SKILLPROJ: SKILL RETRIEVAL BY PROJECTION ONTO REQUIREMENT SUBSPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Skill-based recursive self-improvement (RSI) relies on retrieving accumulated procedures as task requirements evolve. Whole-request encoding compresses multiple needs into one direction, while maximum per-requirement similarity retains only the strongest individual match. We introduce SkillProj, which represents complete active requirements by their vector span and retrieves skills through projection, requirement-level reranking, and residual diversity. We extend public SkillRouter, SkillRet, and R3-Skill resources into three trajectory benchmarks with 429 five-turn dialogues and libraries of up to 79,141 skills. With GPT-5.6 Luna, SkillProj achieves the highest Recall@10 and full coverage among eight methods on all three benchmarks. Averaged equally across benchmarks, its Recall@10 exceeds the strongest baseline by 8.16, 5.30, and 5.95 percentage points under GPT-5.6 Luna, DeepSeek V4.1 Flash, and GLM-5.3-Flash, respectively. With DeepSeek V4.1 Flash and GLM-5.3-Flash, it uses 33–34% fewer generation tokens than SkillDreamer. Agents using SkillProj also achieve the highest content accuracy in 1,128 end-to-end executions. Matched ablations support representing active requirements as a subspace and retaining requirement-level matching during reranking.

1 INTRODUCTION

Agents retain reusable procedures in skill libraries and use execution experience to refine them, as in Voyager and SkillOS (Wang et al., 2023; Ouyang et al., 2026). Recursive self-improvement (RSI) extends this idea to procedures governing improvement itself, including exploration policies in Dream-RSI (Zheng et al., 2026a). In skill-based RSI systems, accumulated procedures can support later tasks through retrieval that matches skills to the current requirements.

Later tasks may combine several needs, which users can add, withdraw, or replace during a conversation. SkillRouter and SkillRet provide models and benchmarks for request–skill matching (Zheng et al., 2026b; Kang et al., 2026). Other approaches improve matching through description expansion, document fields, and candidate refinement (Liu et al., 2026; Goulart et al., 2026; Xiao et al., 2026; Li et al., 2026). A skill may remain useful for one active need even when it shares little content with the rest of the request. Retrieval must preserve these distinct needs.

We argue that representing multiple active requirements is central to this retrieval problem. Encoding the complete request as one vector gives all skills a single matching direction, which can weaken the match to an individual requirement. Figure 1 examines this effect through the distances of target skills to the complete request and to individual requirements. Encoding requirements separately preserves their directions, but MaxSim scores a skill only by its strongest individual match. A joint representation can also account for components distributed across several requirement directions, motivating retrieval in the subspace spanned by the active requirements.

We introduce **SkillProj** to represent active requirements through their shared vector span. Projection retrieves skills aligned with individual or combined requirement directions; requirement-level reranking and residual selection refine the candidate set. We extend public skill resources into three trajectory benchmarks and evaluate eight methods under three language models. SkillProj leads mean Recall@10 in each model setting, with gains of 5.30–8.16 percentage points over the strongest baseline.

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.

Our contributions are:

- A requirement-subspace retrieval pipeline with projection scoring, requirement-level reranking, and residual set selection.
- Three trajectory benchmarks of evolving requirements from public resources: SkillRouter-Trajectory (SR), SkillRet-Trajectory (Ret), and R3-Skill-Trajectory (R3), with 143 base tasks and 429 terminal requests containing the final active requirements.
- Evidence that SkillProj improves mean recall across three language models, with matched ablations identifying subspace retrieval and requirement-level reranking as the main contributors.

2 Related Work

2.1 Skill Retrieval Benchmarks

Tool-use benchmarks evaluate whether agents can select and invoke external capabilities, from multi-tool instructions in ToolBench to executable dialogues in API-Bank (Qin et al., 2024; Li et al., 2023). ToolRet separates retrieval from execution and unifies heterogeneous candidate resources (Shi et al., 2025). For skill libraries, SkillRouter and SkillRet study request–skill matching, while R3-Skill includes supervision for query-conditioned skill compatibility (Zheng et al., 2026b; Kang et al., 2026; Wang et al., 2026). Skill Retrieval Augmentation examines retrieved procedures in downstream task solving (Su et al., 2026). These resources support evaluation of relevant skills and their use. Conversational retrieval adds another dimension: resolving a query from its dialogue history, as in QReCC (Anantha et al., 2021). We extend public skill resources with explicit requirement additions, withdrawals, and replacements, and evaluate retrieval against the requirements that remain active at each terminal state.

2.2 Query and Skill Representations

Two lines of work improve retrieval representations. One changes the query–document interaction: global vectors in DPR and token-level matching in ColBERT expose different matching units (Karpukhin et al., 2020; Khattab & Zaharia, 2020), while subspace representations score membership in the span of multiple inputs (Ishibashi et al., 2024). The other enriches the text being matched: HyDE, Re-Invoke, and SkillDreamer generate retrieval-oriented descriptions, whereas Tool-REX and field-aware retrieval expose document structure (Gao et al., 2023; Chen et al., 2024; Liu et al., 2026; Lu et al., 2026; Goulart et al., 2026). SkillSight further calibrates shared descriptive background (Xiao et al., 2026). Candidate selection addresses a separate issue. ToolRerank and SkillFlow refine relevance through staged reranking, while DPPs and DSR account for similarity within the selected set (Zheng et al., 2024; Li et al., 2026; Kulesza & Taskar, 2012; Chen et al., 2018; Wei et al., 2026). Our proposed SkillProj represents active requirements by the subspace spanned by their vectors. Projection scores candidate skills against this shared space, and reranking evaluates each candidate against the complete text of each requirement.

3 Requirement Subspaces for Skill Retrieval

Skill retrieval in an evolving conversation begins by determining which requirements remain active. These requirements define the information that the retrieval representation must preserve. We first describe the trajectories used to evaluate this setting, then develop a shared requirement space for candidate retrieval and use textual relevance and residual diversity to select the final skills.

3.1 Benchmark Construction under Evolving Requirements

We extend static skill retrieval tasks from public SkillRouter, SkillRet, and R3-Skill resources into requirement trajectories (Zheng et al., 2026b; Kang et al., 2026; Wang et al., 2026). Each source task retains its business context and follows three five-turn paths with additions, withdrawals, replacements, and confirmations. We preserve the full skill libraries and check target annotations against skill documents at each requirement state. Retrieval is evaluated at the terminal state. Target IDs are

withheld; active requirements are maintained from the current dialogue prefix alone. The benchmarks contain 143 base tasks, yielding 429 terminal requests and 2,145 dialogue prefixes (Table 1).

Table 1: Trajectory benchmark counts. A dialogue prefix contains all turns up to a given turn; task families group base tasks that share target skills.

Benchmark	Terminal requests	Dialogue prefixes	Skills	Task families
SkillRouter-Trajectory	153	765	79,141	45
SkillRet-Trajectory	126	630	6,006	42
R3-Skill-Trajectory	150	750	2,050	45

For a skill library $\mathcal{S}$, let $R_t = \{r_1, \ldots, r_m\}$ denote the complete requirements active at turn t. Requirement maintenance updates R_t from the dialogue prefix and produces the complete request Q_t. Methods use the same public dialogues and evaluation targets, with retrieval representations defined by their respective pipelines. Appendix F.3 specifies state construction for each language model. The terminal active set defines which skills are needed, including those serving retained requirements and excluding those serving only withdrawn work.

3.2 Motivation for Requirement Subspaces

A complete request can contain several distinct requirements, yet its embedding gives every skill the same matching direction (Karpukhin et al., 2020). To examine this representation, Figure 1 compares each target skill's cosine distance to the complete-request embedding with its distance to the closest separately encoded requirement. The plot includes all 308 target pairs across the 153 terminal requests in SkillRouter-Trajectory.

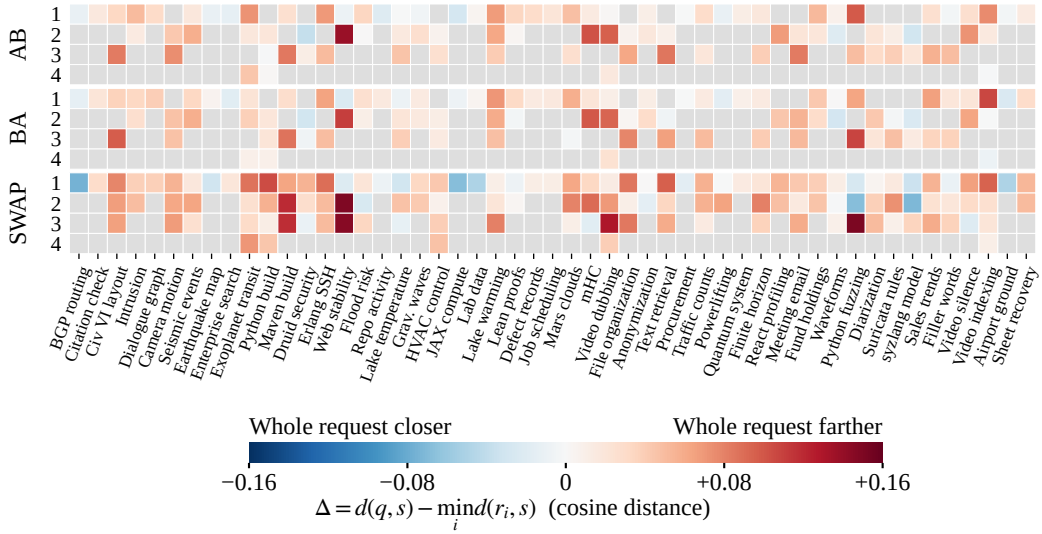


Figure 1: Target-distance shifts under whole-request encoding. Columns are SR tasks; rows index reference skills within each task and path, independently of retrieval ranks. Red/blue indicate greater/smaller distance to the whole-request vector than to the nearest requirement; gray denotes no target. All 308 pairs are retained. AB and BA add requirements A and B in opposite orders before withdrawing A; SWAP replaces A with B and then restores A. Appendix D.4 gives the full task names and diagnostic protocol.

Red cells in Figure 1 identify targets farther from the complete-request vector than from their closest requirement. The shifts vary even within the same request, motivating separate requirement encodings. The next design choice is how to combine these separate encodings into one retrieval score. The requirement span preserves both individual and combined directions, while an orthonormal basis

accounts for correlations. Section 5 compares this representation with whole-request and MaxSim retrieval.

3.3 Process of Requirement-Subspace Retrieval

Step 1: Requirement encoding. A frozen encoder E maps each complete requirement to $q_i = E(r_i)/\|E(r_i)\|_2$ and each skill document to $v_s = E(s)/\|E(s)\|_2$. Requirements retain their requested functions, output formats, and business constraints. Skill vectors are cached for reuse across requests.

Step 2: Subspace construction. Requirement vectors can contain overlapping semantic components. Constructing an orthonormal basis expresses their shared space through mutually orthogonal directions, so projection measures the portion of a skill vector contained in that space. We stack the vectors as $A = [q_1, \ldots, q_m]$ and apply SVD to obtain an orthonormal basis B for the retained subspace U. Directions with $\sigma_j > \max(\epsilon_{\text{abs}}, \epsilon_{\text{rel}}\sigma_1)$ are retained; the tolerances in Section 4.1 exclude numerically negligible directions.

Step 3: Membership retrieval. We use projection-based membership (Ishibashi et al., 2024) to score every skill:

$$M(s \mid R_t) = \|B^\top v_s\|_2^2 = v_s^\top P_U v_s, \qquad P_U = BB^\top. \tag{1}$$

For a unit skill vector, $M \in [0, 1]$ is the fraction of squared length captured by the requirement subspace. The top K scores form C_t, with skill-ID ordering to resolve ties. Appendix E gives the equivalent Gram-matrix expression and computational costs.

Figure 2 summarizes the pipeline: subspace retrieval supplies K candidates, and requirement-level reranking with residual selection returns k skills.

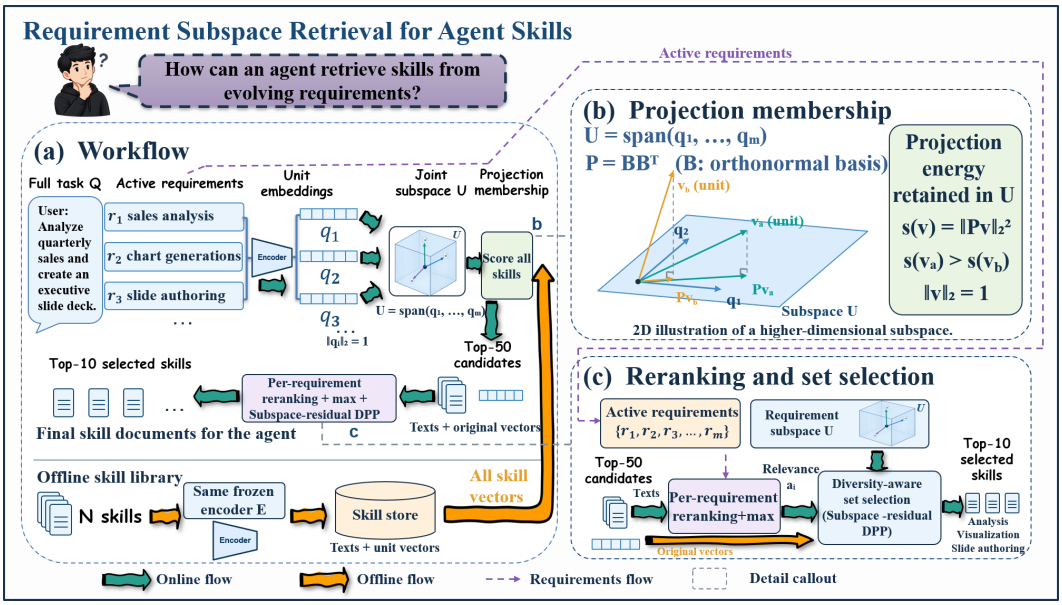


Figure 2: Overview of SkillProj. (a) Active requirements are encoded separately to form a shared subspace for full-library retrieval. (b) Projection energy determines the top K candidates. (c) Requirement-level relevance and subspace-residual diversity guide selection of k skills. Numerical budgets in the illustration correspond to the experimental setting, $K = 50$ and $k = 10$.

3.4 Properties of Requirement-Subspace Scoring

An orthogonal change of basis leaves P_U unchanged, so membership is invariant to the choice of orthonormal basis.

Unit skills aligned with one requirement or with a combination of orthogonal requirements can both have membership one; Appendix E compares their MaxSim scores. Neural reranking assesses textual relevance. Section 5 evaluates these retrieval representations under matched conditions.

3.5 Requirement-Level Reranking and Residual Selection

Projection forms a candidate pool aligned with the active requirement space. Candidates admitted by this geometric score still need to be assessed against the functions and constraints expressed in the requirement texts. The final list must also account for the relationships among the selected skills. We refine the candidate pool with requirement-level relevance scores, then select a complementary set using subspace-residual diversity.

Requirement-level reranking. We score each candidate against every complete active requirement using a frozen neural reranker. For a requirement–skill pair, the relevance probability $\rho(r_i, s) \in [0, 1]$ is obtained by normalizing the yes/no logits. The candidate's quality score is

$$a_s = \max_{r_i \in R_t} \rho(r_i, s), \qquad s \in C_t. \tag{2}$$

A specialized skill can therefore receive a high quality score by matching a single active requirement. Each pair includes the full requirement, including its requested function, output, and business constraints.

Residual set selection. Relevance scores evaluate candidates individually, so several highly scored skills can provide similar functionality. Set selection adds a preference for complementary candidates while retaining their relevance weights. We combine requirement-level quality with the query-residual selection principle of DSR (Wei et al., 2026) and the quality–diversity factorization of a DPP kernel (Kulesza & Taskar, 2012). For each original skill vector, we attenuate its requirement-subspace component and normalize the residual:

$$z_s = \frac{v_s - \lambda P_U v_s}{\|v_s - \lambda P_U v_s\|_2}, \quad S_{sj} = \frac{1 + z_s^\top z_j}{2}, \quad L_{sj} = \bar{a}_s S_{sj} \bar{a}_j, \tag{3}$$

where $\lambda \in [0, 1)$ controls attenuation and $\bar{a}_s = a_s / \max_j a_j$ uses a positive numerical floor for the denominator. Attenuating the shared requirement component lets the diversity score emphasize differences between candidates in the original embedding space. The resulting kernel is positive semidefinite; Appendix E gives its quality–diversity factorization. Greedy determinant selection (Chen et al., 2018) chooses k skills, which are returned in descending order of a_s.

4 Experiments

We evaluate whether the proposed representation improves retrieval of the active target set across different skill libraries. We then examine shorter prefixes of the returned lists, changes in the language model, generation usage, and downstream task completion. Controlled ablations in Section 5 connect these results to the retrieval and selection operations.

4.1 Experimental Setup

Benchmarks. We evaluate all methods on the same 153, 126, and 150 terminal requests from SR, Ret, and R3, respectively. Public dialogue inputs, full source libraries, and terminal target sets are shared; Section 3.1 describes their construction.

Baselines. We use Dense, SkillRouter (Zheng et al., 2026b), DSR (Wei et al., 2026), SkillSight (Xiao et al., 2026), SkillDreamer (Liu et al., 2026), Field-Aware Hybrid-Uniform (Goulart et al., 2026), and SkillFlow (Li et al., 2026). Hybrid-Uniform averages six sparse/dense cosine scores over names, descriptions, and bodies (Field-Aware in tables). SkillFlow retains variable-length outputs; SkillDreamer returns 15 candidates, with the first ten evaluated.

Settings. Methods share the frozen SkillRouter 0.6B dense encoder and retain their own ranking stages. GPT-5.6 Luna uses medium reasoning; DeepSeek V4.1 Flash and GLM-5.3-Flash use high reasoning for independent state and method-specific generation. SkillProj fixes $K = 50$, $k = 10$, $\lambda = 0.85$, and $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-6}$. Appendix F.3 specifies model protocols and token accounting.

Metrics. We report eight metrics at cutoff 10: precision, recall, F1, hit rate, full coverage, MRR, MAP, and nDCG. Recall measures how many target skills are retrieved; full coverage requires every target skill for a request. All metrics are macro-averaged over terminal requests and reported as percentages. Paired 95% intervals resample shared-skill families. Appendices C and F provide metric definitions and bootstrap details.

Execution metrics. Content accuracy counts confirmed correct answers out of 141 requests; unresolved contradictions in final answers or artifacts (Conflicts) are excluded from the correct count. All-check pass rate and mean score average the original task evaluator's binary verdicts and scores, including presentation checks. All-path success rate counts tasks correct on all three paths out of 47. Paired comparisons use 130 jointly resolved requests and 2,000 bootstrap replicates over 40 shared-skill families (Appendix F).

4.2 Main Results

Retrieval with GPT-5.6 Luna. We begin with a common language model to compare how the methods retrieve skills for the same terminal requirements. Recall and full coverage capture complementary aspects of this comparison: the fraction of targets recovered and the frequency with which the entire target set is available. Tables 2–4 report the eight retrieval metrics; Appendix F gives paired estimates and 95% intervals.

SkillRouter-Trajectory. SkillProj leads all eight metrics in Table 2, improving recall by 8.88 percentage points over DSR and fully covering 100 requests versus SkillDreamer's 86.

Table 2: SkillRouter-Trajectory (153 requests). Top-10 scores $\times 100$. MRR: mean reciprocal rank; MAP: mean average precision; nDCG: normalized discounted cumulative gain. Bold/underline: best/second best. Gain (pp): SkillProj minus best baseline, in percentage points.

Method	Precision	Recall	F1	Hit rate	Full coverage	MRR	MAP	nDCG
Dense	10.46	61.60	17.45	78.43	47.06	53.88	42.62	50.39
SkillRouter	12.68	70.21	20.98	86.93	54.25	74.34	58.14	65.29
DSR	<u>12.81</u>	<u>71.73</u>	<u>21.23</u>	<u>88.89</u>	54.90	<u>74.80</u>	58.50	<u>65.97</u>
SkillSight	10.85	61.11	17.97	76.47	45.75	58.38	45.55	52.90
SkillDreamer	12.55	70.21	20.80	84.31	<u>56.21</u>	63.76	51.66	59.65
Field-Aware	10.59	60.13	17.59	75.82	45.75	49.13	38.76	47.00
SkillFlow	11.05	59.75	18.19	75.16	42.48	73.86	<u>58.93</u>	62.72
SkillProj	**15.10**	**80.61**	**24.84**	**93.46**	**65.36**	**76.87**	**62.90**	**71.19**
Gain (pp)	+2.29	+8.88	+3.61	+4.58	+9.15	+2.07	+3.97	+5.23

SkillRet-Trajectory. The improvement carries over to Ret with the same pipeline and parameter settings. SkillProj leads all eight metrics in Table 3, gaining 12.96 recall points over SkillDreamer and increasing full coverage from 79 to 106 requests.

Table 3: SkillRet-Trajectory (126 requests). Top-10 scores $\times 100$; metrics and formatting follow Table 2.

Method	Precision	Recall	F1	Hit rate	Full coverage	MRR	MAP	nDCG
Dense	12.62	69.31	20.95	92.06	46.83	81.06	56.57	65.64
SkillRouter	14.68	79.37	24.32	97.62	59.52	88.45	<u>67.86</u>	<u>75.73</u>
DSR	14.68	79.37	24.32	<u>98.41</u>	58.73	<u>88.53</u>	67.82	<u>75.73</u>
SkillSight	14.68	78.97	24.31	96.83	59.52	84.71	64.69	73.23
SkillDreamer	<u>15.32</u>	<u>80.29</u>	<u>25.26</u>	95.24	<u>62.70</u>	86.90	66.62	75.18
Field-Aware	13.81	72.35	22.77	92.86	50.00	85.96	61.44	70.12
SkillFlow	6.35	31.75	10.43	59.52	10.32	57.94	30.82	36.96
SkillProj	**18.02**	**93.25**	**29.66**	**100.00**	**84.13**	**90.77**	**79.40**	**85.77**
Gain (pp)	+2.70	+12.96	+4.40	+1.59	+21.43	+2.24	+11.54	+10.04

R3-Skill-Trajectory. In Table 4, SkillProj reaches 79.56% Recall@10 and fully covers 100/150 requests. SkillSight reaches 78.67% (91 covered) and SkillDreamer 78.44% (90). SkillProj leads MAP and nDCG, which account for the ranks of multiple relevant skills.

Table 4: R3-Skill-Trajectory (150 requests). Top-10 scores $\times 100$; metrics and formatting follow Table 2.

Method	Precision	Recall	F1	Hit rate	Full coverage	MRR	MAP	nDCG
Dense	12.53	71.11	21.04	93.33	47.33	77.43	57.93	66.11
SkillRouter	13.53	76.67	22.72	94.00	58.67	**85.45**	66.42	73.68
DSR	13.40	76.00	22.50	93.33	58.00	85.38	66.33	73.45
SkillSight	13.80	78.67	23.18	**96.00**	60.67	81.22	63.16	71.62
SkillDreamer	14.00	78.44	23.49	95.33	60.00	82.37	64.29	72.45
Field-Aware	13.33	75.56	22.39	92.00	58.00	74.28	57.56	66.46
SkillFlow	7.67	47.44	13.01	67.33	28.67	65.56	46.56	51.09
SkillProj	**14.33**	**79.56**	**24.00**	92.67	**66.67**	85.03	**68.69**	**75.45**
Gain (pp)	+0.33	+0.89	+0.51	-3.33	+6.00	-0.42	+2.27	+1.76

Retrieval at different cutoffs. The top-ten comparison establishes the quality of the full returned list. Evaluating its shorter prefixes shows how much target coverage is already available when an agent consumes fewer of those skills. Figure 3 follows recall as each frozen final ranking is extended from one to ten skills. SkillProj leads Recall@3 and Recall@5 on all three benchmarks; at $k = 3$, recall reaches 63.78%, 79.89%, and 68.44% on SR, Ret, and R3, respectively. Together with the top-ten results, these curves show SkillProj's strength in covering active requirements with a short list of skills.

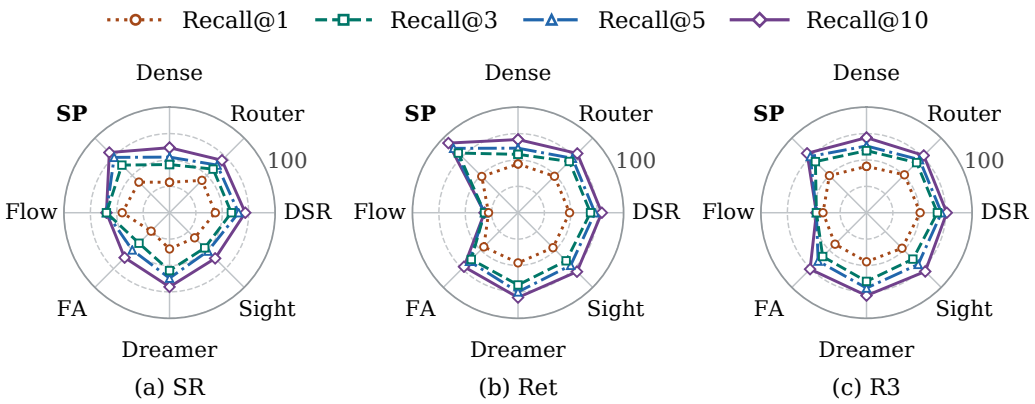


Figure 3: Recall (%) at prefixes of frozen final rankings; rings mark 25, 50, 75, and 100. Panels show SR, Ret, and R3; axes are the eight methods (FA: Field-Aware; SP: SkillProj; Router, Sight, Dreamer, and Flow omit the Skill prefix). Line styles and markers identify $k = 1, 3, 5, 10$. Each k evaluates a prefix of the same final output, with native short and empty lists retained.

Retrieval across language models. We next examine whether the pattern of method differences persists when the language model used to construct states and method-specific outputs changes. DeepSeek V4.1 Flash and GLM-5.3-Flash independently generate these inputs, and the eight retrieval methods are compared within each model setting. Table 5 shows that SkillProj leads mean Recall@10 under all three language models, exceeding SkillDreamer by 8.16, 5.30, and 5.95 points. With DeepSeek V4.1 Flash and GLM-5.3-Flash, it uses 33.4% and 34.4% fewer generation tokens than SkillDreamer: retrieval operates directly on the shared active requirements. SkillDreamer adds a skill-description generation stage. Figure 4 shows full coverage for every method and benchmark. SkillProj leads full coverage on SR and Ret with each language model. Appendix F.3 reports all scores and paired intervals.

Table 5: Retrieval and generation usage across three language models. R@10: equal-weight mean Recall@10 (%) over SR, Ret, and R3. Tok.: completed-response input plus output tokens, summed over all 429 five-turn trajectories and divided by 429, in thousands. Usage includes shared state and method-specific generation; local encoding, reranking, and accounting details appear in Appendix F.3.

	GPT-5.6 Luna (medium)		DeepSeek V4.1 Flash (high)		GLM-5.3-Flash (high)	
Method	R@10	Tok.	R@10	Tok.	R@10	Tok.
Dense	67.34	83.36	64.27	13.56	66.02	11.03
SkillRouter	75.41	83.36	70.74	13.56	72.55	11.03
DSR	75.70	83.36	71.24	13.56	72.24	11.03
SkillSight	72.92	83.36	65.81	13.56	68.38	11.03
SkillDreamer	76.31	91.44	72.85	20.36	74.28	16.81
Field-Aware	69.35	83.36	65.62	13.56	68.83	11.03
SkillFlow	46.31	99.26	48.66	23.37	30.06	20.80
SkillProj	**84.47**	83.36	**78.15**	13.56	**80.23**	11.03

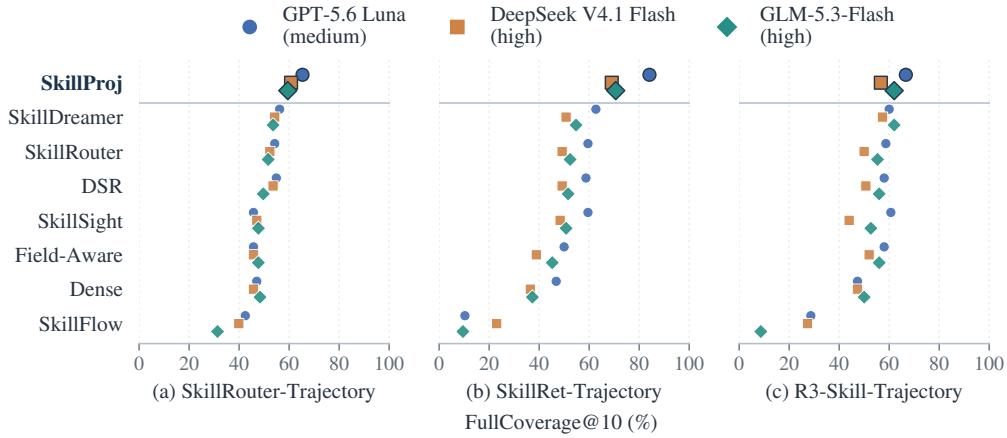


Figure 4: FullCoverage@10 across three benchmarks and language models. Each point is the fraction of requests whose complete target set appears in the top ten. All panels use the same scale and method order; outlined markers highlight SkillProj.

End-to-end setup. We compare eight methods on 47 executable SR tasks (141 requests, 1,128 executions). Executions use GPT-5.6 Luna (medium reasoning), fresh contexts, and fixed inputs, containers, and tool budgets. Methods supply full documents for at most ten skills, retaining native short and empty lists. Appendix F.2 details the execution protocol.

End-to-end task performance. SkillProj correctly completes 92/141 requests (65.25%), the highest content accuracy in Table 6, exceeding SkillSight by 2.13 percentage points. It also leads mean score (88.66) and ties SkillFlow on all-path success rate (24/47 tasks, 51.06%). Appendix F, Table 12, reports paired estimates with 95% intervals and the protocol sensitivity analysis.

5 Analysis

We compare four variants on the same 153 SR requests, keeping model weights, skill libraries, and budgets fixed ($K = 50$, $k = 10$). *Full-request retrieval* encodes the complete request as one vector and retrieves candidates by cosine similarity. *Per-requirement MaxSim retrieval* uses the same separately encoded requirements as SkillProj and scores each skill by its largest cosine similarity to any requirement. *Full-request reranking* retains the projection candidates and replaces maximum per-requirement neural relevance with one score against the complete request. *Direct top-10 selection*

Table 6: End-to-end task execution on SR: 141 requests and 47 tasks per method. Correct/Conflicts count correct submissions/unresolved final-answer contradictions; other columns are percentages. Metrics are defined in Section 4.1. Bold/underline: best/second best.

Method	Correct	Conflicts	Content accuracy	All-check pass rate	Mean score	All-path success rate
Dense	88	2	62.41	51.06	85.63	<u>48.94</u>
SkillRouter	86	2	60.99	51.06	86.84	44.68
DSR	88	2	62.41	**58.87**	87.31	<u>48.94</u>
SkillSight	89	2	<u>63.12</u>	55.32	86.54	46.81
SkillDreamer	88	1	62.41	56.03	<u>87.48</u>	44.68
Field-Aware	84	3	59.57	53.19	85.77	44.68
SkillFlow	85	0	60.28	51.06	86.88	**51.06**
SkillProj	**92**	2	**65.25**	<u>56.74</u>	**88.66**	**51.06**

retains projection and requirement-level reranking, then returns the ten highest-scoring candidates without DPP. Each variant changes one operation; the remaining stages follow the full pipeline.

Table 7 identifies subspace retrieval and requirement-level reranking as the main contributors on SR. Replacing projection with full-request or MaxSim retrieval reduces Recall@10 from 80.61% to 72.49% and 76.80%, respectively. The projection–MaxSim comparison holds the requirement embeddings, candidate budget, neural reranker, and residual selector fixed, directly testing how separately encoded requirements are combined. The 3.81-point gain over MaxSim (95% interval [0.65, 7.83]) therefore supports scoring against the joint requirement space beyond the benefit of requirement decomposition alone. With the projection candidates fixed, full-request reranking reduces recall to 72.98% and full coverage from 100 to 85 requests, showing the importance of retaining requirement-level matching when selecting from the candidate pool. Appendix F reports the paired estimates and uncertainty intervals.

Table 7: SR ablations with fixed models and budgets ($K = 50$, $k = 10$). Bold marks the changed operation. Per-req. max: maximum neural relevance over requirements. Recall is a percentage; full coverage counts requests out of 153.

Configuration	Retrieval	Reranking	Selection	Recall @50	Recall @10	Full coverage
SkillProj (full pipeline)	Projection	Per-req. max	DPP	84.10	80.61	100
Full-request retrieval	**Single query**	Per-req. max	DPP	75.00	72.49	85
Per-requirement MaxSim retrieval	**MaxSim**	Per-req. max	DPP	79.41	76.80	90
Full-request reranking	Projection	**Full request**	DPP	84.10	72.98	85
Direct top-10 selection	Projection	Per-req. max	**Top scores**	84.10	79.96	100

6 Conclusion

SkillProj represents active requirements by their vector span and retrieves skills through projection, requirement-level reranking, and residual selection. Across three trajectory benchmarks, it achieves the highest mean Recall@10 under each of three language models. Matched ablations show that jointly representing requirements during candidate retrieval and preserving requirement-level matching during reranking are the main sources of improvement. We hope to integrate SkillProj into skill-based recursive self-improvement (RSI) systems and explore how feedback-driven skill updates affect retrieval and subsequent task performance.

AI USE STATEMENT

Generative AI assisted benchmark construction, implementation, experiment planning, literature search, analysis, translation, and manuscript editing. The authors reviewed this work and take responsibility for the final content. Experimental language models are GPT-5.6 Luna, DeepSeek V4.1 Flash, and GLM-5.3-Flash, with configurations in Appendix F.3.

REPRODUCIBILITY STATEMENT

Equations 1–3 specify the retrieval and selection rules. Appendices A–F describe the data, prompts, evaluation, implementation, and supplementary results. The supplement provides the SkillProj pipeline, dialogue inputs, terminal requirements, and relevance labels for the three benchmarks, together with instructions and code for importing their complete candidate libraries from the original sources.

REFERENCES

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. Open-domain question answering goes conversational via question rewriting. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 520–534, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.44. URL https://aclanthology.org/2021.naacl-main.44/.

Laming Chen, Guoxin Zhang, and Eric Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

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 Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4705–4726, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.270. URL https://aclanthology.org/2024.findings-emnlp.270/.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.99. URL https://aclanthology.org/2023.acl-long.99/.

Paimon Goulart, Liang Wu, Kelly Wan, Evangelos E. Papalexakis, and Liangjie Hong. Field-aware agent skill retrieval, 2026. URL https://arxiv.org/abs/2608.02880.

Yoichi Ishibashi, Sho Yokoi, Katsuhito Sudoh, and Satoshi Nakamura. Subspace representations for soft set operations and sentence similarities. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3512–3524, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.192. URL https://aclanthology.org/2024.naacl-long.192/.

Ryangkyung Kang, Hongcheol Cho, and Youngeun Kim. Skillret: A large-scale benchmark for skill retrieval in llm agents, 2026. URL https://arxiv.org/abs/2605.05726.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie

Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pp. 39–48. ACM, July 2020. doi: 10.1145/3397271.3401075. URL http://dx.doi.org/10.1145/3397271.3401075.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2-3):123–286, December 2012. ISSN 1935-8245. doi: 10.1561/2200000044. URL http://dx.doi.org/10.1561/2200000044.

Fangzhou Li, Pagkratios Tagkopoulos, and Ilias Tagkopoulos. Skillflow: Scalable and efficient agent skill retrieval system, 2026. URL https://arxiv.org/abs/2504.06188.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-bank: A comprehensive benchmark for tool-augmented LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3102–3116, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.187. URL https://aclanthology.org/2023.emnlp-main.187/.

Shuo Liu, Yutong Yang, Haohao Xiao, Mouxing Yang, and Xi Peng. Imagine before retrieval: Prospective skill retrieval for llm agents, 2026. URL https://arxiv.org/abs/2609.01642.

Xuan Lu, Haohang Huang, Rui Meng, Yaohui Jin, Wenjun Zeng, and Xiaoyu Shen. Tools are under-documented: Simple document expansion boosts tool retrieval. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 141631–141653, 2026.

Siru Ouyang, Jun Yan, Yanfei Chen, Rujun Han, Zifeng Wang, Bhavana Dalvi Mishra, Rui Meng, Chun-Liang Li, Yizhu Jiao, Kaiwen Zha, Maohao Shen, Vishy Tirumalashetty, George Lee, Jiawei Han, Tomas Pfister, and Chen-Yu Lee. SkillOS: Learning skill curation for self-evolving agents, 2026. URL https://arxiv.org/abs/2605.06614.

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 B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9695–9717, 2024.

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 Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24497–24524, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.1258. URL https://aclanthology.org/2025.findings-acl.1258/.

Weihang Su, Jianming Long, Qingyao Ai, Qiaozhi He, Yichen Tang, Changyue Wang, Yiteng Tu, Yingbo Wang, and Yiqun Liu. Skill retrieval augmentation for agentic ai, 2026. URL https://arxiv.org/abs/2604.24594.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Zifei Wang, Wei Wen, Qiang Ji, Keyu Chen, Ruizhi Qiao, and Xing Sun. Skill is not document: Query-conditioned compatibility for llm agent skill routing, 2026. URL https://arxiv.org/abs/2606.03565.

Wang Wei, Tiankai Yang, Samyadeep Basu, Hongjie Chen, Yue Zhao, Zhengzhong Tu, Xiyang Hu, Franck Dernoncourt, Ryan A. Rossi, and Hoda Eldardiry. Beyond top-k skill retrieval: Diversity-aware skill routing for llm agents, 2026. URL https://arxiv.org/abs/2609.05824.

Jinying Xiao, Bin Li, Xiaopeng Li, Jianling Li, Jiacheng Jie, Xiaodong Liu, Ma Jun, Chao Wang, Nyima Tashi, and Jie Yu. Skillsight: Calibrating generic content bias for skill retrieval, 2026. URL https://arxiv.org/abs/2607.18785.

Tong Zheng, Xidong Wu, Zheng Zhang, Zhankui He, Chaoyi Zhang, Benjamin Coleman, Ruoqiao Wei, Di Bai, Haolin Liu, Rui Liu, Xue Wang, Yue Zhuan, Wang-Cheng Kang, Renkai Xiang, Heng Huang, Xinwu Cheng, and Yunsong Guo. Dream-RSI: Recursive self-improvement through evolving worlds. Technical report, Google, 2026a. URL https://www.dream-rsi.com/assets/dream-rsi.pdf.

YanZhao Zheng, ZhenTao Zhang, Chao Ma, YuanQiang Yu, JiHuai Zhu, Yong Wu, Tianze Xu, Baohua Dong, Hangcheng Zhu, Ruohui Huang, and Gang Yu. Skillrouter: Skill routing for llm agents at scale, 2026b. URL https://arxiv.org/abs/2603.22455.

Yuanhang Zheng, Peng Li, Wei Liu, Yang Liu, Jian Luan, and Bin Wang. ToolRerank: Adaptive and hierarchy-aware reranking for tool retrieval. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 16263–16273, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.1413/.

A Details of the Benchmarks

A.1 Dataset Sources and Task Composition

The three trajectory benchmarks comprise 143 base tasks with three five-turn paths each, yielding 429 terminal requests. They adapt public skill-use requests into controlled conversations and retain the full source retrieval libraries (Zheng et al., 2026b; Kang et al., 2026; Wang et al., 2026). The source collections and evaluated subsets are described below.

SkillRouter-Trajectory. We screened 87 source tasks and retained 75 under the source collection's default inclusion rules. The evaluated partition contains 51 tasks, 153 terminal requests, and 765 dialogue prefixes; the remaining 24 development tasks are outside the reported evaluation. Conversations are constructed in Chinese and preserve the source task's domain and technical constraints. The complete retrieval library contains 79,141 skill records. Tasks range from repository repair and formal reasoning to scientific analysis, document processing, and media editing.

SkillRet-Trajectory. We select a balanced sample of 42 requests from the source collection's 4,392 test requests, with three tasks in each of 14 domains: frontend, backend, databases, AI, cloud operations, security, testing, mobile development, systems analysis, graphics and games, educational content, business, research, and office collaboration. Each request is decomposed into its existing workstreams. The evaluation set was fixed before the retrieval runs and uses the full library of 6,006 skills with the same fixed method parameters. Nine tasks were previously used for benchmark construction and offline format validation. Terminal target sets vary across paths for 29 tasks and remain the same for 13.

R3-Skill-Trajectory. We select 50 tasks from 505 multi-skill requests within a source collection of 5,696 requests, preserving all 2,050 skill documents. The selected source requests comprise 41 with two original labels and nine with three. After adapting the requests and checking document support, the 150 terminal requests contain 38 single-target, 102 two-target, and ten three-target cases. Terminal targets change across paths for 26 tasks and remain the same for 24. Cancellations can leave a single target even when the source request originally required several skills.

A.2 Canonicalization of Dialogue and Requirement Fields

The three benchmarks use separate fields for standing context, public user turns, active requirements, and evaluation targets (Table 8). Context supplies shared inputs and constraints; explicit edits in

user turns determine which workstreams are active. Each active requirement preserves the complete requested work, including negation, quantities, alternatives, and exceptions.

Table 8: Common fields for controlled requirement trajectories. Evaluation targets are held separately from retrieval inputs.

Field	Meaning and use
Context	Standing task setting, available inputs, and shared constraints.
User turn	One public addition, withdrawal, replacement, restoration, or confirmation.
Active requirements	Complete work items retained after applying the current edit.
Maintained state	The language model's goal and current conditions, with change evidence and uncertainties.
Complete request	Context followed by the goal and retained conditions from the maintained state.
Target skills	Document-supported reference skills for the current scope; used only in evaluation.

Dialogue paths. Write b for a base requirement and A, B for controlled additions. AB visits b, $b + A$, $b + A + B$, $b + B$, and confirmation. BA visits b, $b + B$, $b + A + B$, $b + B$, and confirmation. SWAP visits b, $b + A$, $b + B$, $b + A + B$, and confirmation. Different paths provide distinct textual histories, and confirmation preserves the active requirements. Retrieval metrics are computed at the terminal state of each path.

From turns to retrieval inputs. In the Luna setting, public dialogue edits determine the active requirement texts R_t; each text is paired with the standing context for separate encoding. Luna maintains a state containing the goal and current conditions. The complete request Q_t is rendered from that state in a fixed order: context, goal, and remaining condition fields. DeepSeek and GLM also extract R_t with the language model, as specified in Appendix F.3.

A.3 Audit of Requirement and Skill Alignment

Each adapted workstream is checked against its source request and the associated skill bodies. The checks cover whether the work is requested, whether each target's documented capabilities support it, and whether cancelling another workstream leaves the required inputs available. Requirement-level assignments and supporting passages record the evidence for each retained target.

Source overlap. R3 shares 1,829 skill bodies with SkillRouter. Normalized whole-request matching found zero exact duplicates between the selected R3 tasks and the SkillRouter and SkillRet source pools. Shared target skills and identical skill bodies define the task families used for resampling.

B Prompts for Requirement Processing and Skill Retrieval

B.1 Requirement-State Maintenance Prompt

The shared state process receives one user turn at a time and retains the same conversation across the five turns. The panel below summarizes the requirement-state maintenance instructions used in the Luna runs. SkillRet and R3 add an instruction to preserve English requirement content. Appendix F.3 specifies the language-model settings.

Requirement-state maintenance instructions

Maintain the requirements for the current turn only; later turns will arrive separately. Use only the supplied context, the user's message, and the state you maintained earlier in this conversation. Do not inspect skills, answers, or future turns, browse the web, execute the task, or select skills. Treat the supplied context solely as the task setting.

Preserve the goal and every requirement still in force. Failure to repeat a requirement does not cancel it. Preserve negations, numbers, alternatives, and exceptions. Permission does not imply an obligation. A synonymous restatement does not change an existing condition's wording. Apply additions, removals, replacements, restorations, confirmations, or task switches only as explicitly requested. Add no unstated task or constraint.

Keep the context separately and unchanged. Use stable, meaningful condition names and a fixed goal field. Record unresolved information as uncertainty. Return valid JSON without Markdown: the current state, changes with previous and new values and an exact quotation from the current user turn, the semantic edit type, and uncertainties. Exclude withdrawn conditions. Use null for absent previous or new values and an empty change list for a no-op. Return no score, skill identifier, or execution plan.

The response fields are `state`, `changes`, `semantic_edit`, and `uncertainties`. Each change records its field, previous value, new value, edit kind, and evidence quotation. If structural validation fails, one repair request asks for corrected JSON format and types while preserving meaning.

B.2 Judgment Instructions

Judgment has two distinct roles: checking that reference targets are supported by skill documents, and checking submitted task answers against active requirements. The benchmark checks follow Appendix C. For execution, the review system assembles the frozen answer, task-specific criteria, tool evidence, and the original strict verdict into a method-blinded packet. The panel below summarizes the rules applied in criterion-by-criterion review.

Evidence-based answer review instructions

Assess only the requirements still active at the terminal turn. Check that every substantive deliverable is present and that the facts, calculations, code behavior, and media content are correct. When the task requires an action, inspect evidence that the action actually occurred.

Use the original submission and tool records as evidence. An unambiguous, lossless interpretation of an equivalent field name, nesting structure, or file location is allowed. Presentation or serialization errors alone do not make the content incorrect. A strict-check pass does not excuse missing or incorrect substantive work.

Check whether final artifacts and final assertions agree. An explicit final correction can supersede an intermediate value; unresolved contradictory final answers are recorded separately. For every criterion, record a decision, a substantive reason, and supporting evidence. Missing review evidence requires further inspection of the same frozen submission.

B.3 Reranking Prompt

The fixed SkillRouter reranker scores each active requirement against each candidate document. Its semantic prompt is reproduced below, with the query and document replaced by readable placeholders.

The document consists of its name, the first 500 description characters, and the first 2,000 body characters. The model input uses its original chat-role delimiters and an empty thinking block before the answer.

> **System:** Judge whether the Document meets the requirements based on the Query and the Instruct provided. Note that the answer can only be "yes" or "no".
>
> **Instruct:** Given a coding task description, judge whether the skill document is relevant and useful for completing the task
>
> **Query:** [Standing context and one complete active requirement]
>
> **Document:** [Skill name | description | skill body]

We normalize the next-token logits for "yes" and "no" and use the "yes" probability as the pair score. For each candidate, its quality is the maximum over active requirements. All domains share the same reranking prompt and model weights.

C Annotation and Evaluation Guidelines

C.1 Annotation Procedure

Annotators identify workstreams in each source task that can be explicitly added or withdrawn and check their supporting skill passages. A requirement may use several skills, and one skill may support several requirements. Public conversations, reference requirement states, and target assignments are stored separately. Requirement maintenance uses only the public dialogue prefix as its benchmark input.

The annotation checks are organized around four questions:

- **Faithfulness:** Does the adapted requirement preserve the source task's goal, inputs, and technical constraints without adding unrelated work?
- **Document support:** Which passage in the skill body supports the requested capability? A matching name alone is insufficient.
- **Scope consistency:** Do the targets reflect the work still active after the current edit, excluding work that was withdrawn?
- **Input independence:** Can a retained workstream use the standing inputs when another workstream is cancelled? If necessary, the constructed setting explicitly supplies an intermediate input.

The recorded reviews retain the supporting passages and reasons for retaining or revising each target.

C.2 Content-Correctness Evaluation

Each execution is reviewed for active scope, answer completeness, substantive correctness, evidence of required execution, and consistency across final artifacts. Task-specific rules refine these checks. For instance, PDF anonymization checks identifying metadata as well as visible text, and a visual-stability request requiring observation after an API response is assessed over that post-response interval. The same substantive interpretation is applied to all affected methods.

Decision rules. Each criterion decision records a verdict, an explanation, and an evidence reference. A submission is correct when all applicable substantive criteria pass and incorrect when any substantive criterion fails. Unresolved contradictory final answers receive a conflict label and contribute zero to the correct count over 141 requests; paired comparisons use jointly resolved requests. Insufficient evidence triggers further review of the same submitted materials. Wrong answers and timeouts remain in the evaluated set.

C.3 Retrieval and Execution Metrics

Retrieval metric definitions. For a terminal request with target set G, let b_j indicate whether the skill at rank $j \leq 10$ is a target, taking $b_j = 0$ for absent positions, and let $h = \sum_{j=1}^{10} b_j$. Then $P = h/10$, $R = h/|G|$, $F1 = 2PR/(P + R)$ (zero when $P + R = 0$), Hit $= \mathbb{1}[h > 0]$, and FC $= \mathbb{1}[h = |G|]$. Reciprocal rank is $1/\min\{j : b_j = 1\}$, or zero if no target is retrieved. Average precision is $\mathrm{AP@10} = \sum_{j=1}^{10} b_j(\sum_{\ell=1}^{j} b_\ell)/j \,/\, \min(10, |G|)$. Binary-relevance $\mathrm{nDCG@10}$ divides $\sum_{j=1}^{10} b_j/\log_2(j+1)$ by $\sum_{j=1}^{\min(10,|G|)} 1/\log_2(j+1)$. All metrics are averaged over terminal requests; MRR and MAP are the respective means of reciprocal rank and average precision. The displayed F1 is computed for each request before averaging. Short lists receive no new skills, and empty lists score zero on all eight metrics.

Content accuracy counts submissions verified by evidence review to satisfy all active substantive requirements. All-check pass rate averages the original task evaluator's binary all-check verdicts, while mean score averages its scores in $[0, 1]$, including task-specific interface and presentation checks. All-path success rate is the fraction of base tasks with content-correct answers on all three paths. Execution inputs and paired statistical procedures are detailed in Appendix F.

D Examples and Case Studies

The three cases illustrate candidate retrieval and reranking, multiple workstreams sharing one target skill, and target-set changes after requirement restoration. Each case connects public requests and active requirements to skill-document evidence and the resulting ranks.

D.1 Sales Analysis: Recovering Complementary Skills

> **Setting.** A raw transaction table, a cleaned daily-sales table, and a demographic panel for difference-in-differences analysis are all supplied. Each analysis can use its corresponding existing input.
>
> **Turn 1.** First identify unusual increases and decreases in sales trends during the target month.
>
> **Turn 2.** Also clean duplicates and missing values in the raw transaction table again.
>
> **Turn 3.** Also use the supplied panel for a demographic difference-in-differences analysis.
>
> **Turn 4.** Do not perform the demographic analysis just added; keep the other previously requested work.
>
> **Turn 5.** Keep the current scope: the initial task and cleaning duplicates and missing values in the raw transaction table.

The panel summarizes the five user turns, omitting fixed acknowledgments. The terminal request retains anomaly detection and data cleaning. Supplying both raw and cleaned inputs makes these two deliverables independently actionable.

Skill evidence and ranks. The anomaly-detection document describes a Prophet-based comparison of observed series with expected trends, including positive and negative anomaly indices. The data-cleaning document supplies deduplication, missing-value handling, and a configurable tabular-data pipeline. These passages support the two retained deliverables. Full-request retrieval places both targets outside its first 50 candidates; subspace retrieval admits them at ranks 14 and 29. Within that candidate pool, full-request reranking retains anomaly detection at rank 6 and excludes data cleaning from the final ten. SkillProj ranks anomaly detection fifth and data cleaning seventh.

D.2 Logistics Dashboard: Several Requirements, One Target

Setting. A Next.js logistics dashboard renders more than 5,000 shipment rows. The user requests architecture guidance and key code for the active workstreams.

Initial work. Replace the table with a virtualized, sortable, filterable grid using TanStack Table and react-window, with accessible status badges.

First addition. Move duplicated filters into a shared Zustand store synchronized with URL search parameters.

Second addition. Use Server Components and Suspense for the initial load, incremental SWR fetching for pagination, and page metadata.

Withdrawal and confirmation. Cancel the shared-filter refactoring and retain the grid and data-loading work.

The panel condenses the evaluated request. The frontend-discovery skill covers React components, the Next.js App Router, Zustand, SWR, accessibility, and frontend performance. These capabilities support all three workstreams, so restoring the filter requirement leaves the target unchanged (Table 9).

Table 9: Logistics-dashboard case: SkillProj's terminal target ranks.

Path	Target skill	Rank
AB, BA, SWAP	Frontend discovery	10

D.3 Support-Bot Evaluation: Requirement Restoration and Coverage

Setting. A support team has outputs from two prompt versions on about 45 conversations.

Initial work. Compare the versions case by case and identify regressions and hallucination patterns.

First addition. Present the release decision as trade-offs across accuracy, latency, cost per call, and observed safety behavior.

Second addition. Assess whether responses became more or less stable across similar test cases.

Scope changes. The two withdrawal paths end with comparison and stability analysis. The replacement-and-restoration path also retains the release decision.

The prompt-regression skill describes case comparisons, breakage analysis, and stability metrics. The decision-matrix skill describes comparing alternatives across weighted criteria and presenting the trade-offs. The release decision therefore adds the decision-matrix skill to the target set. Table 10 separates target ranks from complete coverage of the active target set.

Table 10: Support-bot case: SkillProj's terminal target ranks and coverage. A dash denotes a skill outside the active target set; > 10 denotes a target outside the returned top ten.

Path	Prompt-regression rank	Decision-matrix rank	Retrieved/targets	Full coverage
AB	1	–	1/1	Yes
BA	1	–	1/1	Yes
SWAP	1	> 10	1/2	No

D.4 TARGET-DISTANCE DIAGNOSTIC

This diagnostic compares whole-request and nearest-requirement geometry using SR complete requests, active requirements, and skill embeddings. Complete requests are rendered from language-model states. For each target skill s, $\Delta = d(q, s) - \min_i d(r_i, s)$, where q encodes the complete request, r_i encodes one active requirement, and d is cosine distance. Positive values indicate a larger distance under whole-request encoding. The minimum selects the nearest requirement. The heatmaps show all 308 labeled target pairs (92 AB, 92 BA, 124 SWAP), one cell per pair. Targets retain a fixed order within each task; all paths use the same 51 tasks. Section 5 reports retrieval-performance comparisons.

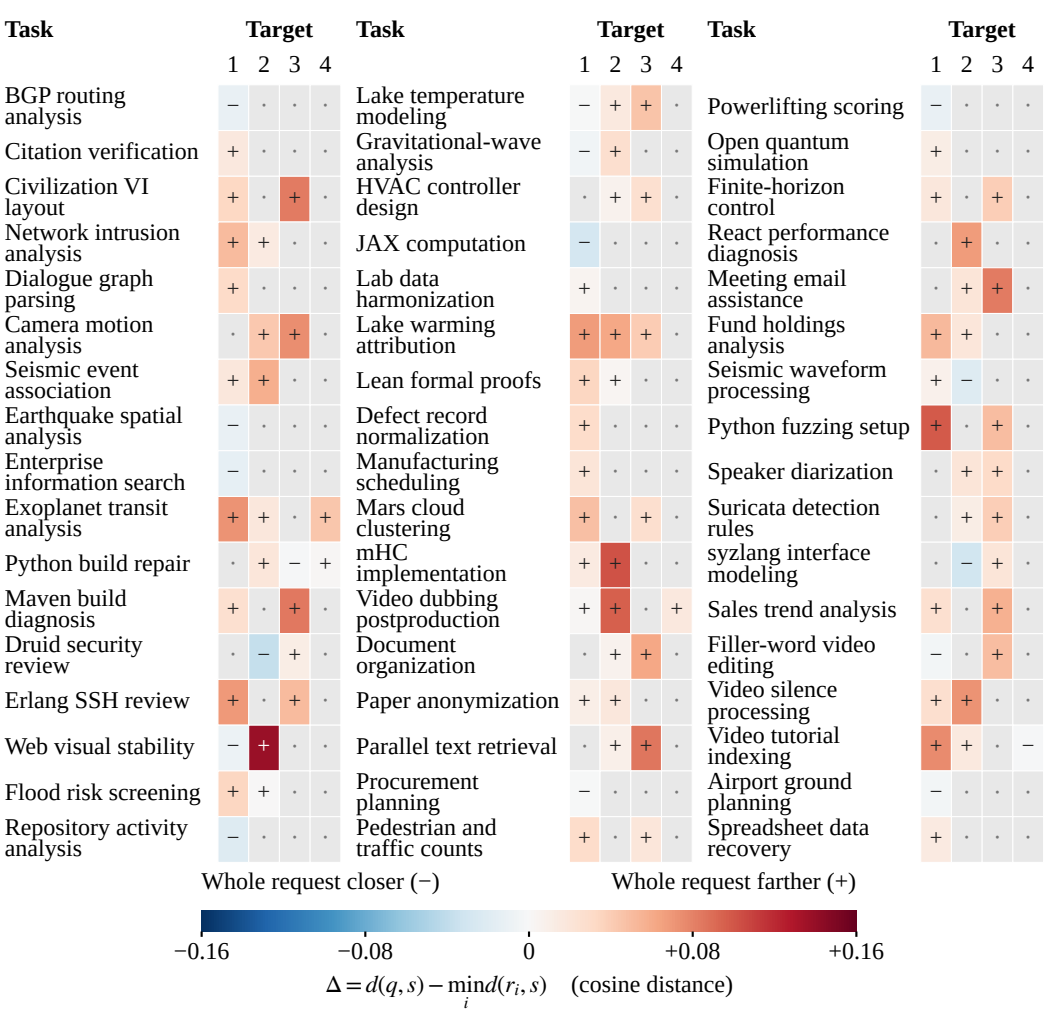


Figure 5: Target-distance shifts for all 51 SR tasks on the AB path (92 target pairs). Each row is a task; columns are reference-skill indices within that task in the same order as Figure 1. Red and blue indicate positive and negative shifts, respectively; the signs repeat this encoding, and gray cells with dots have no target. Task descriptions are abbreviated; Table 11 lists readable task titles.

Table 11: Readable task index for the target-distance diagnostic, in the column order of Figure 1.

Task	Source task title
BGP routing analysis	Azure BGP oscillation and route leak

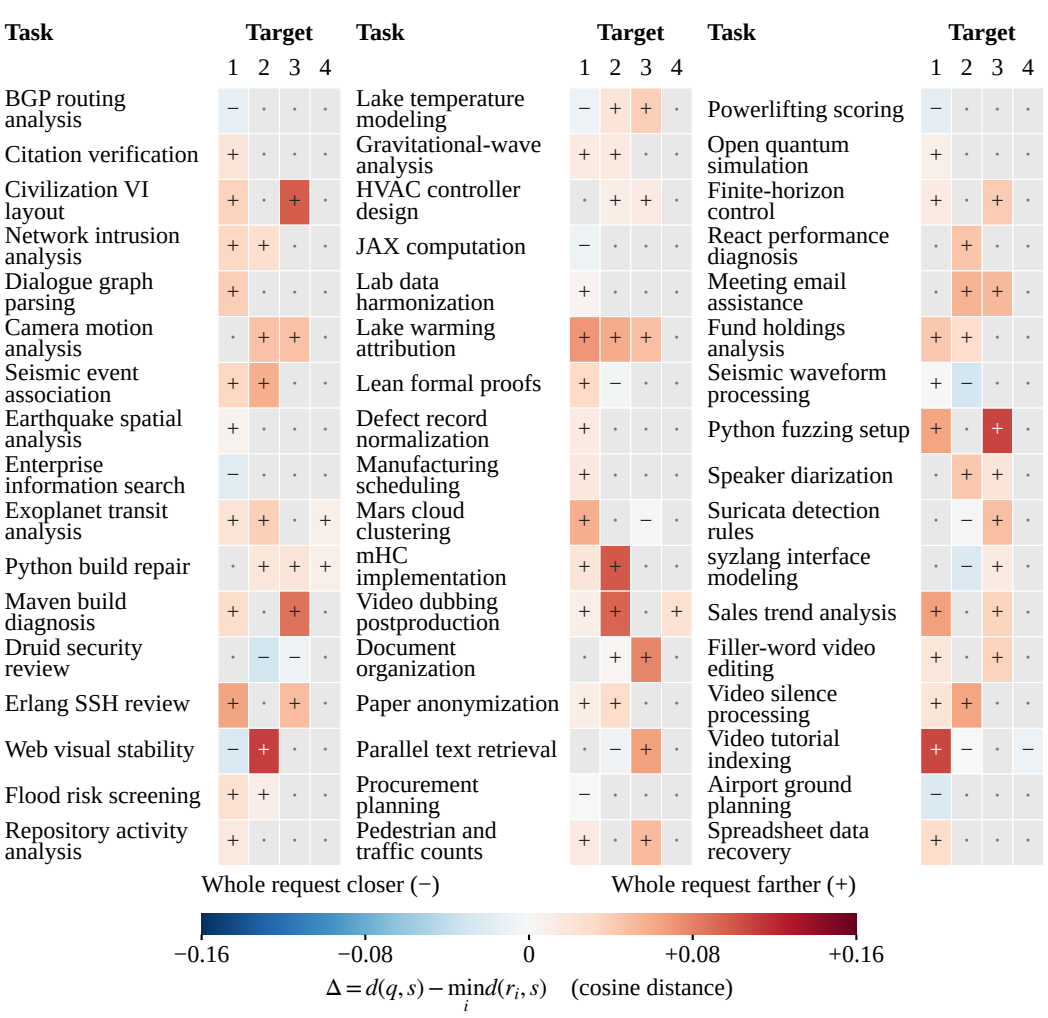


Figure 6: Target-distance shifts for all 51 SR tasks on the BA path (92 target pairs). Each row is a task; columns are reference-skill indices within that task in the same order as Figure 1. Red and blue indicate positive and negative shifts, respectively; the signs repeat this encoding, and gray cells with dots have no target. Task descriptions are abbreviated; Table 11 lists readable task titles.

Table 11 (continued)

Task	Source task title
Citation verification	Citation checking
Civilization VI layout	Civilization VI adjacency optimization
Network intrusion analysis	Network intrusion detection
Dialogue graph parsing	Dialogue parsing
Camera motion analysis	Dynamic-object-aware camera motion
Seismic event association	Earthquake phase association
Earthquake spatial analysis	Earthquake and tectonic plate calculations
Enterprise information search	Enterprise information search
Exoplanet transit analysis	Exoplanet detection and period estimation
Python build repair	AgentOps build repair
Maven build diagnosis	Google Auto build repair
Druid security review	Druid vulnerability repair
Erlang SSH review	Erlang SSH vulnerability repair
Web visual stability	Visual stability repair

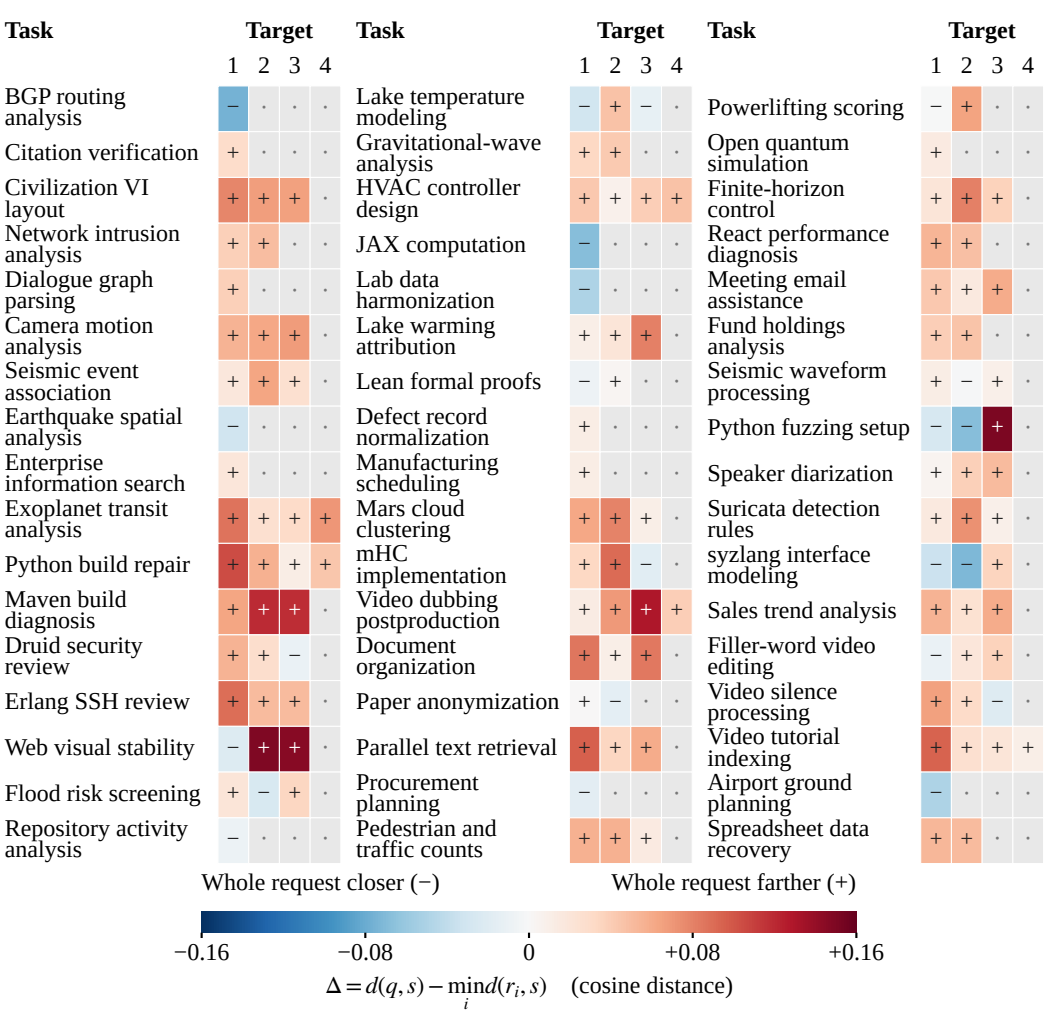


Figure 7: Target-distance shifts for all 51 SR tasks on the SWAP path (124 target pairs). Each row is a task; columns are reference-skill indices within that task in the same order as Figure 1. Red and blue indicate positive and negative shifts, respectively; the signs repeat this encoding, and gray cells with dots have no target. Task descriptions are abbreviated; Table 11 lists readable task titles.

Table 11 (continued)

Task	Source task title
Flood risk screening	Flood risk analysis
Repository activity analysis	GitHub repository analytics
Lake temperature modeling	Lake Mendota modeling
Gravitational-wave analysis	Gravitational-wave detection
HVAC controller design	HVAC control
JAX computation	JAX computing
Lab data harmonization	Laboratory unit harmonization
Lake warming attribution	Lake warming attribution
Lean formal proofs	Lean formal proof
Defect record normalization	Manufacturing codebook normalization
Manufacturing scheduling	Flexible job-shop scheduling
Mars cloud clustering	Mars cloud clustering
mHC implementation	mHC layer implementation
Video dubbing postproduction	Multilingual video dubbing

Table 11 (continued)

Task	Source task title
Document organization	File organization
Paper anonymization	Paper anonymization
Parallel text retrieval	Parallel TF-IDF search
Procurement planning	PDDL procurement planning
Pedestrian and traffic counts	Pedestrian and traffic counting
Powerlifting scoring	Powerlifting coefficient calculation
Open quantum simulation	Quantum numerical simulation
Finite-horizon control	Model predictive control
React performance diagnosis	React performance debugging
Meeting email assistance	Scheduling email assistance
Fund holdings analysis	SEC financial reports
Seismic waveform processing	Seismic phase picking
Python fuzzing setup	Python fuzzing setup
Speaker diarization	Speaker diarization and subtitles
Suricata detection rules	Suricata exfiltration detection
syzlang interface modeling	Syzkaller parallel-port interface description
Sales trend analysis	Trend anomalies and causal inference
Filler-word video editing	Video filler-word removal
Video silence processing	Video silence removal
Video tutorial indexing	Video tutorial indexing
Airport ground planning	Airport ground-traffic planning
Spreadsheet data recovery	Spreadsheet data recovery

E Implementation and Geometric Properties

E.1 Model Inputs and Computation

Model components. We use the SkillRouter encoder and reranker with fixed weights throughout the comparisons. The implementation uses yes/no relevance normalization, per-requirement maximum, and the original-space residual kernel. Candidate ties and zero-gain DPP steps follow a fixed candidate order. Skill bodies are capped at 8,000 characters for embedding and 2,000 for reranking; descriptions use 500 characters, with a 4,096-token model input limit.

Query encoding retains the shared encoder instruction: "Given a coding task description, retrieve the most relevant skill document that would help an agent complete the task." The query text then supplies either the complete request or one contextualized requirement, depending on the comparison. Skill embeddings are shared across these comparisons; the choice of request representation changes the query-side computation.

Comparison settings. Methods use the same public dialogues and task context, full candidate libraries, and frozen SkillRouter encoder, with target annotations used only for evaluation. Each method retains its query construction and ranking stages, including SkillDreamer's pseudo-skills, SkillFlow's generated queries, and SkillProj's requirement subspaces. Returned lists are evaluated with the same top-ten metrics. The SR projection–MaxSim ablation fixes requirement embeddings, candidate budgets, requirement-level reranking, and residual selection, changing only candidate scoring.

Computation. Membership scoring costs $O(ndr)$ for n skills, dimension d, and subspace rank r, after basis construction. Skill embeddings are cached across requests. Reranking scores up to Km pairs for m active requirements.

E.2 Projection and Residual-Kernel Properties

Projection and residual-kernel properties. Writing $c = A^\top v_s$ and $G = A^\top A$ gives $M = c^\top G^+ c$, with the pseudoinverse taken over the retained singular directions. This expression accounts for correlations when combining requirement components. Consider two orthonormal requirement vectors q_1, q_2. A specialized skill $v_s = q_1$ has both membership and MaxSim score one. For

the combination $v_c = (q_1 + q_2)/\sqrt{2}$, membership is one and MaxSim is $1/\sqrt{2}$. MaxSim ranks the specialized direction above the combination; projection gives both directions full membership. This example illustrates how projection preserves combinations alongside individual requirement directions. The shifted cosine matrix S is a Gram matrix of the augmented unit vectors $(1, z_s)/\sqrt{2}$, giving a positive semidefinite kernel after quality weighting. For a selected set Y, the factorization $\det(L_Y) = (\prod_{s \in Y} \bar{a}_s^2) \det(S_Y)$ expresses the joint contribution of individual relevance and residual diversity.

F Additional Experimental Results

F.1 Retrieval Comparisons and Uncertainty

Evaluation inputs. Luna SR uses generated states, complete requests, and matched rankings and token records for the seven baselines; SkillProj uses rankings derived from deterministically maintained requirements under its fixed retrieval configuration. Ret and R3 each use matched rankings and usage records. The SR ablation compares five configurations on the same 153 requests with matched inputs. The Ret reranking comparison fixes candidates, model weights, and DPP parameters.

Prediction processing. Rankings are jointly fixed within each comparison before scoring. Evaluation uses the original returned lists, including short and empty selections, under the metric definitions in Appendix C.

Retrieval statistics. Terminal metrics are averaged across each task's three paths. Shared target skills define the resampled task families. We use 10,000 bootstrap replicates with pointwise 95% intervals. SR, Ret, and R3 have 45, 42, and 45 families, respectively. SkillProj exceeds DSR by 8.88 percentage points on SR (95% interval [4.86, 13.46]) and SkillDreamer by 12.96 on Ret ([5.29, 21.30]). In the matched SR ablations, SkillProj exceeds full-request retrieval by 8.12 points ([3.97, 12.96]), MaxSim retrieval by 3.81 ([0.65, 7.83]), and full-request reranking by 7.63 ([4.04, 11.64]).

Paired estimates. Figure 8 gives point estimates and 95% intervals for candidate retrieval, reranking, residual selection, and the R3 comparisons.

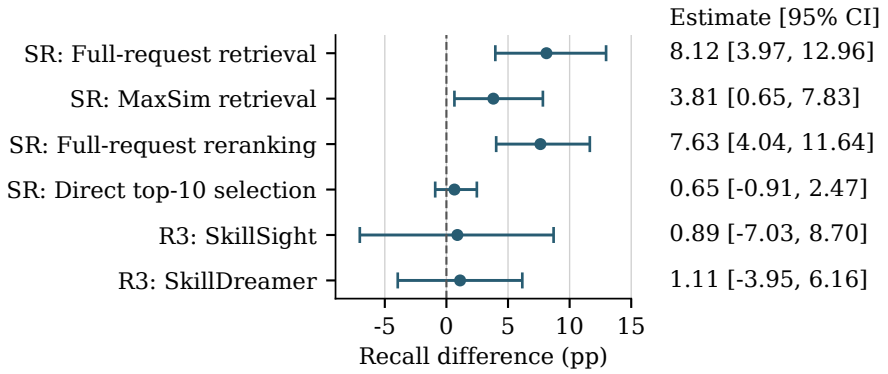


Figure 8: Paired Recall@10 differences: SkillProj minus each named comparison, in percentage points, with 95% family-bootstrap intervals.

F.2 Execution Protocol and Paired Comparisons

Execution protocol. Each method supplies its frozen skill list and full documents for one execution per terminal request on the 47-task SR subset. Retrieval accounting and end-to-end evaluation use separate frozen ranking sets; SkillProj uses requirement-level reranking for execution. All methods share task inputs, a 600-second limit (300 for lake warming), and a 16 KiB tool-output budget; six SkillFlow inputs contain empty lists. The 1,128 executions use the same model and reasoning settings across two batches: 705 for five baselines and 423 for SkillProj, Field-Aware, and SkillFlow. Outputs were fixed before strict scoring and content review under Appendix C.

Execution statistics. We bootstrap 40 shared-skill families 2,000 times, pooling terminal-level correctness differences within each replicate. Table 12 gives pointwise 95% linear-percentile intervals on 130 jointly resolved requests from 45 base tasks. Sensitivity analysis uniformly excludes two BA-path requests with unintended exposure to cross-task materials or queue and method metadata. This 128-request comparison uses the same submissions and statistical procedure.

Table 12: Paired content-accuracy differences: SkillProj minus each baseline, in percentage points (positive favors SkillProj). Brackets give pointwise 95% family-bootstrap intervals. The second comparison excludes two requests with recorded protocol events.

Baseline	All jointly resolved requests ($n = 130$)	Protocol-event exclusions ($n = 128$)
Dense	4.62 [−1.80, 11.49]	4.69 [−1.86, 11.63]
SkillRouter	3.85 [−3.71, 11.28]	3.91 [−3.73, 11.49]
DSR	3.08 [−4.03, 9.56]	3.12 [−4.13, 9.70]
SkillSight	3.85 [−1.65, 9.17]	3.91 [−1.67, 9.30]
SkillDreamer	3.85 [−3.25, 11.11]	3.91 [−3.31, 11.29]
Field-Aware	6.15 [0.00, 12.12]	6.25 [0.00, 12.41]
SkillFlow	7.69 [0.79, 14.41]	7.81 [0.81, 14.73]

Table 13: Execution results by reranking configuration, using separately generated rankings and execution batches. The paired comparison uses 137 jointly resolved requests; accuracy differences are in percentage points with a pointwise 95% family-bootstrap interval.

Measure	Complete-request reranking	Requirement-level reranking
Correct requests	91/141	92/141
Content accuracy (%)	64.54	65.25
All-path success	25/47	24/47
Paired accuracy difference 95% interval	Reference	1.46 [−3.76, 7.09]

F.3 Language Models, Complete Scores, and Generation Usage

State construction. With GPT-5.6 Luna (medium reasoning), public dialogue operations deterministically update active requirements, while Luna generates contextual states and complete requests. DeepSeek V4.1 Flash and GLM-5.3-Flash (high reasoning) independently extract requirements from dialogue prefixes and generate complete requests and method-specific outputs. Each language-model setting shares its states and requests across the eight retrieval methods. Evaluation under DeepSeek and GLM comprises 2,145 states, 429 terminal requests, and 3,432 rankings per model. Turns remain sequential within each dialogue. SkillFlow uses English queries in Luna SR, with at most one language-only correction per query before scoring. All settings share the SkillRouter 0.6B weights, candidate libraries, target annotations, method parameters, and scoring rules.

Complete scores. Tables 14–16 report eight top-ten metrics for each method, benchmark, and language model. The main-text mean assigns each benchmark weight $1/3$. Full coverage requires retrieval of every target skill.

Table 14: SkillRouter-Trajectory: complete results across language models. All metrics are evaluated at ten and multiplied by 100. Bold marks the best value within each language model. FC: full coverage; Hit: hit rate.

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GPT-5.6 Luna (medium)								
Dense	10.46	61.60	17.45	78.43	47.06	53.88	42.62	50.39

Table 14 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GPT-5.6 Luna (medium)								
SkillRouter	12.68	70.21	20.98	86.93	54.25	74.34	58.14	65.29
DSR	12.81	71.73	21.23	88.89	54.90	74.80	58.50	65.97
SkillSight	10.85	61.11	17.97	76.47	45.75	58.38	45.55	52.90
SkillDreamer	12.55	70.21	20.80	84.31	56.21	63.76	51.66	59.65
Field-Aware	10.59	60.13	17.59	75.82	45.75	49.13	38.76	47.00
SkillFlow	11.05	59.75	18.19	75.16	42.48	73.86	58.93	62.72
SkillProj	**15.10**	**80.61**	**24.84**	**93.46**	**65.36**	**76.87**	**62.90**	**71.19**

Table 14 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
DeepSeek V4.1 Flash (high)								
Dense	10.20	60.57	17.03	77.12	45.75	53.53	42.12	49.82
SkillRouter	12.42	68.52	20.51	86.27	52.29	76.19	57.64	64.85
DSR	12.55	69.77	20.74	87.58	53.59	76.55	58.13	65.51
SkillSight	10.72	61.06	17.78	77.12	47.06	57.72	45.46	52.65
SkillDreamer	12.35	69.55	20.50	85.62	54.25	64.02	51.89	59.64
Field-Aware	10.39	59.42	17.28	74.51	45.75	47.95	37.67	45.92
SkillFlow	10.39	54.36	17.04	68.63	39.87	67.86	53.70	57.19
SkillProj	**14.71**	**77.94**	**24.16**	**93.46**	**60.78**	**77.51**	**61.01**	**69.61**

Table 14 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GLM-5.3-Flash (high)								
Dense	10.39	61.27	17.34	76.47	48.37	53.91	43.10	50.59
SkillRouter	12.29	68.95	20.35	86.93	51.63	75.86	56.69	64.37
DSR	12.09	67.65	20.01	86.93	49.67	76.35	57.07	64.44
SkillSight	10.85	61.71	18.01	77.12	47.71	57.88	45.50	52.89
SkillDreamer	12.09	67.54	20.05	83.01	53.59	62.74	50.72	58.25
Field-Aware	10.46	59.64	17.38	73.20	47.71	48.34	38.03	46.26
SkillFlow	8.50	45.81	14.01	58.17	31.37	57.52	45.26	48.37
SkillProj	**15.03**	**78.49**	**24.64**	**94.77**	**59.48**	**80.08**	**61.48**	**70.67**

Table 15: SkillRet-Trajectory: complete results across language models. All metrics are evaluated at ten and multiplied by 100. Bold marks the best value within each language model. FC: full coverage; Hit: hit rate.

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GPT-5.6 Luna (medium)								
Dense	12.62	69.31	20.95	92.06	46.83	81.06	56.57	65.64
SkillRouter	14.68	79.37	24.32	97.62	59.52	88.45	67.86	75.73
DSR	14.68	79.37	24.32	98.41	58.73	88.53	67.82	75.73
SkillSight	14.68	78.97	24.31	96.83	59.52	84.71	64.69	73.23
SkillDreamer	15.32	80.29	25.26	95.24	62.70	86.90	66.62	75.18
Field-Aware	13.81	72.35	22.77	92.86	50.00	85.96	61.44	70.12
SkillFlow	6.35	31.75	10.43	59.52	10.32	57.94	30.82	36.96
SkillProj	**18.02**	**93.25**	**29.66**	**100.00**	**84.13**	**90.77**	**79.40**	**85.77**

Table 15 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
DeepSeek V4.1 Flash (high)								
Dense	11.43	61.90	18.95	89.68	36.51	75.51	48.89	58.35
SkillRouter	13.25	71.83	21.98	**93.65**	49.21	84.12	61.88	69.64
DSR	13.25	71.83	21.98	**93.65**	49.21	84.11	61.87	69.63
SkillSight	13.49	71.03	22.28	92.06	48.41	77.85	56.62	65.41
SkillDreamer	13.97	72.22	23.00	90.48	50.79	80.24	57.61	66.82
Field-Aware	12.70	66.01	20.93	90.48	38.89	75.85	51.85	61.35
SkillFlow	7.86	42.72	13.03	65.08	23.02	63.69	41.83	46.95
SkillProj	**16.51**	**82.41**	**27.08**	92.06	**69.05**	**85.16**	**72.37**	**78.27**

Table 15 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GLM-5.3-Flash (high)								
Dense	11.83	64.55	19.63	93.65	37.30	79.14	52.35	61.76
SkillRouter	13.73	74.47	22.76	**96.03**	52.38	**86.68**	64.42	72.30
DSR	13.65	74.07	22.63	**96.03**	51.59	**86.68**	64.34	72.16
SkillSight	13.57	72.09	22.43	90.48	50.79	80.15	61.05	68.65
SkillDreamer	14.52	75.53	23.93	92.06	54.76	84.35	62.65	71.35
Field-Aware	13.49	70.50	22.24	92.86	45.24	77.47	53.56	63.73
SkillFlow	4.60	24.47	7.62	42.06	9.52	42.06	24.47	28.35
SkillProj	**16.67**	**83.20**	**27.34**	92.06	**70.63**	85.56	**72.20**	**78.51**

Table 16: R3-Skill-Trajectory: complete results across language models. All metrics are evaluated at ten and multiplied by 100. Bold marks the best value within each language model. FC: full coverage; Hit: hit rate.

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GPT-5.6 Luna (medium)								
Dense	12.53	71.11	21.04	93.33	47.33	77.43	57.93	66.11
SkillRouter	13.53	76.67	22.72	94.00	58.67	**85.45**	66.42	73.68
DSR	13.40	76.00	22.50	93.33	58.00	85.38	66.33	73.45
SkillSight	13.80	78.67	23.18	**96.00**	60.67	81.22	63.16	71.62
SkillDreamer	14.00	78.44	23.49	95.33	60.00	82.37	64.29	72.45
Field-Aware	13.33	75.56	22.39	92.00	58.00	74.28	57.56	66.46
SkillFlow	7.67	47.44	13.01	67.33	28.67	65.56	46.56	51.09
SkillProj	**14.33**	**79.56**	**24.00**	92.67	**66.67**	85.03	**68.69**	**75.45**

Table 16 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
DeepSeek V4.1 Flash (high)								
Dense	12.27	70.33	20.62	92.67	47.33	74.13	53.09	62.66
SkillRouter	12.53	71.89	21.07	93.33	50.00	**84.10**	61.68	69.66
DSR	12.60	72.11	21.17	93.33	50.67	**84.10**	61.75	**69.75**
SkillSight	11.47	65.33	19.26	86.67	44.00	69.71	50.50	58.98
SkillDreamer	**13.67**	**76.78**	**22.91**	**95.33**	**57.33**	79.94	60.05	69.31
Field-Aware	12.47	71.44	20.98	90.67	52.00	65.42	49.30	59.17
SkillFlow	8.27	48.89	13.94	70.00	27.33	67.22	47.44	52.43
SkillProj	13.07	74.11	21.92	91.33	56.67	80.56	**61.85**	69.51

Table 16 (continued)

Method	Precision	Recall	F1	Hit	FC	MRR	MAP	nDCG
GLM-5.3-Flash (high)								
Dense	12.53	72.22	21.10	94.00	50.00	76.39	56.00	65.11
SkillRouter	13.07	74.22	21.93	92.00	55.33	83.51	63.61	71.13
DSR	13.20	75.00	22.16	93.33	56.00	83.64	63.69	71.38
SkillSight	12.47	71.33	20.94	89.33	52.67	74.02	55.77	64.25
SkillDreamer	**14.20**	**79.78**	**23.81**	**96.00**	**62.00**	**85.06**	63.82	**73.14**
Field-Aware	13.40	76.33	22.52	95.33	56.00	69.82	52.89	63.27
SkillFlow	3.27	19.89	5.53	31.33	8.67	31.33	19.89	22.47
SkillProj	13.93	79.00	23.40	95.33	**62.00**	82.05	**64.53**	72.53

Generation accounting. Table 17 counts input and output tokens from all completed responses, including cached inputs and reasoning outputs. Each standalone method's usage comprises one shared state-and-request stage plus its method-specific generation. Dense, SkillRouter, DSR, SkillSight, Field-Aware, and SkillProj use the shared stage; SkillDreamer adds skill descriptions, and SkillFlow adds queries and selectors. Total usage across the three benchmarks is divided by 429 terminal trajectories. Savings compare methods using the same language model, tokenization, context handling, and state protocol.

Table 17: Input plus output tokens from completed responses by generation stage. Method-specific columns are increments added to the shared stage. SR, Ret, and R3 contain 153, 126, and 150 terminal trajectories.

Language model	Benchmark	Shared	SkillDreamer	SkillFlow
GPT-5.6 Luna	SR	11,883,552	1,065,565	2,358,336
GPT-5.6 Luna	Ret	11,994,236	1,328,888	2,087,998
GPT-5.6 Luna	R3	11,882,314	1,072,053	2,375,512
DeepSeek V4.1 Flash	SR	1,368,303	786,770	1,439,454
DeepSeek V4.1 Flash	Ret	2,428,644	1,079,631	1,248,989
DeepSeek V4.1 Flash	R3	2,019,395	1,053,616	1,522,044
GLM-5.3-Flash	SR	1,195,579	646,103	1,315,489
GLM-5.3-Flash	Ret	1,984,526	1,067,844	1,396,941
GLM-5.3-Flash	R3	1,551,734	767,347	1,477,991

Local computation. SkillProj encodes active requirements, scores the skill library by projection, reranks up to $50m$ pairs for m requirements, and performs ten greedy residual-selection steps. The main experiment host has two NVIDIA A100-SXM4-80GB GPUs, 24 allocated AMD EPYC 7453 cores, and 240 GB RAM. Encoder and reranker revisions are fixed across comparisons.

Paired comparisons. Within each language-model setting, we estimate task-family sampling variation using 10,000 paired bootstrap replicates over 45 SR, 42 Ret, and 45 R3 families, reporting pointwise 95% intervals. Tables 18–20 report SkillProj's differences from all seven baselines under DeepSeek and GLM, alongside the Luna comparisons. Figure 4 shows the corresponding full-coverage point estimates.

Table 18: SkillRouter-Trajectory: SkillProj minus each baseline, in percentage points, with pointwise 95% task-family bootstrap intervals.

Baseline	Recall@10 difference	FullCoverage@10 difference
GPT-5.6 Luna (medium)		
Dense	+19.01 [11.94, 26.67]	+18.30 [9.72, 28.15]
SkillRouter	+10.40 [5.29, 16.28]	+11.11 [4.24, 18.95]
DSR	+8.88 [4.86, 13.46]	+10.46 [4.02, 17.95]
SkillSight	+19.50 [11.70, 28.26]	+19.61 [9.80, 31.16]

Table 18 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
GPT-5.6 Luna (medium)		
SkillDreamer	+10.40 [5.39, 16.28]	+9.15 [2.42, 17.33]
Field-Aware	+20.48 [11.69, 30.44]	+19.61 [8.93, 31.25]
SkillFlow	+20.86 [12.04, 30.68]	+22.88 [11.56, 35.37]

Table 18 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
DeepSeek V4.1 Flash (high)		
Dense	+17.37 [8.84, 25.75]	+15.03 [5.08, 25.53]
SkillRouter	+9.42 [2.66, 16.06]	+8.50 [0.00, 17.36]
DSR	+8.17 [1.81, 14.12]	+7.19 [-2.00, 16.67]
SkillSight	+16.88 [9.93, 24.57]	+13.73 [5.03, 23.91]
SkillDreamer	+8.39 [1.93, 14.37]	+6.54 [-1.72, 14.97]
Field-Aware	+18.52 [9.30, 27.88]	+15.03 [4.17, 26.42]
SkillFlow	+23.58 [13.21, 34.56]	+20.92 [8.81, 34.55]

Table 18 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
GLM-5.3-Flash (high)		
Dense	+17.21 [8.67, 25.51]	+11.11 [1.85, 21.01]
SkillRouter	+9.53 [3.05, 15.61]	+7.84 [0.00, 16.39]
DSR	+10.84 [4.83, 16.73]	+9.80 [1.89, 19.05]
SkillSight	+16.78 [9.59, 24.43]	+11.76 [3.57, 21.48]
SkillDreamer	+10.95 [4.08, 17.78]	+5.88 [-2.67, 14.89]
Field-Aware	+18.85 [10.01, 28.09]	+11.76 [1.39, 22.67]
SkillFlow	+32.68 [22.27, 43.50]	+28.10 [16.02, 41.50]

Table 19: SkillRet-Trajectory: SkillProj minus each baseline, in percentage points, with pointwise 95% task-family bootstrap intervals.

Baseline	Recall@10 difference	FullCoverage@10 difference
GPT-5.6 Luna (medium)		
SkillSight	+14.29 [7.54, 20.90]	+24.60 [11.90, 37.30]
SkillDreamer	+12.96 [5.29, 21.30]	+21.43 [8.73, 34.92]

Table 19 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
DeepSeek V4.1 Flash (high)		
Dense	+20.50 [12.30, 28.31]	+32.54 [19.84, 45.24]
SkillRouter	+10.58 [1.59, 18.92]	+19.84 [6.35, 33.33]
DSR	+10.58 [1.59, 18.92]	+19.84 [6.35, 33.33]
SkillSight	+11.38 [1.32, 21.16]	+20.63 [6.35, 34.92]
SkillDreamer	+10.19 [3.31, 16.67]	+18.25 [6.35, 30.16]
Field-Aware	+16.40 [9.13, 23.41]	+30.16 [17.46, 43.65]
SkillFlow	+39.68 [29.23, 50.00]	+46.03 [32.54, 59.52]

Table 19 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
GLM-5.3-Flash (high)		
Dense	+18.65 [8.73, 27.25]	+33.33 [19.05, 46.83]
SkillRouter	+8.73 [-1.59, 18.12]	+18.25 [3.17, 32.54]
DSR	+9.13 [-1.19, 18.52]	+19.05 [3.97, 33.33]
SkillSight	+11.11 [-0.66, 22.09]	+19.84 [4.76, 34.13]
SkillDreamer	+7.67 [-1.59, 16.14]	+15.87 [2.38, 29.37]
Field-Aware	+12.70 [4.23, 20.24]	+25.40 [11.90, 38.10]
SkillFlow	+58.73 [46.16, 69.71]	+61.11 [47.62, 73.81]

Table 20: R3-Skill-Trajectory: SkillProj minus each baseline, in percentage points, with pointwise 95% task-family bootstrap intervals.

Baseline	Recall@10 difference	FullCoverage@10 difference
GPT-5.6 Luna (medium)		
Dense	+8.44 [1.33, 15.74]	+19.33 [8.16, 30.77]
SkillRouter	+2.89 [-3.12, 8.68]	+8.00 [-2.08, 18.67]
DSR	+3.56 [-2.44, 9.33]	+8.67 [-1.76, 19.44]
SkillSight	+0.89 [-7.03, 8.70]	+6.00 [-6.92, 19.26]
SkillDreamer	+1.11 [-3.95, 6.16]	+6.67 [-2.04, 15.97]
Field-Aware	+4.00 [-1.13, 9.48]	+8.67 [-1.45, 19.33]
SkillFlow	+32.11 [22.80, 41.78]	+38.00 [25.16, 51.57]

Table 20 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
DeepSeek V4.1 Flash (high)		
Dense	+3.78 [-4.33, 12.14]	+9.33 [-2.47, 20.92]
SkillRouter	+2.22 [-4.70, 8.82]	+6.67 [-2.78, 16.31]
DSR	+2.00 [-4.86, 8.56]	+6.00 [-3.40, 15.65]
SkillSight	+8.78 [0.31, 17.54]	+12.67 [1.19, 25.36]
SkillDreamer	-2.67 [-8.98, 4.31]	-0.67 [-11.33, 10.67]
Field-Aware	+2.67 [-4.42, 9.98]	+4.67 [-6.94, 16.31]
SkillFlow	+25.22 [16.06, 35.07]	+29.33 [17.33, 42.22]

Table 20 (continued)

Baseline	Recall@10 difference	FullCoverage@10 difference
GLM-5.3-Flash (high)		
Dense	+6.78 [1.11, 12.37]	+12.00 [3.33, 21.01]
SkillRouter	+4.78 [1.13, 8.64]	+6.67 [0.00, 13.33]
DSR	+4.00 [0.45, 7.67]	+6.00 [-0.68, 12.82]
SkillSight	+7.67 [1.02, 15.53]	+9.33 [-0.67, 20.57]
SkillDreamer	-0.78 [-5.10, 3.74]	+0.00 [-7.88, 8.16]
Field-Aware	+2.67 [-3.22, 8.44]	+6.00 [-4.17, 16.31]
SkillFlow	+59.11 [49.54, 68.41]	+53.33 [42.77, 64.49]