent LCEA and dependency-aware decoding modules cannot recover it. Therefore, for a predicted target set size k ($1 \le k \le |C_i|$), the theoretical maximum Sample-F1 strictly bounded by the candidate pool is:

$$\text{Sample-F1}_i^{\text{oracle}}(k) = \frac{2\min(k, c_i)}{k + |G_i|}. \tag{52}$$

This oracle metric explicitly isolates global/local recall errors from candidate-internal decoding errors. It provides a strict performance upper bound to analyze whether bottlenecks stem from initial candidate retrieval or from the downstream cardinality estimation and Beam Search selection.

E ADDITIONAL RESULTS AND DIAGNOSTICS

This section provides a comprehensive evaluation of our method's retrieval and ranking capabilities, alongside diagnostic analyses of set selection, training efficiency, and error attribution. These supplementary results further validate the robustness and superior performance of our proposed framework across diverse multi-skill benchmarks.

E.1 FIXED-CUTOFF RETRIEVAL AND RANKING

Candidate coverage. Table 7 measures the fraction of relevant skills successfully retrieved within the top K candidates. High recall is critical for multi-skill tasks, as the downstream dependency-aware decoding strictly relies on the quality of this candidate pool. Our method achieves state-of-the-art average recall scores of 48.2%, 74.3%, and 82.6% at $K \in \{1, 5, 10\}$, respectively. While dataset-specific dense retrievers occasionally perform well on their source datasets (e.g., ToolBench retriever at Recall@5), our framework consistently generalizes across all benchmarks, delivering the most robust and complete candidate pools for composite queries.

Table 7: Retrieval recall at $K \in \{1, 5, 10\}$ (%). The best and second-best results are highlighted.

Method	R3-Skill			SkillRet			ToolBench			ToolRet			MCP-Bench			Avg		
	Recall @1	Recall @5	Recall @10	Recall @1	Recall @5	Recall @10	Recall @1	Recall @5	Recall @10	Recall @1	Recall @5	Recall @10	Recall @1	Recall @5	Recall @10	Recall @1	Recall @5	Recall @10
Traditional retrieval																		
BM25	13.3	27.3	31.8	25.0	41.7	47.8	5.2	21.9	31.4	7.0	27.4	28.5	27.4	36.0	39.4	15.6	30.9	35.8
Pretrained retrievers																		
Qwen3 embedding 8B	29.6	56.0	67.8	25.1	45.9	51.0	19.7	47.4	57.7	22.7	51.9	70.2	64.9	81.5	88.5	32.4	56.5	67.0
R3 embedding	32.6	68.2	71.8	37.1	79.8	85.6	14.7	48.7	63.2	17.2	32.0	34.3	59.4	78.0	85.4	32.2	54.6	61.1
SkillRet	30.7	58.4	68.7	36.8	75.8	82.5	11.6	46.6	61.1	18.2	30.5	33.3	64.0	81.0	87.0	32.3	58.5	66.5
ToolBench retriever	5.8	13.1	16.4	13.0	22.0	27.3	21.1	**61.8**†	79.3	13.4	23.8	25.6	38.0	49.9	54.6	18.3	34.1	40.6
ToolRet E5-large	17.6	31.1	36.9	29.1	47.0	54.1	13.5	40.3	55.9	17.3	37.4	46.2	52.6	69.0	75.6	26.0	45.0	53.7
ToolRet BGE-large	15.3	25.6	32.7	29.0	48.4	57.1	23.8	59.4	70.4	13.5	28.2	34.0	52.7	69.2	75.8	26.9	46.2	54.0
COLT	14.9	25.1	32.9	21.0	31.8	39.7	12.1	37.9	51.0	15.3	27.6	29.8	27.5	36.1	39.6	18.2	31.7	38.6
Reranking systems																		
SkillRouter	32.1	60.7	67.3	41.3	71.0	73.7	17.4	55.7	64.5	19.5	33.4	38.1	17.1	22.5	24.6	25.5	48.7	53.6
R3	23.3	63.3	71.5	26.8	71.7	80.4	18.8	56.6	65.8	17.4	28.0	35.4	18.8	24.7	27.0	21.0	48.9	56.0
SkillRet + reranker	32.5	68.0	72.0	36.5	74.4	81.6	15.3	52.7	64.6	20.2	33.0	40.1	59.5	78.2	85.6	32.8	61.3	68.8
Reranker 0.6B	37.3	67.1	71.1	33.2	53.7	56.8	17.6	53.6	65.1	26.5	53.8	71.5	66.7	83.1	89.2	36.3	62.3	70.7
Adaptation methods																		
Re-Invoke	25.8	55.1	63.3	29.9	54.7	62.2	12.4	42.5	57.8	21.1	47.4	53.9	51.8	68.1	74.6	28.2	53.6	62.4
ToolDreamer	9.8	18.4	26.9	14.7	26.0	32.3	12.6	35.2	45.0	25.2	53.5	63.3	46.9	61.5	67.4	21.8	38.9	47.0
ToolQP	24.0	51.1	59.8	28.7	53.3	61.3	11.7	40.6	58.0	21.2	47.5	54.0	52.8	69.3	75.9	27.7	52.4	61.8
COLT-style	14.2	32.0	40.4	18.9	44.6	50.8	9.2	34.0	45.3	12.9	48.1	49.0	37.7	49.5	54.2	18.6	41.6	47.9
MFTR-style	18.4	33.3	39.8	27.7	46.0	51.7	11.5	31.1	40.5	7.9	26.4	30.5	48.0	63.0	69.0	22.7	40.0	46.3
Tool-REX	19.3	32.4	41.3	36.6	65.7	73.0	8.8	29.9	39.1	8.0	26.5	28.6	43.4	57.0	62.4	23.2	42.3	48.9
ToolQP-REX	24.1	51.2	59.9	36.7	65.8	73.1	11.8	40.7	58.1	21.3	47.6	54.1	43.6	57.2	62.6	27.5	52.5	61.6
Our system																		
Ours	**45.1**†	**70.2**†	**72.5**†	**46.5**†	**84.1**†	**86.8**†	**35.2**†	59.3	**81.5**†	**38.6**†	**67.3**†	**80.2**†	**75.8**†	**90.5**†	**92.0**†	**48.2**†	**74.3**†	**82.6**†

Precision throughout the ranking. Table 8 reports Mean Average Precision (MAP), which explicitly accounts for the positional rank of relevant entries within the prefix. By surfacing the necessary skills earlier in the ranking, a system significantly reduces the computational overhead required for downstream evidence aggregation and decoding. Our method achieves impressive cross-dataset MAP averages of 77.8%, 69.8%, and 72.0% across the evaluated cutoffs, substantially outperforming all compared adaptation baselines and ranking systems.

Table 8: Mean average precision at $K \in \{1, 5, 10\}$ (MAP@K, %).

Method	R3-Skill			SkillRet			ToolBench			ToolRet			MCP-Bench			Avg		
	MAP @1	MAP @5	MAP @10	MAP @1	MAP @5	MAP @10	MAP @1	MAP @5	MAP @10	MAP @1	MAP @5	MAP @10	MAP @1	MAP @5	MAP @10	MAP @1	MAP @5	MAP @10
Traditional retrieval																		
BM25	28.0	21.3	22.1	56.0	35.0	36.4	10.7	13.0	14.8	17.3	18.1	18.7	27.4	28.0	28.5	27.9	23.1	24.1
Pretrained retrievers																		
Qwen3 embedding 8B	61.3	44.9	47.5	54.0	36.5	37.7	42.0	35.8	38.3	61.5	44.9	50.9	64.9	65.5	66.0	56.7	45.5	48.1
R3 embedding	74.6	63.8	65.7	85.1	70.0	71.8	30.7	31.4	35.0	46.2	27.5	28.0	59.4	60.0	61.2	59.2	50.5	52.3
SkillRet	64.0	49.2	51.5	84.0	65.5	67.5	24.7	28.8	32.5	50.0	28.1	28.7	64.0	65.1	66.3	57.3	47.3	49.3
ToolBench retriever	12.0	8.9	9.4	28.0	17.0	17.9	47.3	44.4	49.2	38.5	21.4	22.3	38.0	39.2	40.5	32.8	26.2	27.9
ToolRet E5-large	36.0	25.9	27.2	64.0	39.9	41.3	29.3	26.4	30.5	48.1	29.7	31.7	52.6	53.8	54.9	46.0	35.1	37.1
ToolRet BGE-large	32.0	20.1	21.4	64.1	40.6	42.6	52.7	44.7	47.7	38.6	23.4	24.6	52.7	54.0	55.2	48.0	36.6	38.3
COLT	30.7	20.3	21.8	46.7	26.1	27.8	25.3	25.0	28.1	42.3	25.7	26.0	27.5	28.5	29.6	34.5	25.1	26.7
Reranking systems																		
SkillRouter	77.3	56.0	57.4	85.8	65.3	66.0	38.7	37.5	40.1	51.9	29.0	30.1	17.1	18.0	18.5	54.2	41.2	42.4
R3	48.0	46.7	49.2	58.0	54.1	57.7	40.7	38.6	41.3	46.3	25.3	26.7	18.8	19.5	20.2	42.4	36.8	39.0
SkillRet + reranker	72.0	55.9	57.8	82.0	64.9	67.6	33.3	34.3	37.4	53.9	29.9	31.3	59.5	60.5	61.8	60.1	49.1	51.2
Reranker 0.6B	74.8	56.9	57.9	72.0	48.0	48.7	39.3	36.5	39.4	**73.1**†	51.5	53.8	66.7	**67.5**†	68.2	65.2	52.1	53.6
Adaptation methods																		
Re-Invoke	53.3	41.0	43.1	64.7	45.0	46.8	26.7	27.8	31.3	59.6	40.0	41.6	51.8	52.9	54.1	51.2	41.3	43.4
ToolDreamer	20.0	13.4	14.7	31.3	19.1	20.2	27.3	23.7	25.9	71.2	50.8	52.4	46.9	48.2	49.5	39.3	31.0	32.5
ToolQP	49.3	38.9	41.0	62.0	44.4	46.1	25.4	26.9	30.7	59.7	41.3	43.0	52.8	54.2	55.6	49.8	41.1	43.3
COLT-style	29.3	22.9	24.5	41.3	32.3	33.8	19.3	23.0	25.3	36.5	39.1	39.2	37.7	38.9	40.1	32.8	31.2	32.6
MFTR-style	38.7	26.0	27.3	60.7	38.8	40.2	25.5	21.9	24.0	19.2	19.8	20.8	48.0	49.5	50.8	38.4	31.2	32.6
Tool-REX	40.0	27.3	29.2	80.0	56.9	58.8	18.7	18.4	20.4	19.3	20.8	21.6	43.4	44.6	45.9	40.3	33.6	35.2
ToolQP-REX	49.4	39.0	41.1	80.1	57.0	58.9	25.6	27.0	30.8	59.8	41.4	43.1	43.6	44.8	46.1	51.7	41.8	44.0
Our system																		
Ours	**84.1**†	**71.5**†	**74.3**†	**96.1**†	**81.3**†	**85.5**†	**61.3**†	**53.6**†	**56.4**†	71.6	**66.1**†	**66.8**†	**75.8**†	76.5	**77.2**†	**77.8**†	**69.8**†	**72.0**†

First relevant result. Table 9 reports Mean Reciprocal Rank (MRR), effectively measuring a system's ability to instantly expose at least one highly applicable skill. This is a crucial metric for evaluating the initial precision of the retrieval process. Our method exhibits exceptional intent-matching capabilities, achieving top average MRR values of 77.8%, 79.5%, and 80.5%, confirming that the most relevant skills are consistently surfaced at the very top of the ranking.

Table 9: Mean reciprocal rank at $K \in \{1, 5, 10\}$ (MRR@K, %).

Method	R3-Skill			SkillRet			ToolBench			ToolRet			MCP-Bench			Avg		
	MRR @1	MRR @5	MRR @10	MRR @1	MRR @5	MRR @10	MRR @1	MRR @5	MRR @10	MRR @1	MRR @5	MRR @10	MRR @1	MRR @5	MRR @10	MRR @1	MRR @5	MRR @10
Traditional retrieval																		
BM25	28.0	34.2	35.0	56.0	62.6	63.4	10.7	22.4	24.0	17.3	24.6	24.6	27.4	29.2	29.8	27.9	34.6	35.4
Pretrained retrievers																		
Qwen3 embedding 8B	61.3	72.5	73.3	54.0	65.1	65.7	42.0	53.1	54.4	61.5	66.4	66.8	64.9	69.2	70.8	56.7	65.3	66.2
R3 embedding	74.6	84.5	84.7	85.1	86.1	86.3	30.7	47.4	48.5	46.2	48.2	48.7	59.4	63.2	64.6	59.2	65.9	66.6
SkillRet	64.0	71.3	72.4	84.0	89.0	89.3	24.7	44.3	45.4	50.0	51.4	52.0	64.0	68.0	69.6	57.3	64.8	65.7
ToolBench retriever	12.0	16.8	17.4	28.0	35.0	36.3	47.3	62.7	**65.1**†	38.5	40.7	40.7	38.0	40.4	41.3	32.8	39.1	40.2
ToolRet E5-large	36.0	41.6	42.3	64.0	70.6	71.7	29.3	42.1	43.4	48.1	54.5	55.7	52.6	55.8	57.2	46.0	52.9	54.1
ToolRet BGE-large	32.0	37.0	37.8	64.1	73.0	73.8	52.7	**64.6**†	64.7	38.6	43.0	43.7	52.7	56.0	57.3	48.0	54.7	55.5
COLT	30.7	34.7	36.0	46.7	52.9	53.9	25.3	38.1	39.9	42.3	43.8	44.5	27.5	29.3	29.9	34.5	39.8	40.8
Reranking systems																		
SkillRouter	77.3	77.8	78.5	85.8	92.9	92.9	38.7	57.3	57.6	51.9	55.9	56.9	17.1	18.2	18.6	54.2	60.4	60.9
R3	48.0	63.0	63.7	58.0	73.5	74.1	40.7	55.9	56.4	46.3	48.5	49.5	18.8	20.0	20.4	42.4	52.2	52.8
SkillRet + reranker	72.0	79.7	80.1	82.0	88.1	88.3	33.3	51.4	52.0	53.9	56.1	57.4	59.5	63.3	64.8	60.1	67.7	68.5
Reranker 0.6B	74.8	79.2	79.4	72.0	77.9	78.2	39.3	55.6	56.4	**73.1**†	75.3	75.6	66.7	70.9	72.6	65.2	71.8	72.4
Adaptation methods																		
Re-Invoke	53.3	64.7	65.1	64.7	73.3	74.0	26.7	42.0	43.7	59.6	66.4	67.1	51.8	55.1	56.4	51.2	60.3	61.3
ToolDreamer	20.0	26.1	27.8	31.3	40.5	41.9	27.3	37.9	38.9	71.2	75.0	75.6	46.9	49.8	51.0	39.3	45.9	47.0
ToolQP	49.3	61.7	62.1	62.0	71.4	72.5	25.4	40.6	42.8	59.7	67.6	68.5	52.8	56.1	57.4	49.8	59.5	60.7
COLT-style	29.3	37.8	38.9	41.3	54.3	54.8	19.3	30.3	31.8	36.5	43.1	43.4	37.7	40.0	41.0	32.8	41.1	42.0
MFTR-style	38.7	43.4	44.2	60.7	67.1	67.6	25.5	35.4	36.6	19.2	32.2	33.1	48.0	51.0	52.2	38.4	45.8	46.7
Tool-REX	40.0	44.8	45.8	80.0	85.3	85.5	18.7	29.0	30.5	19.3	32.7	33.0	43.4	46.2	47.2	40.3	47.6	48.4
ToolQP-REX	49.4	61.8	62.2	80.1	85.4	85.6	25.6	40.7	42.9	59.8	67.7	68.6	43.6	46.4	47.4	51.7	60.4	61.3
Our system																		
Ours	**84.1**†	**86.5**†	**87.2**†	**96.1**†	**96.8**†	**97.0**†	**61.3**†	62.0	63.5	71.6	**75.0**†	**76.2**†	**75.8**†	**77.0**†	**78.5**†	**77.8**†	**79.5**†	**80.5**†

Rank-discounted relevance. Table 10 measures ranking quality comprehensively by weighing relevant entries against an ideal ranking (nDCG). Our framework consistently dominates this metric with cross-dataset averages of 77.8%, 76.7%, and 79.8%, showcasing its robustness in managing complex, multi-skill relevance distributions better than both standalone pretrained models and competing reranking baselines.

Table 10: Normalized discounted cumulative gain at $K \in \{1, 5, 10\}$ (nDCG@K, %).

Method	R3-Skill			SkillRet			ToolBench			ToolRet			MCP-Bench			Avg		
	nDCG @1	nDCG @5	nDCG @10	nDCG @1	nDCG @5	nDCG @10	nDCG @1	nDCG @5	nDCG @10	nDCG @1	nDCG @5	nDCG @10	nDCG @1	nDCG @5	nDCG @10	nDCG @1	nDCG @5	nDCG @10
Traditional retrieval																		
BM25	28.0	26.0	27.8	56.0	43.2	45.7	10.7	17.9	21.8	17.3	22.6	23.1	27.4	30.2	31.6	27.9	28.0	30.0
Pretrained retrievers																		
Qwen3 embedding 8B	61.3	54.5	59.1	54.0	45.8	47.9	42.0	43.3	47.8	61.5	51.9	60.1	64.9	71.6	74.9	56.7	53.4	58.0
R3 embedding	74.6	70.8	74.4	85.1	**77.6**†	80.1	30.7	40.4	46.5	46.2	33.2	34.1	59.4	65.4	68.4	59.2	57.5	60.7
SkillRet	64.0	57.0	61.2	84.0	69.8	73.2	24.7	37.8	43.9	50.0	33.6	34.8	64.0	70.4	73.6	57.3	53.7	57.3
ToolBench retriever	12.0	12.0	13.3	28.0	22.5	24.8	47.3	54.0	61.4	38.5	26.2	27.2	38.0	41.8	43.7	32.8	31.3	34.1
ToolRet E5-large	36.0	31.0	33.4	64.0	48.8	51.8	29.3	34.2	40.8	48.1	37.3	41.1	52.6	57.8	60.4	46.0	41.8	45.5
ToolRet BGE-large	32.0	25.5	28.3	64.1	50.3	53.8	52.7	53.7	58.3	38.6	28.9	31.2	52.7	57.9	60.6	48.0	43.3	46.4
COLT	30.7	25.0	28.1	46.7	33.7	36.9	25.3	32.0	37.6	42.3	30.0	31.0	27.5	30.3	31.7	34.5	30.2	33.1
Reranking systems																		
SkillRouter	77.3	62.3	64.9	85.8	73.3	74.4	38.7	47.6	51.3	51.9	36.1	38.1	17.1	18.8	19.7	54.2	47.6	49.7
R3	48.0	55.4	59.7	58.0	64.0	69.4	40.7	48.0	52.1	46.3	31.2	34.0	18.8	20.7	21.6	42.4	43.9	47.4
SkillRet + reranker	72.0	64.8	67.8	82.0	73.2	77.1	33.3	43.9	48.9	53.9	36.3	39.2	59.5	65.5	68.5	60.1	56.7	60.3
Reranker 0.6B	74.8	65.5	69.1	72.0	56.4	57.7	39.3	46.1	51.0	**73.1**†	57.2	63.2	66.7	73.4	76.7	65.2	59.7	63.5
Adaptation methods																		
Re-Invoke	53.3	50.6	53.9	64.7	54.4	57.4	26.7	35.6	42.0	59.6	48.2	50.9	51.8	57.0	59.6	51.2	49.2	52.8
ToolDreamer	20.0	17.8	21.1	31.3	26.0	28.6	27.3	30.4	34.5	71.2	55.8	61.4	46.9	51.6	53.9	39.3	36.3	39.9
ToolQP	49.3	47.7	51.2	62.0	53.3	56.5	25.4	34.3	41.5	59.7	49.1	51.9	52.8	58.0	60.7	49.8	48.5	52.4
COLT-style	29.3	29.0	32.3	41.3	41.0	43.6	19.3	28.1	32.7	36.5	43.1	43.5	37.7	41.4	43.3	32.8	36.5	39.1
MFTR-style	38.7	32.1	34.6	60.7	47.3	49.6	25.5	27.8	31.7	19.2	24.7	26.5	48.0	52.8	55.2	38.4	36.9	39.5
Tool-REX	40.0	32.8	36.4	80.0	65.9	68.9	18.7	24.3	28.2	19.3	25.4	26.4	43.4	47.8	50.0	40.3	39.2	42.0
ToolQP-REX	49.4	47.7	51.2	80.1	65.9	68.9	25.6	34.3	41.5	59.8	49.1	51.9	43.6	48.0	50.2	51.7	49.0	52.7
Our system																		
Ours	**84.1**†	**83.1**†	**85.3**†	**96.1**†	83.2	**87.2**†	**61.3**†	**65.4**†	**67.5**†	71.6	**71.3**†	**76.5**†	**75.8**†	**80.6**†	**82.5**†	**77.8**†	**76.7**†	**79.8**†

E.2 Returned-Set Size and Supervision

Fixed selection budgets. Figure 9 clearly illustrates the precision-recall trade-off inherent in multi-skill retrieval. Across different datasets, the optimal fixed cutoff varies significantly (e.g., $k = 2$ for R3-Skill, SkillRet, ToolBench, and MCP; $k = 3$ for ToolRet). Forcing a static top-K selection inevitably introduces false positives once the dataset-specific optimum is surpassed, degrading final Sample-F1 (SS-F1). This strong empirical variance across datasets directly motivates and validates the necessity of our dynamic cardinality estimation and beam-search decoding modules, which effectively tailor the selection size to the unique requirements of each query.

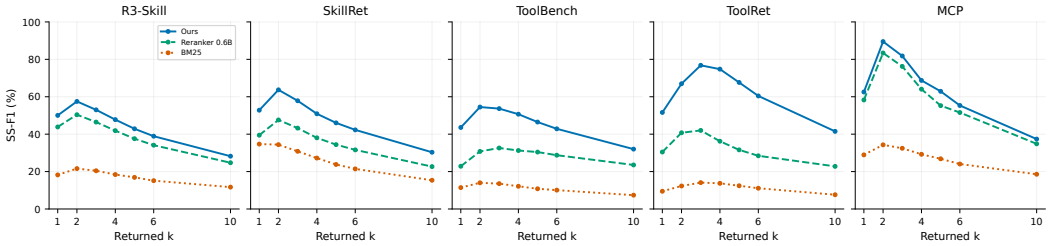


Figure 9: Final SS-F1 versus the number of returned skills. The varying peaks demonstrate that fixed-cutoff strategies are suboptimal across heterogeneous datasets.

E.3 Stratified Performance

Set cardinality and document length. Figure 10 breaks down system performance across different difficulty strata, specifically analyzing varying gold-set cardinalities and document lengths. Our method maintains a definitive edge over baselines like BM25 across both small and large target skill sets. Crucially, in the lower panel, our framework shows exceptional resilience to the long-document distraction problem. The performance remains consistently high even on strata containing excessively long skill descriptions, underscoring the efficacy of our Long-Context Evidence Aggregation (LCEA) mechanism in extracting localized evidence without succumbing to truncation or semantic dilution.

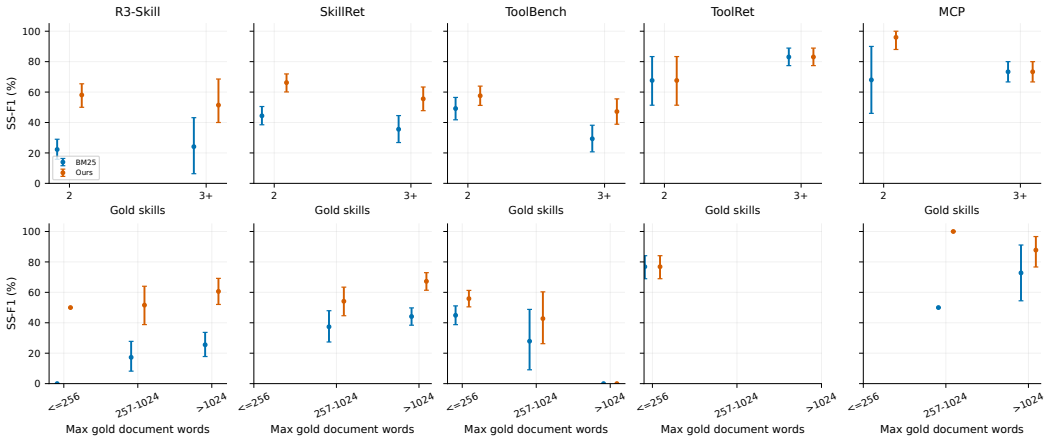


Figure 10: Stratified SS-F1 analysis by gold-set cardinality (top) and maximum gold-document length (bottom). The counts indicate the number of evaluated queries per stratum.