SKILL IS NOT DOCUMENT: QUERY-CONDITIONED COMPATIBILITY FOR LLM AGENT SKILL ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model agents increasingly rely on reusable skills, making skill retrieval a critical front-end component of agent systems. Skill retrieval, however, is not ordinary document retrieval: a useful top-K result must contain individually relevant skills that also form an executable set for the current query. Existing benchmarks and training pipelines largely supervise pairwise relevance and discard the rejection decisions produced when a language model judges a sampled skill combination to be implausible. We introduce R3-Skill, a Chinese–English benchmark that retains these rejections as query-conditioned compatibility supervision. R3-Skill contains 10,246 deduplicated skills, 41,592 accepted queries, and 32,828 rejected annotations across four language directions; all multi-skill test labels were independently reviewed by multiple experts, and 15,962 parseable rejections are organized into an eight-class taxonomy. We further propose a two-stage system composed of R3-Embedding, a multi-positive bi-encoder for large-pool recall, and R3-Reranker, a cross-encoder trained with graded ListNet supervision. Our analysis shows that this signal is stage-dependent, helping cross-encoder reranking while providing no benefit for the tested bi-encoder objective. On R3-Skill, the complete pipeline achieves 75.39% Hit@1, 81.97% NDCG@10, and 33.27% Set-Compat, a 36.6% relative gain over the strongest reranking baseline. It also obtains 83.87% NDCG@10 on SkillRet, demonstrating transfer beyond R3-Skill.

1 INTRODUCTION

Reusable skills package instructions, constraints, scripts, and tool-use procedures into capabilities that an agent can retrieve and execute. This abstraction extends earlier tool-use systems such as ReAct (Yao et al., 2023) and Toolformer (Schick et al., 2023): whereas a tool exposes an atomic call, a skill describes how to solve a class of tasks, often through a multi-step procedure. The Agent Skills specification (Anthropic, 2025a) and recent skill ecosystems (Anthropic, 2025b; Xu & Yan, 2026; Jiang et al., 2026) have made this abstraction increasingly common. Skills can also function as updatable agent memory; for example, Memento-Skills reports substantial relative gains on GAIA and HLE (Zhou et al., 2026).

Large skill libraries make an explicit routing layer necessary. Loading every skill into the prompt is expensive and eventually impossible: public collections already contain tens or hundreds of thousands of entries (Li et al., 2025; Su et al., 2026; Li et al., 2026a), while a single skill may include a long body of API constraints, dependencies, examples, and canonical code. In R3-Skill, the complete library contains 21.2M tokens, and the longest skill contains 31,571 tokens (Figure 2). Moreover, metadata alone is insufficient when many skills have similar names or descriptions; SkillRouter reports absolute drops of 31–44 points when the body is removed from routing inputs (Zheng et al., 2026). A practical agent must therefore retrieve a small set of skills from full, heterogeneous skill documents.

The central difficulty is that retrieved skills are acted upon rather than merely read. In document retrieval, returning two relevant passages together is usually harmless. In skill routing, two individually plausible skills may be redundant, contradictory, or impossible to compose for the current request. Figure 1 illustrates this query dependence: the same pair may be jointly necessary for one task but unnecessary for another. Multi-skill benchmarks also show that adding skills does not

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

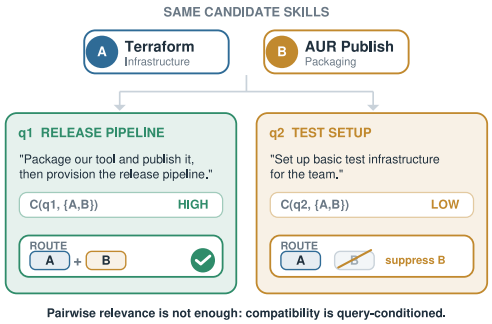


Figure 1: Query-conditioned skill compatibility. Given the same candidate skills, q_1 requires the pair $\{A, B\}$, whereas q_2 requires only A, making B redundant. Pairwise relevance alone cannot represent this set-level decision.

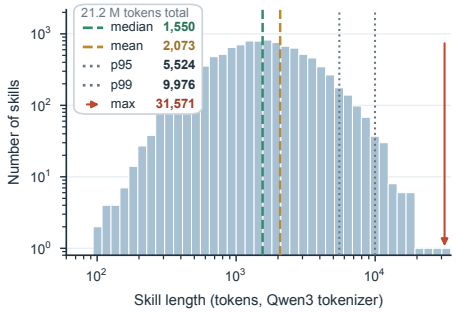


Figure 2: Skill-length distribution in R3-Skill (name, description, and body; Qwen3 tokenizer).

uniformly improve task completion (Li et al., 2026b; Liu et al., 2026; Zhang et al., 2026). Thus, successful routing requires both *relevance* between each query–skill pair and *compatibility* of the retrieved set under the query.

Existing skill-retrieval resources emphasize relevance supervision. End-to-end benchmarks provide limited retriever training data, whereas retrieval benchmarks such as SkillRet (Cho et al., 2026) and SkillRouter (Zheng et al., 2026) inherit the standard query–candidate formulation. Meanwhile, language-model-based data synthesis already computes a useful additional signal: before writing a query for a sampled skill set, the generator often rejects combinations that do not form a natural task. These rejection decisions are normally discarded. We instead treat them as supervision about what should *not* be retrieved together under a query. We introduce **R3-Skill** and the **Reject-as-Resource Retriever (R3)**. R3-Skill retains the WRITE/SKIP judgments, covers four Chinese–English directions, separates train and test skills, and provides expert-verified test labels. R3 uses a bi-encoder for large-pool recall and a cross-encoder for compatibility-aware reranking. The design deliberately assigns rejection supervision to the stage that can model query–candidate interaction.

Our contributions are fourfold:

- We formulate skill routing as relevance plus query-conditioned set compatibility, clarifying why independently relevant skills need not form a valid routed set.
- We construct R3-Skill with 10,246 skills, 41,592 accepted queries, 32,828 rejection annotations, four language directions, disjoint skill pools, and an eight-class rejection taxonomy.
- We develop R3-Embedding and R3-Reranker, showing that rejection supervision benefits graded listwise reranking but not the tested bi-encoder objective.
- The complete system reaches 75.39% Hit@1, 81.97% NDCG@10, and 33.27% SetCompat on R3-Skill, and transfers to the SkillRet benchmark.

2 Related Work

Agent research has progressed from atomic tool calls toward reusable procedural capabilities (Yao et al., 2023; Schick et al., 2023; Xu & Yan, 2026; Jiang et al., 2026). Recent systems organize, acquire, internalize, or evolve skill libraries (Zhou et al., 2026; Li et al., 2025; 2026a; Lu et al., 2026; Zhang et al., 2026). Complementary benchmarks evaluate whether skills improve downstream task completion (Su et al., 2026; Li et al., 2026b; Liu et al., 2026). These studies establish the utility and variability of skills, but end-to-end failures conflate retrieval, composition, interpretation, and execution. R3-Skill isolates the upstream routing problem while retaining supervision about multi-skill composition.

Classical sparse and dense retrieval methods, including BM25 (Robertson & Zaragoza, 2009), DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2020), and RocketQA (Qu et al., 2021), rank

candidates primarily through query–document relevance. SkillRouter (Zheng et al., 2026) adapts a retrieve-and-rerank pipeline to large full-text skill collections and demonstrates the importance of body content. SkillRet (Cho et al., 2026) provides large disjoint training and evaluation pools and shows strong gains from domain-specific fine-tuning. Unlike prior skill-retrieval resources such as SkillRet and SkillRouter, R3-Skill retains rejected multi-skill combinations and lets us study where their compatibility signal should enter a two-stage retriever.

Multilingual embedding models improve cross-language semantic matching (Chen et al., 2024), but multilinguality alone does not provide supervision about whether several capabilities should be used together. This distinction is important because public skill ecosystems are strongly language-imbalanced (Hu et al., 2026). Related work also identifies structural alignment as a source of retrieval-augmented agent failures (Wei et al., 2026). R3-Skill combines bilingual routing with explicit WRITE/SKIP judgments, enabling separate analysis of semantic relevance and query-conditioned compatibility.

3 Problem Formulation

Let $\mathcal{S}$ be a skill library and let $S_q^* \subseteq \mathcal{S}$ denote the set of skills required by query q. A router returns an ordered list π_q and succeeds only when the required skills appear near the top. Pairwise relevance is necessary, but it does not fully describe a multi-skill result: the selected skills must also form a coherent task-specific set. We use the following decomposition as a modeling view:

$$\Pr[\, S_q^* \subseteq \text{top-}K \mid q \,] \approx \underbrace{\prod_{s \in S_q^*} \Pr[\, s \in \text{top-}K \mid q \,]}_{\text{pairwise relevance}} \cdot \underbrace{C(q, S_q^*)}_{\text{set compatibility}},$$

where $C(q, S_q^*) \in \mathbb{R}_{\geq 0}$ is a query-conditioned correction; $C \equiv 1$ recovers an independent-relevance view. R3-Skill operationalizes this missing factor through two generator outcomes. WRITE indicates that a sampled skill set supports a natural query, whereas SKIP records that the set should not be jointly selected for that query context. This formulation motivates two requirements. First, evaluation should measure complete set recovery, not only per-item recall. Second, the model component consuming SKIP supervision must represent query-dependent interactions. A bi-encoder assigns each skill a shared vector across all queries, while a cross-encoder can condition each candidate score on the full query. We test this architectural distinction directly.

4 R3-Skill Benchmark

Task and design goals. R3-Skill isolates skill routing as a standalone retrieval problem. Each evaluation item consists of a user query q, a held-out skill library, and a ground-truth set S_q^* containing one to three skills. The router must rank the required capabilities above thousands of alternatives without observing downstream execution. This separation is deliberate: end-to-end task failure can arise from retrieval, composition, interpretation, or execution, whereas R3-Skill measures the upstream decision directly. The benchmark is built around four goals. First, it tests *full-text routing*: candidates contain a name, description, and long procedural body rather than a short tool signature. Second, it tests *multi-skill sufficiency*: 15,791 accepted queries require two or three skills, so success depends on recovering a coherent set. Third, it tests *cross-lingual routing* across en2en, en2zh, zh2en, and zh2zh. Finally, it preserves *hard rejected combinations*. These SKIPs are sampled from nearby skills and therefore distinguish query-conditioned incompatibility from trivial topical mismatch; Table 1 shows representative WRITE and SKIP cases.

Scope and positioning. R3-Skill targets large registries in which many candidates overlap semantically and a request may require a small skill set. It is not intended to estimate the natural prevalence of incompatible combinations in live traffic: the sampling procedure deliberately concentrates on difficult neighborhoods. Relative to SkillRouter (Zheng et al., 2026) and SkillRet (Cho et al., 2026), R3-Skill has a smaller skill pool than the former and fewer training queries than the latter, but adds four bilingual directions, 32,828 retained rejection annotations, and an eight-class rejection taxonomy. These resources make it possible to study not only whether a retriever finds relevant skills, but also where compatibility supervision should enter the model.

Skill combination	Decision	Why
Terraform IaC + AUR Publish	WRITE	Natural release and infrastructure pipeline
Code Review + Systematic Code Review	SKIP	Redundant capabilities; one skill is sufficient
Rust Reliability + Go Security Patterns	SKIP	Inconsistent language ecosystems for one request

Table 1: Illustrative R3-Skill decisions. SKIP sets remain topically plausible but are rejected because their skills are redundant or cannot form one natural request.

Stage	#skill
Raw collection (English, GitHub)	94,966
Raw collection (Chinese, GitHub)	246
Total raw	95,212
After cross-source dedup + file-op removal	10,246
train / test pool split (80 / 20)	8,196 / 2,050

Table 2: Skill pool construction. Cross-source forks and mirrors are removed by two-layer (filename + content) hash deduplication; pure file-operation entries are filtered per source.

These decisions are query-conditioned: the same capability can be jointly necessary in one request yet redundant or unsuitable in another.

4.1 Skill Collection and Deduplication

We collect 95,212 raw skills from public GitHub repositories: 94,966 English entries and 246 Chinese entries. The raw collection contains extensive forks, mirrors, and near-identical file-operation skills. We apply cross-source filename and content hashing, remove pure file-operation entries, and retain 10,246 skills. The resulting documents preserve each skill's name, description, and body. Table 2 summarizes the filtering and pool split.

4.2 Compatibility-Oriented Sampling

Uniformly sampled combinations produce mostly trivial rejections because unrelated skills are easy to identify. We instead sample semantically close combinations. BGE-M3 encodes the skill documents, after which we compare KMeans configurations with $K \in \{10, 20, 30, 40\}$ using clustering quality and human inspection. We select 40 sub-clusters and merge them into eight thematic super-domains. Single-skill cases are sampled freely; two-skill cases draw both skills from the same sub-cluster; and three-skill cases draw all skills from the same super-domain. These constraints produce nontrivial decisions. The generator accepts 56.3% of two-skill combinations and 36.5% of three-skill combinations, even though all sampled skills are semantically nearby under the hierarchy. Thus, semantic proximity does not imply that a combination forms a natural task.

4.3 WRITE/SKIP Generation

Figure 3 presents the complete construction process. Each sampled set is first judged jointly. A WRITE decision produces a user query; a SKIP decision preserves the rejected set and its rationale. For $k = 1$, invalid generations are retained only for accounting purposes. Compatibility supervision uses the $k \geq 2$ rejections, for which the model explicitly judges joint plausibility before query generation. The WRITE branch uses six request styles: *task_direct*, *scene*, *role_setup*, *constraint*, *multi_step*, and *debug*. Generation covers four directions: en2en, en2zh, zh2en, and zh2zh. Table 3 reports the accepted-query distribution by direction and number of required skills.

DeepSeek-V4-Pro (DeepSeek-AI, 2026) generates training queries and SKIP annotations. Qwen3-235B-A22B (Qwen Team, 2025) drafts the test queries and labels. We deliberately use different model families for training and evaluation to reduce the risk that a retriever benefits from a single generator's recurring phrasing. The full prompts specify the requested language, one of six user styles, a 50–300 word or character range, and explicit prohibitions against skill-name disclosure and sentence copying.

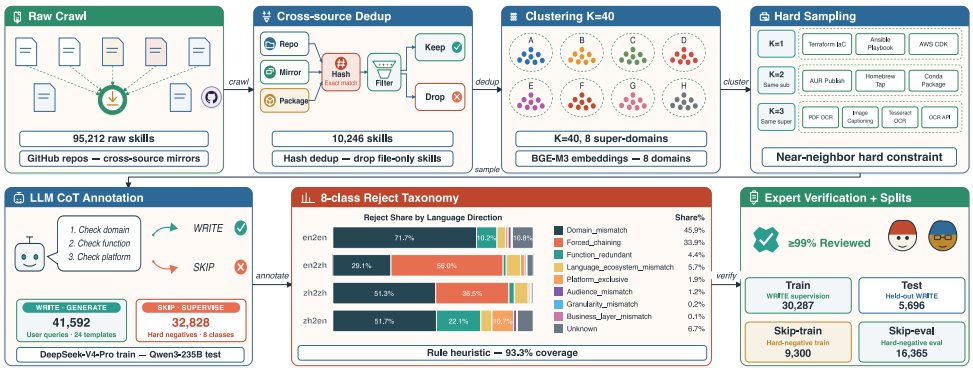


Figure 3: Detailed view of the R3-Skill data production pipeline: raw 95,212 → cross-source dedup 10,246 → clustering → near-neighbor hard-constraint sampling → LLM CoT annotation (WRITE / SKIP) → multi-expert verification → benchmark splits and the 8-class Reject Taxonomy.

Direction	#query	k=1	k=2	k=3
en2en	21,332	12,227	6,535	2,570
en2zh	17,957	12,326	4,674	957
zh2en	1,148	641	353	154
zh2zh	1,155	607	378	170
Total	41,592	25,801	11,940	3,851

Table 3: Query distribution by direction & by k.

4.4 Rejection Taxonomy and Quality Control

We derive an eight-class taxonomy from 15,962 SKIP records with parseable rationales. The rules match specific causes first and use *forced_chaining* as a final fallback for rationales describing an artificial combination of otherwise plausible skills. Rule coverage is 93.3%; the remaining 6.7% is labeled *unknown*. Taxonomy proportions are computed only on this parseable subset and exclude single-skill generation failures.

The taxonomy exposes language-dependent rejection patterns. en2en rejections are dominated by domain mismatch, whereas 56.0% of en2zh rejections are assigned to forced chaining. For en2zh with $k = 3$, the SKIP rate reaches 80.9%. Under the strictest three-skill sampling condition, 63.5% of combinations are still rejected, showing that nearest-neighbor sampling remains far from sufficient for compatibility.

4.5 Splits and Benchmark Scope

We split at the skill level: 8,196 skills form the training pool and 2,050 form the test pool, with no overlap. The final training split contains 30,287 accepted queries and the test split contains 5,696. We discard 5,609 queries whose required skills cross the two pools, preventing a model from memorizing test capabilities during training.

For quality assurance, all multi-skill test labels are independently reviewed by nine experts. The initial disagreement rate was below 5%; disputed cases underwent expert re-review followed by a second round of independent scoring. In addition, 25% of the test queries are sampled for naturalness review. Automated checks find no duplicate final queries. A conservative skill-name substring audit yields leakage rates of 7.8% for en2en, 2.3% for en2zh, 2.1% for zh2zh, and 0.0% for zh2en; these matches include legitimate technology terms and are reported as an audit rather than treated as proof of semantic leakage. We also spot-check taxonomy assignments and obtain at least 99% correctness on the inspected rule hits. Appendices D, F, and G report the complete checklist, taxonomy distributions, and prompt templates, respectively.

Together, the disjoint pools, cross-generator construction, expert review, and explicit leakage audit mitigate several risks: memorization, generator-style overfitting, label error, and lexical shortcuts. They do not make synthetic requests equivalent to live traffic, but they provide a controlled benchmark for studying large-pool, multilingual, and compatibility-aware routing.

5 METHOD

R3 follows a retrieve-and-rerank architecture. **R3-Embedding**, initialized from Qwen3-Embedding-0.6B (Zhang et al., 2025), retrieves 20 candidates from the full skill pool. **R3-Reranker**, initialized from Qwen3-Reranker-0.6B (Zhang et al., 2025), then reranks these candidates. The stages divide the problem by capacity: the bi-encoder provides efficient global recall, while the cross-encoder resolves fine-grained, query-dependent competition.

5.1 R3-EMBEDDING: MULTI-POSITIVE RECALL

For query q with ground-truth set S_q^*, we sample one main positive $s^+ \in S_q^*$ and mine one offline hard negative s^- using Qwen3-Embedding-8B (rank in $[20, 50)$ and cosine similarity below 0.85). Positives from other queries in the batch act as additional negatives. Let $z_c = \mathbf{e}(q)^\top \mathbf{e}(c)/\tau$, $\mathcal{C}_q = \{s^+, s^-\} \cup$ in-batch, and $S_q^\diamond = S_q^* \setminus \{s^+\}$. To avoid treating sibling ground-truth skills as negatives, we define $\widetilde{\mathcal{C}}_q = \mathcal{C}_q \setminus S_q^\diamond$ and $\mathcal{N}_q = \mathcal{C}_q \setminus (\{s^+\} \cup S_q^\diamond)$. The objective is

$$\mathcal{L}_{\text{emb}}(q) = \mathcal{L}_{\text{nce}}(q) + \lambda \mathcal{L}_{\text{sib}}(q),$$

$$\mathcal{L}_{\text{nce}}(q) = -\log \frac{\exp(z_{s^+})}{\sum_{c \in \widetilde{\mathcal{C}}_q} \exp(z_c)},$$

$$\mathcal{L}_{\text{sib}}(q) = -\frac{1}{|S_q^\diamond|} \sum_{s^\diamond \in S_q^\diamond} \log \frac{\exp(z_{s^\diamond})}{\exp(z_{s^\diamond}) + \sum_{c \in \mathcal{N}_q} \exp(z_c)}.$$

We use $\tau = 1/30$ and $\lambda = 0.25$. For single-skill queries, $S_q^\diamond = \emptyset$ and the objective reduces to standard InfoNCE (van den Oord et al., 2018). For multi-skill queries, the sibling term pulls the query toward every required skill without treating sibling positives as in-batch negatives.

5.2 R3-RERANKER: REJECTION-AWARE LISTWISE RANKING

For each WRITE query, SKIP partners are drawn from the same clustered neighborhood as the accepted skill set, making them topically close alternatives to the required capabilities. This construction tests whether the reranker can distinguish joint necessity from topical proximity. The reranker training list contains the ground-truth skills, these SKIP partners, and top-ranked candidates from R3-Embedding, padded to 10 candidates. We assign graded labels and optimize ListNet:

$$\text{label}(s) = \begin{cases} 3, & s \in S_q^*, \\ 1, & s \text{ is a SKIP partner}, \\ 0, & \text{otherwise}, \end{cases}$$

$$\mathcal{L}_{\text{rrk}}(q) = -\sum_s y_s \log p_s.$$

where y_s is the softmax-normalized target label and p_s is the softmax of the reranker scores. SKIP partners are never treated as evaluation ground truth; the intermediate label only shapes their relative position within the training list. This placement is intentional. A bi-encoder must map each skill to one query-independent vector, so a rejection that is valid for one query can conflict with positive evidence from another. A cross-encoder instead scores each (q, s) pair jointly and can use a rejected candidate as query-specific evidence for ranking nearby capabilities below the required skills.

6 EXPERIMENTS

6.1 EVALUATION SETUP

We ask whether task-specific training improves large-pool recall, whether rejection-aware reranking improves complete-set recovery, and whether the learned models transfer across language directions and benchmarks. We evaluate on R3-Skill and SkillRet (Cho et al., 2026). Baselines include

Model	H@1	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15	SC
Embedding											
BM25	6.04	8.00	8.82	9.25	9.93	12.41	14.01	9.32	11.69	13.18	0.40
BGE-M3 (base)	39.59	46.94	49.20	50.27	53.88	60.72	64.67	51.62	58.55	62.57	6.93
Qwen3-Embedding-0.6B (base)	43.12	51.58	53.72	54.74	59.44	65.82	69.56	57.18	63.54	67.40	4.16
Qwen3-Embedding-8B (base)	62.29	71.05	73.01	73.74	79.13	84.98	87.65	76.56	82.79	85.71	10.69
SkillRouter-Embedding-0.6B	22.09	29.51	32.14	33.54	36.45	44.48	49.67	34.94	42.78	47.82	2.18
SkillRet-Embedding-0.6B	46.72	55.86	58.04	59.10	64.47	71.01	74.92	62.31	69.10	73.03	11.88
R3-Embedding	72.07	80.06	81.60	82.05	**87.22**	91.77	93.38	**85.31**	90.40	92.22	28.12
+ Rerank (upstream = R3-Embedding, top-20)											
+ Qwen3-Reranker-0.6B	68.08	77.61	79.54	80.11	86.45	**92.12**	**94.12**	84.46	**90.70**	**93.05**	17.82
+ SkillRouter-Reranker-0.6B	67.98	75.57	77.65	78.63	82.57	88.77	92.34	80.62	87.20	91.03	24.36
+ R3-Reranker	**75.39**	**80.40**	**81.97**	**82.74**	85.23	89.94	92.76	83.41	88.48	91.52	**33.27**

Table 4: Complete results on R3-Skill (%). H@1 is Hit@1; N@k, R@k, C@k, and SC denote NDCG@k, Recall@k, Comp@k, and Set-Compat. The reranking block uses the same R3-Embedding top-20 candidates. Bold marks the best result in each column among the displayed systems.

BM25 (Robertson & Zaragoza, 2009), BGE-M3 (Chen et al., 2024), Qwen3 models (Qwen Team, 2025; Zhang et al., 2025), and released SkillRouter (Zheng et al., 2026) and SkillRet checkpoints.

All systems receive the same retrieval instruction. Queries are truncated to 512 tokens and skill documents to 4,096 tokens. R3-Embedding retrieves 20 candidates, matching the candidate budget used by SkillRouter and SkillRet, after which each reranker reorders the same pool. We report Hit@1, NDCG@K, Recall@K, and Comp@K under the same definitions as SkillRet. We further introduce **Set-Compat**, computed only when $|S_q^*| \geq 2$:

$$\mathrm{SC}(q) = \mathbb{I}\Big[S_q^* \subseteq \mathrm{top}_{|S_q^*|}(\pi_q)\Big].$$

Set-Compat requires all annotated jointly necessary skills within the first $|S_q^*|$ ranks and is therefore stricter than recall at a larger cutoff. It rewards ranking the complete required set ahead of optional nearby skills, which is the practical requirement for forming an action plan from a short list.

6.2 Main Results on R3-Skill

Table 4 separates large-pool retrieval from top-20 reranking. At the embedding stage, R3-Embedding achieves 72.07% Hit@1 and 81.60% NDCG@10, improving over the strongest base encoder, Qwen3-Embedding-8B, by 9.78 and 8.59 points, respectively. It also raises Recall@10 from 84.98% to 91.77%. Thus, routing-specific supervision and sibling-positive handling let the 0.6B model improve both early precision and candidate coverage beyond the 8B base.

External skill retrievers transfer unevenly to R3-Skill: SkillRet-Embedding beats the general-purpose 0.6B Qwen3 but trails the 8B base on Hit@1, and SkillRouter-Embedding is weak at all cutoffs. This reflects a benchmark mismatch—R3-Skill's bilingual queries, held-out skill pools, and multi-skill compatibility decisions are absent from their training resources, whereas R3-Embedding is trained on the benchmark's full-text, multi-positive structure.

Generic rerankers preserve some large-cutoff recall but reduce Hit@1 and Set-Compat. R3-Reranker instead raises Hit@1 to 75.39% and Set-Compat to 33.27%, relative gains of 4.6% and 18.3% over R3-Embedding, while keeping NDCG@10 nearly unchanged. The absolute Set-Compat value is modest by design: complete-set recovery is a deliberately hard target, and no baseline exceeds 25%. The generic reranker retains slightly higher Recall@10–15, but Set-Compat—not large-cutoff recall—is the deployment-relevant quantity: an agent loads only a short prefix, so ranking the complete required set into the top $|S_q^*|$ positions matters more than coverage at rank 15.

Direction	Hit@1		Set-Compat		ΔSC
	Emb.	+Rerank	Emb.	+Rerank	
en2en	73.41	75.07	32.87	36.33	+3.46
en2zh	69.91	74.95	18.97	25.64	+6.67
zh2en	86.67	93.33	45.45	63.64	+18.19
zh2zh	79.35	77.17	50.00	60.00	+10.00

Table 5: Cross-lingual breakdown on R3-Skill (%). "Emb." is R3-Embedding and "+Rerank" adds R3-Reranker.

Model	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15
Embedding									
BM25	46.47	48.86	49.90	50.09	56.55	59.95	34.96	41.09	44.27
Qwen3-Embedding-0.6B (base)	56.23	58.35	59.34	59.29	64.89	68.06	41.98	47.27	50.31
Qwen3-Embedding-8B (base)	57.57	59.98	61.05	60.71	67.06	70.45	43.33	50.01	53.69
SkillRouter-Embedding-0.6B	68.39	70.38	71.22	70.50	75.63	78.28	52.79	59.04	62.26
SkillRet-Embedding-0.6B	75.57	78.03	78.87	79.15	85.42	88.09	65.96	75.09	79.03
R3-Embedding	78.79	81.06	81.89	81.82	87.64	90.20	68.78	77.43	81.33
+ Rerank (upstream = R3-Embedding, top-20)									
+ Qwen3-Reranker-0.6B	73.24	76.03	76.97	78.34	85.69	88.70	64.52	74.28	78.47
+ SkillRouter-Reranker-0.6B	80.49	82.15	82.71	82.98	87.30	89.11	70.46	76.33	78.89
+ R3-Reranker	**82.41**	**83.87**	**84.38**	**85.47**	**89.36**	**91.06**	**75.48**	**81.01**	**83.15**

Table 6: Complete transfer results on SkillRet (%). Horizontal rules separate embedding retrieval from top-20 reranking with R3-Embedding as the shared upstream model.

6.3 Cross-Lingual Compatibility Analysis

The rejection taxonomy suggests two regimes. en2en SKIPs usually reflect domain-level mismatches, which a bi-encoder can represent as semantic distance. en2zh SKIPs more often describe forced chaining: the skills may be plausible in isolation or in other contexts, but do not form a jointly necessary set for the current query.

Table 5 supports this compatibility interpretation for the two high-volume directions. At the embedding stage, en2zh Set-Compat is 18.97%, below en2en at 32.87%. R3-Reranker improves Set-Compat by 6.67 points on en2zh, compared with 3.46 points on en2en, consistent with the prevalence of forced chaining in en2zh. The smaller zh2en and zh2zh slices are less stable, so we treat them as diagnostic rather than confirmatory evidence.

6.4 Cross-Benchmark Transfer

Table 6 evaluates transfer to an independently constructed English benchmark. At the embedding stage, R3-Embedding reaches 81.06% NDCG@10, exceeding SkillRet-Embedding-0.6B by 3.03 points and SkillRouter-Embedding-0.6B by 10.68 points. The gain is consistent across every cutoff and metric family. In particular, R3-Embedding improves Comp@10 from 75.09% to 77.43% over the SkillRet-trained encoder, showing that R3-Embedding transfers to complete-set retrieval even though SkillRet does not provide the same rejection supervision.

The result also separates task adaptation from scale. Qwen3-Embedding-8B reaches only 59.98% NDCG@10, whereas all three skill-specific encoders perform substantially better. R3-Embedding's improvement is therefore unlikely to come from backbone capacity; it reflects the retrieval data, full skill representation, and multi-skill objective.

Reranking exhibits a clear domain-adaptation hierarchy. The generic Qwen3 reranker lowers every NDCG metric relative to R3-Embedding and also hurts completeness, showing a general pair scorer can override a strong domain-specific retriever. SkillRouter-Reranker improves early NDCG but slightly reduces Recall@10 and Comp@10, so its ordering preference transfers only partially. R3-Reranker improves all nine metrics, reaching 83.87% NDCG@10, 89.36% Recall@10, and 81.01%

Stage	Negative source	H@1	N@10	R@10	SC
Embedding (InfoNCE)					
A	8B-mined	**72.07**	**81.60**	**91.77**	**28.12**
B	SKIP if available	69.93	79.94	90.90	21.58
Reranker (listwise)					
C	no SKIP labels	72.31	81.09	**91.08**	29.50
D	shuffled SKIP labels	75.11	81.93	89.78	29.70
E	SKIP labels	**75.39**	**81.97**	89.94	**33.27**

Table 7: Stage-wise SKIP ablation on R3-Skill (%). D shuffles E's matched SKIP partners; A/E are deployed.

Comp@10. Unlike the R3-Skill result, no recall trade-off appears on SkillRet—the reranker improves both early ordering and set coverage.

This cross-benchmark pattern strengthens the stage-wise interpretation. On R3-Skill, SKIP supervision mainly changes the smallest sufficient prefix and yields the largest gain in Set-Compat. On SkillRet, where evaluation uses larger-cutoff ranking and completeness metrics, the same reranker also improves broad coverage. The combined evidence shows that R3-Reranker's gains transfer to an evaluation setting without rejection annotations.

6.5 Where Should SKIP Supervision Enter?

Table 7 gives a clear stage-wise answer. With all else fixed, B substitutes a SKIP partner when available and otherwise keeps A's 8B-mined negative; it drops Hit@1 from 72.07% to 69.93% and Set-Compat from 28.12% to 21.58%. D matches E in language direction, required-set size, candidate similarity, skill length, and training exposure, but assigns label-1 candidates to different queries; relative to C it raises Hit@1/NDCG@10 from $72.31\%/81.09\%$ to $75.11\%/81.93\%$. Using the original query–SKIP pairing in E leaves these nearly unchanged but lifts Set-Compat from 29.70% to 33.27%. The C/D and D/E contrasts thus separate the benefit of adding matched hard candidates from that of preserving correct query–SKIP alignment, supporting SKIP labels for reranking rather than bi-encoder negatives.

7 Conclusion

We argued that skill routing differs from document retrieval because a successful result must satisfy both per-skill relevance and query-conditioned set compatibility. R3-Skill makes this distinction measurable by retaining the language-model rejection decisions that synthesis pipelines normally discard, and R3 separates efficient multi-positive recall from rejection-aware listwise reranking. Our central finding is not that rejection supervision is harmful, but that it must be matched to the component able to represent its conditional structure: forced into a single shared embedding it does not help, whereas in a query-conditioned cross-encoder it yields the largest gains at top ranks and in complete-set recovery, and these gains transfer to SkillRet, which carries no rejection annotations. Within the benchmark's current scope—synthetic queries, offline evaluation, small non-English pools, a rule-based taxonomy, and top-20 reranking—these results point to a broader principle: a generator's implicit judgment about what should *not* be used together is recoverable as first-class supervision.

AI use statement

In this work, we used generative AI tools for benchmark construction. DeepSeek-V4-Pro generates the training queries and SKIP annotations, and Qwen3-235B-A22B drafts the test queries and labels. The two model families are separated so that evaluation labels are not produced by the training generator. Multi-skill test labels were independently reviewed by multiple experts. We have reviewed the generated queries, rejection decisions, and the claims in this paper. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Ethics statement

Multi-skill test labels were independently reviewed by multiple experts. The queries are synthetic and the evaluation is offline. As stated in the benchmark design, the sampling procedure concentrates on difficult neighborhoods and is not an estimate of how often incompatible skill combinations occur in live traffic.

Reproducibility statement

Benchmark construction, the train and test skill split, the four language directions, and the rejection taxonomy are described in Section 4. Architectures, losses, and the division between recall and reranking are in Section 5. Metrics, baselines, the R3-Skill and SkillRet results, and the stage-wise ablation are in Section 6. Generation constraints on language, request style, length, and skill-name disclosure are stated with the WRITE/SKIP procedure. Appendix A discusses the gradient intuition, deployment cost, and Set-Compat.

References

Anthropic. Agent skills specification. `https://agentskills.io/specification`; companion repo: `https://github.com/anthropics/skills`, 2025a. Online specification document and reference repository.

Anthropic. anthropics/skills: Public repository for agent skills. `https://github.com/anthropics/skills`, 2025b.

Jianlv 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, 2024.

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

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. `https://huggingface.co/deepseek-ai/DeepSeek-V4-Pro/resolve/main/DeepSeek_V4.pdf`; `https://api-docs.deepseek.com/`, 2026.

Haichuan Hu, Ye Shang, and Quanjun Zhang. Red skills or blue skills? a dive into skills published on ClawHub. *arXiv preprint arXiv:2604.13064*, 2026.

Yanna Jiang, Delong Li, Haiyu Deng, Baihe Ma, Xu Wang, Qin Wang, and Guangsheng Yu. SoK: Agentic skills – beyond tool use in LLM agents. *arXiv preprint arXiv:2602.20867*, 2026.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

Fangzhou Li, Pagkratios Tagkopoulos, and Ilias Tagkopoulos. SkillFlow: Scalable and efficient agent skill retrieval system. *arXiv preprint arXiv:2504.06188*, 2025.

Hao Li, Chunjiang Mu, Jianhao Chen, Siyue Ren, Zhiyao Cui, Yiqun Zhang, Lei Bai, and Shuyue Hu. Organizing, orchestrating, and benchmarking agent skills at ecosystem scale. *arXiv preprint arXiv:2603.02176*, 2026a.

Xiangyi Li, Wenbo Chen, Yimin Liu, et al. SkillsBench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026b.

Yujian Liu, Jiabao Ji, Li An, Tommi Jaakkola, Yang Zhang, and Shiyu Chang. How well do agentic skills work in the wild: Benchmarking LLM skill usage in realistic settings. *arXiv preprint arXiv:2604.04323*, 2026.

Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Chengcheng Han, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Skill0: In-context agentic reinforcement learning for skill internalization. *arXiv preprint arXiv:2604.02268*, 2026.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 5835–5847, 2021.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Stephen Robertson and Hugo Zaragoza. *The Probabilistic Relevance Framework: BM25 and Beyond*. Now Publishers Inc., 2009.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 68539–68551, 2023.

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. *arXiv preprint arXiv:2604.24594*, 2026.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Kai Wei, Raymond Li, Xi Zhu, Zhaoqian Xue, Jiaojiao Han, Jingcheng Niu, and Fan Yang. SkillRAG: Failure-state-aware retrieval augmentation via hidden-state probing and skill routing. *arXiv preprint arXiv:2604.15771*, 2026.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*, 2020.

Renjun Xu and Yang Yan. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Hanrong Zhang, Shicheng Fan, Henry Peng Zou, Yankai Chen, Zhenting Wang, Jiayu Zhou, Chengze Li, Wei-Chieh Huang, Yifei Yao, Kening Zheng, Xue Liu, Xiaoxiao Li, and Philip S. Yu. CoEvoSkills: Self-evolving agent skills via co-evolutionary verification. *arXiv preprint arXiv:2604.01687*, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, et al. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

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. *arXiv preprint arXiv:2603.22455*, 2026.

Huichi Zhou, Siyuan Guo, Anjie Liu, Zhongwei Yu, Ziqin Gong, Bowen Zhao, Zhixun Chen, Menglong Zhang, Yihang Chen, Jinsong Li, Runyu Yang, Qiangbin Liu, Xinlei Yu, Jianmin Zhou, Na Wang, Chunyang Sun, and Jun Wang. Memento-skills: Let agents design agents. *arXiv preprint arXiv:2603.18743*, 2026.

This appendix provides a discussion of findings, deployment trade-offs, and Set-Compat (Appendix A), evaluation conventions (Appendix B), cutoff-wise results (Appendix C), benchmark comparison and the quality-assurance checklist (Appendix D), the bilateral-balancing derivation and alternative loss analysis (Appendix E), clustering, taxonomy, and query statistics (Appendix F), and all bilingual generation prompts verbatim (Appendix G).

A Discussion

Why SKIP conflicts with the bi-encoder. The embedding degradation is structural. For a SKIP partner s' under q, the InfoNCE gradient is $+p_{s'}\mathbf{e}(q)/\tau$; when s' is positive for another query $\tilde{q}$, it is $-(1-\tilde{p}_{s'})\mathbf{e}(\tilde{q})/\tau$. One shared vector must therefore reconcile a query-specific push with positive pulls elsewhere. The cross-encoder avoids this shared-vector constraint by recomputing a representation for each (q, s) pair; Appendix E gives the derivation.

Empirical findings and interpretation. A SKIP judgment does not mean every skill in the rejected set is globally irrelevant; it indicates the set is not jointly necessary for a given request. Treating this as a conventional contrastive negative asks a shared embedding to encode mutually inconsistent relations, whereas graded cross-encoder supervision lets the score vary with the query, aligning the model with the annotation semantics. The rejection taxonomy adds a diagnostic layer beyond aggregate accuracy: domain mismatch behaves like a conventional hard negative, while forced chaining more directly captures query-conditioned incompatibility. The larger en2zh Set-Compat gain fits this distinction, though the small zh2en and zh2zh subsets remain too limited for a strong language-specific claim. The D/E control sharpens this: matched hard candidates improve early ranking, whereas preserving the original query pairing yields the larger Set-Compat gain—linking rejection annotations to the short-list decision that matters when an agent assembles a small action plan.

Deployment trade-offs. Deployment need not always invoke both stages. On the complete 10,246-skill collection, reranking 20 candidates (0.20% of the pool) cuts query–skill comparisons by about $512\times$: at the evaluation caps, the 20 pairs require at most $20 \times (512 + 4{,}096) = 92{,}160$ input tokens per request, versus $10{,}246 \times (512 + 4{,}096) \approx 47.2$ million for full-pool scoring. These are capped input volumes, not measured latency, which also depends on hardware, batching, precision, indexing, and realized lengths. Use R3-Embedding alone for broad shortlist coverage, R3-Reranker for a prefiltered pool, and the full pipeline when both are required.

Why Set-Compat matters in deployment. An agent rarely loads a long skill list: each added skill consumes context and may introduce redundant or incompatible instructions. Set-Compat asks whether the annotated jointly necessary skills fit within the smallest actionable prefix. A set recovered only at ranks 10–15 can lift long-cutoff coverage yet rarely enters an action plan built from the leading candidates. In Table 4, R3-Embedding gives the strongest short-list coverage—highest Recall@5 (87.22%) and Comp@5 (85.31%)—while R3-Reranker attains the best Hit@1 (75.39%), NDCG, and Set-Compat (33.27%). The SC gain thus captures an ordering improvement directly useful under a small execution budget.

B Evaluation Details

Metrics. We use the same Recall@K and Comp@K definitions as SkillRet. Set-Compat is the stricter all-ground-truth indicator $\mathbb{1}[S_q^* \subseteq \text{top-}m]$ with $m = |S_q^*|$, evaluated only on multi-skill queries. NDCG@5/10/15 and Hit@1 follow standard definitions.

Lengths and prompt. Lengths are uniformly set to: query up to 512 tokens, doc up to 4096 tokens, skill body up to 4096 tokens. All baselines and our models are evaluated under the same instruction: `Instruct: Given a user request, retrieve the agent skill that solves it.` followed by the query.

Reranker pool and the rank trade-off. The reranker input pool is the top-20 from R3-Embedding ($K = 20$, matching the SkillRouter / SkillRet setting). Since the pool already covers the embedding top-10, every GT in the embedding top-10 is also in the reranker candidate set; aggregate R@10 / Comp@10 movement at the reranker stage is therefore not due to candidate truncation. Internally, the reranker promotes the entire GT set toward the top, but may move some individual GT items lower in the process—the rank trade-off paid for the gain in Set-Compat. Whether this surfaces as a small aggregate R@10 dip or a small lift depends on how saturated the embedding top-10 already is on a given test set.

C Complete Evaluation Results

Tables 8–11 give all evaluation cutoffs; Table 12 gives the by-direction breakdown.

Model	H@1	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15	SC
BM25	6.04	8.00	8.82	9.25	9.93	12.41	14.01	9.32	11.69	13.18	0.40
BGE-M3 (base)	39.59	46.94	49.20	50.27	53.88	60.72	64.67	51.62	58.55	62.57	6.93
Qwen3-Embedding-0.6B (base)	43.12	51.58	53.72	54.74	59.44	65.82	69.56	57.18	63.54	67.40	4.16
Qwen3-Embedding-8B (base)	62.29	71.05	73.01	73.74	79.13	84.98	87.65	76.56	82.79	85.71	10.69
SkillRouter-Embedding-0.6B	22.09	29.51	32.14	33.54	36.45	44.48	49.67	34.94	42.78	47.82	2.18
SkillRet-Embedding-0.6B	46.72	55.86	58.04	59.10	64.47	71.01	74.92	62.31	69.10	73.03	11.88
R3-Embedding	**72.07**	**80.06**	**81.60**	**82.05**	**87.22**	**91.77**	**93.38**	**85.31**	**90.40**	**92.22**	**28.12**

Table 8: Embedding on R3-Skill test (%), all cutoffs.

Model	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15
BM25	46.47	48.86	49.90	50.09	56.55	59.95	34.96	41.09	44.27
Qwen3-Embedding-0.6B (base)	56.23	58.35	59.34	59.29	64.89	68.06	41.98	47.27	50.31
Qwen3-Embedding-8B (base)	57.57	59.98	61.05	60.71	67.06	70.45	43.33	50.01	53.69
SkillRouter-Embedding-0.6B	68.39	70.38	71.22	70.50	75.63	78.28	52.79	59.04	62.26
SkillRet-Embedding-0.6B	75.57	78.03	78.87	79.15	85.42	88.09	65.96	75.09	79.03
R3-Embedding	**78.79**	**81.06**	**81.89**	**81.82**	**87.64**	**90.20**	**68.78**	**77.43**	**81.33**

Table 9: Embedding on SkillRet official test (%).

Reranker	H@1	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15	SC
Upstream embedding = R3-Embedding											
(no reranker, embedding only)	72.07	80.06	81.60	82.05	**87.22**	91.77	93.38	**85.31**	90.40	92.22	28.12
SkillRouter-Reranker-0.6B	67.98	75.57	77.65	78.63	82.57	88.77	92.34	80.62	87.20	91.03	24.36
Qwen3-Reranker-0.6B	68.08	77.61	79.54	80.11	86.45	**92.12**	**94.12**	84.46	**90.70**	**93.05**	17.82
R3-Reranker	**75.39**	**80.40**	**81.97**	**82.74**	85.23	89.94	92.76	83.41	88.48	91.52	**33.27**

Table 10: Rerankers on R3-Skill test (%) over the shared R3-Embedding top-20 pool.

Reranker	N@5	N@10	N@15	R@5	R@10	R@15	C@5	C@10	C@15
Upstream embedding = SkillRet-Embedding-0.6B									
(no reranker, embedding only)	75.57	78.03	78.87	79.15	85.42	88.09	65.96	75.09	79.03
Qwen3-Reranker-0.6B	72.84	75.81	76.68	78.23	85.86	88.57	64.72	75.48	79.93
Qwen3-Reranker-4B	73.24	76.20	77.12	78.58	86.18	89.04	65.52	76.09	80.47
Qwen3-Reranker-8B	73.21	76.09	76.92	79.12	86.48	89.07	66.40	76.53	80.67
SkillRouter-Reranker-0.6B	79.98	81.85	82.37	82.85	87.64	89.24	70.66	78.15	80.75
SkillRet-Reranker-0.6B	80.71	82.18	82.66	83.73	87.61	89.20	73.28	78.95	81.09
Upstream embedding = R3-Embedding									
(no reranker, embedding only)	78.79	81.06	81.89	81.82	87.64	90.20	68.78	77.43	81.33
Qwen3-Reranker-0.6B	73.24	76.03	76.97	78.34	85.69	88.70	64.52	74.28	78.47
SkillRouter-Reranker-0.6B	80.49	82.15	82.71	82.98	87.30	89.11	70.46	76.33	78.89
R3-Reranker	**82.41**	**83.87**	**84.38**	**85.47**	**89.36**	**91.06**	**75.48**	**81.01**	**83.15**

Table 11: Reranker on SkillRet official test (%). Grouped by upstream embedding.

Direction	Stage	H@1	N@10	R@5	R@10	C@5	C@10	C@15	SC
en2en	R3-Embedding	73.41	82.14	87.13	91.78	84.99	90.34	91.86	32.87
en2en	+ R3-Reranker	75.07	81.04	83.65	88.83	81.48	87.19	90.74	36.33
en2zh	R3-Embedding	69.91	80.44	86.71	91.43	84.97	90.05	92.29	18.97
en2zh	+ R3-Reranker	74.95	82.28	86.20	90.59	84.68	89.25	91.96	25.64
zh2en	R3-Embedding	86.67	92.87	97.46	98.73	96.19	98.10	98.10	45.45
zh2en	+ R3-Reranker	93.33	96.37	97.30	99.68	96.19	99.05	99.05	63.64
zh2zh	R3-Embedding	79.35	87.14	93.48	94.02	92.39	93.48	94.57	50.00
zh2zh	+ R3-Reranker	77.17	84.44	90.22	92.93	89.13	92.39	93.48	60.00

Table 12: Full cross-lingual by-direction results on R3-Skill (%).

D Benchmark Comparison and Quality Assurance

The main paper summarizes why R3-Skill differs from existing resources and reports the most important safeguards. The tables below provide an attribute-level comparison and the complete audit checklist.

Attribute	SkillRouter	SkillRet	R3-Skill (ours)
#skill	∼80,000	17,810	10,246
#WRITE query	75 eval[†]	63,259 train + 4,997 eval	41,592
#SKIP	—	—	32,828
LangDir	en	en	en2en / en2zh / zh2en / zh2zh
Reject Signal	× discarded	× discarded	✓ retained + 8-class taxonomy
Train / Test split	unreleased	no overlap	no overlap

Table 13: Dataset attribute comparison with concurrent skill retrieval datasets.

[†]SkillRouter releases only 75 eval queries; its 37,979 training pairs are unreleased.

Safeguard	Procedure and resulting figure
Cross-source LLMs for train/test	Train set generated by DeepSeek-V4-Pro; test set by Qwen3-235B-A22B—avoiding shared-generator bias between training and evaluation
Manual test-set verification	All multi-skill GT labels independently reviewed by nine experts; initial disagreement below 5%; disputed cases underwent expert re-review followed by a second round of independent scoring; overall query naturalness: 25% sampled review (LLM-simulated rather than real-user validated)
Duplicate query detection	Verified by `distinct == n`; final duplicate count is 0
Cross-lingual leakage check	Substring match of skill names against query text: en2en 7.8% / en2zh 2.3% / zh2zh 2.1% / zh2en 0.0%
No train/test pool overlap	Split is at the skill-pool level; skills in the test set never appear in training

Table 14: Quality assurance checks (summary of procedures already described in the main paper).

E Bilateral Balancing: Derivation and Alternative Loss Forms

This appendix states Theorem 1, derives its two gradients, and compares the InfoNCE candidate-pool injection with three alternative loss forms.

E.1 SETUP

Let $\mathbf{e} : \mathcal{S} \cup \mathcal{Q} \to \mathbb{S}^{d-1}$ be an L2-normalized shared encoder with temperature $\tau > 0$; the logit between query q and candidate s is $z_s := \mathbf{e}(q)^\top \mathbf{e}(s)/\tau$. The SKIP partner s' is included alongside the GT s^+ and random negatives $\{s_j^-\}$ in the candidate pool $\mathcal{C} = \{s^+, \{s_j^-\}, s'\}$, with InfoNCE softmax probability $p_{s'} = \exp(z_{s'}) / \sum_{c \in \mathcal{C}} \exp(z_c)$. Gradients below are taken with respect to the L2-normalized $\mathbf{e}(s')$; back-propagating further to the pre-normalization representation multiplies by the L2-normalization Jacobian but preserves the direction of the similarity change.

Theorem 1 (InfoNCE bilateral balancing on $\mathbf{e}(s')$)**.** *When s' is a SKIP partner under query q,*

$$\nabla_{\mathbf{e}(s')} \mathcal{L}_{NCE}^{(q)} = + \frac{p_{s'}}{\tau} \mathbf{e}(q),$$

pushing $\mathbf{e}(s')$ away from $\mathbf{e}(q)$. If the same s' also appears as a positive under another query $\tilde{q}$ (with softmax probability $\tilde{p}_{s'}$ on its candidate pool),

$$\nabla_{\mathbf{e}(s')} \mathcal{L}_{NCE}^{(\tilde{q})} = - \frac{1 - \tilde{p}_{s'}}{\tau} \mathbf{e}(\tilde{q}),$$

pulling $\mathbf{e}(s')$ toward $\mathbf{e}(\tilde{q})$.

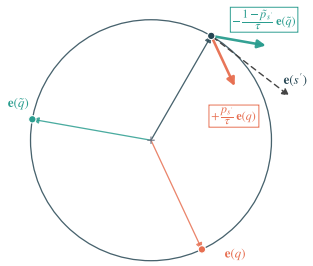


Figure 4: Bilateral balancing on the shared $\mathbf{e}(s')$: a SKIP push under query q is balanced by a positive pull under another query $\tilde{q}$; their equilibrium fixes the geometric position of $\mathbf{e}(s')$.

E.2 DERIVATION OF THEOREM 1

The InfoNCE loss on the candidate pool $\mathcal{C}$ is $\mathcal{L}_{\text{NCE}} = -\log p_{s^+}$. Among the logits $\{z_c\}_{c \in \mathcal{C}}$, only $z_{s'}$ depends on $\mathbf{e}(s')$, with $\partial z_{s'} / \partial \mathbf{e}(s') = \mathbf{e}(q)/\tau$. Differentiating yields

$$\nabla_{\mathbf{e}(s')} \mathcal{L}_{\text{NCE}}^{(q)} = - \frac{\partial \log p_{s^+}}{\partial \mathbf{e}(s')} = + \frac{p_{s'}}{\tau} \mathbf{e}(q),$$

the push-away gradient. When the same s' appears in another sample $(\tilde{q}, s'^+, \ldots)$ as the positive (so $s' = s'^+$ in that sample), the relevant logit is $z_{s'} = \mathbf{e}(\tilde{q})^\top \mathbf{e}(s')/\tau$, and differentiating $-\log p_{s'}$ gives

$$\nabla_{\mathbf{e}(s')} \mathcal{L}_{\text{NCE}}^{(\tilde{q})} = - \frac{1 - \tilde{p}_{s'}}{\tau} \mathbf{e}(\tilde{q}),$$

the pull-toward gradient.

The two gradients act on the same shared $\mathbf{e}(s')$ across batches: a SKIP push is weighted by $p_{s'}$, while a positive pull from $\tilde{q}$ is weighted by $1 - \tilde{p}_{s'}$, which is large when s' is not yet confidently ranked as the positive and vanishes as $\tilde{p}_{s'} \to 1$. The position of $\mathbf{e}(s')$ is the equilibrium of these query-specific forces (Figure 4); no single query can unilaterally dominate.

E.3 NEGATIVE RESULTS: LIMITATIONS OF OTHER FORMS ON A BI-ENCODER

- Independent listwise hinge $\max(0, m - \mathbf{e}(s)\cdot\mathbf{e}(q) + \mathbf{e}(s')\cdot\mathbf{e}(q))$: computed only on the SKIP row, it provides no opposite-direction gradient on rows where s' is a positive, leading to one-sided pushing.

- BCE compatibility loss $-\log \sigma(-\mathbf{e}(s)\cdot\mathbf{e}(s'))$: the gradient direction is along $-\mathbf{e}(s)$, independent of the query, and pushes globally; it disrupts the local neighborhood structure when s and s' co-occur as positives under other queries.
- Distillation form $\|\mathbf{e}(s)+\mathbf{e}(s')-\mathbf{e}_{\text{tgt}}\|^2$ (where $\mathbf{e}_{\text{tgt}}$ is a target vector from some external teacher model): the gradient direction is along $\mathbf{e}_{\text{tgt}}$, independent of the current query $\mathbf{e}(q)$, essentially pulling $\mathbf{e}(s)$ toward a fixed reference point—it lacks the "push away s' under q" semantics required by SKIP.

Among the bi-encoder variants we examined, the InfoNCE candidate-pool injection is the only form that is both query-aware in its gradient direction and bilaterally balanced. However, Theorem 1 also bounds what it can achieve: the shared $\mathbf{e}(s')$ has to compromise between "pushed away under q" and "pulled in under $\tilde{q}$", so the SKIP push from any single query is diluted and the gain is capped by the encoder's capacity. This is why we apply the SKIP signal at the R3-Reranker (cross-encoder) stage instead.

F R3-SKILL DIAGNOSTICS AND DATASET STATISTICS

This section expands the benchmark analysis in the main paper. It reports why $K = 40$ was selected, how the 40 clusters map to eight super-domains, how WRITE/SKIP annotation proceeds, and how query style and rejection reasons vary across slices.

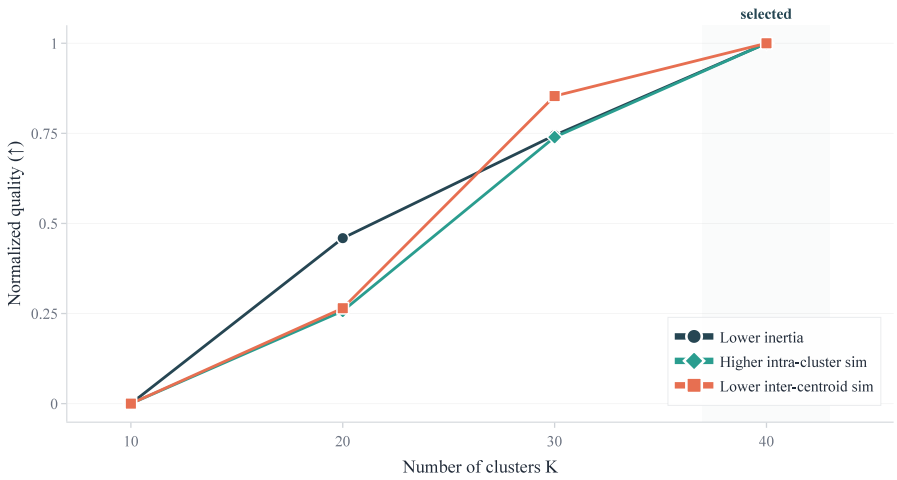


Figure 5: Cluster quality vs. K ($K \in \{10, 20, 30, 40\}$). The three indicators are inertia (within-cluster sum of squared distances, lower is better), intra-cluster sim (within-cluster topical similarity, higher is better), and inter-centroid sim (similarity between adjacent centroids, lower is better); for joint comparison, all three curves are normalized to [0,1] and re-oriented so that "higher is better". $K = 40$ is selected based on the joint criteria.

Table 15: Query length p50 by styles.

Style	en2en	en2zh	zh2zh	zh2en
task_direct	97	107	146	170
scene	170	164	190	209
role_setup	170	177	192	216
constraint	183	193	203	262
multi_step	180	197	213	226
debug	194	176	196	227

Table 16: SKIP statistics under different countings.

Protocol	k=1	k=2	k=3	Total
Raw SKIP	7,163	15,698	9,967	32,828
Classifiable SKIP	—	9,277	6,685	15,962
Training-split SKIPs	—	5,978	3,322	9,300
Evaluation-split SKIPs	—	—	—	16,365



Figure 6: Hierarchical taxonomy of the R3-Skill skill pool ($K = 40$). Inner ring: 8 super-domains (A–H); outer ring: 40 sub-clusters with arc lengths proportional to skill counts. Colors encode super-domain membership, underlying the sampling rule of "$k = 2$ same sub-cluster, $k = 3$ same super-domain".

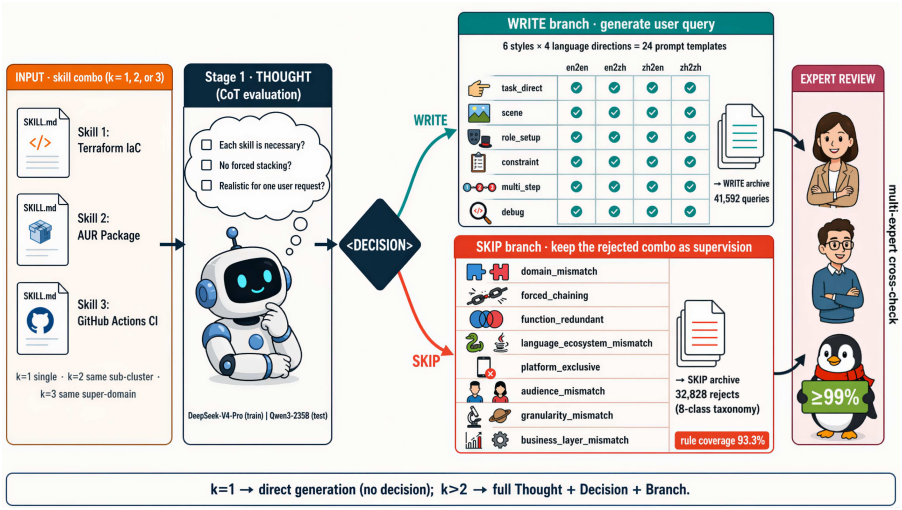


Figure 7: LLM annotation flow: skill set → DECISION; the WRITE branch generates queries in 6 styles × 4 directions, while the SKIP branch is archived; both then enter taxonomy classification and multi-expert verification.

Table 17: Representative WRITE / SKIP examples from R3-Skill.

Skill combination	Decision	Core LLM rationale	Reject reason class
Godot Multiplayer + Legal Policy Writer	SKIP	platform / domain mutual exclusion	*platform_exclusive*
Rust Production Reliability + Go Security Patterns	SKIP	inconsistent language ecosystems	*language_ecosystem_mismatch*
Code Review + Systematic Code Review	SKIP	overlapping functionality	*function_redundant*
Terraform IaC + AUR Package Publish	WRITE	forms a natural release pipeline	—
SendGrid Email + Xero Invoice	WRITE	forms a natural invoice-email flow	—

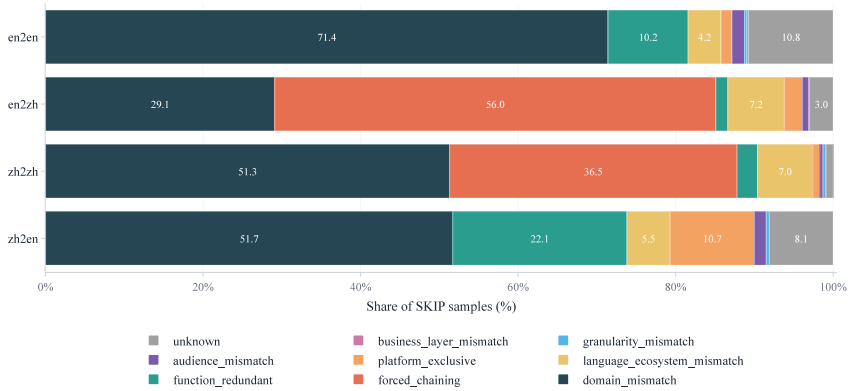


Figure 8: Stacked bar chart of the 8 reject reason classes across the 4 language directions. en2en is mostly assigned to *domain_mismatch*; en2zh contains many *forced_chaining*-style rejections (0.560).

Table 18: 8-class reject reason taxonomy across directions and k.

Reason	Meaning	en2en	en2zh	zh2zh	zh2en	k=2	k=3	Share
domain_mismatch	Domains do not overlap	71.7%	29.1%	51.3%	51.7%	45.3%	46.6%	45.9%
forced_chaining	Chinese "forced stacking" fallback	¡1%	56.0%	36.5%	¡5%	34.6%	33.0%	33.9%
function_redundant	Overlapping function; only one needed	10.2%	1.5%	2.6%	22.1%	4.7%	4.0%	4.4%
language_ecosystem_mismatch	Different languages / ecosystems	4.2%	7.2%	7.0%	5.5%	6.1%	5.1%	5.7%
platform_exclusive	Platform mutual exclusion	1.4%	2.3%	0.9%	10.7%	1.9%	1.9%	1.9%
audience_mismatch	Different user groups	1.6%	0.8%	0.4%	1.5%	1.4%	1.0%	1.2%
granularity_mismatch	Low-level vs. high-level	0.3%	0.1%	0.4%	0.4%	0.3%	0.1%	0.2%
business_layer_mismatch	Strategy vs. operations	0.2%	¡0.1%	¡0.5%	¡0.5%	0.1%	¡0.1%	0.1%
unknown	No rule hit	10.8%	3.0%	0.9%	8.1%	5.6%	8.8%	6.7%

G FULL R3-SKILL QUERY GENERATION PROMPTS

The prompts below are included verbatim for reproducibility. Their constraints operationalize three benchmark requirements discussed in the main paper: natural user phrasing, no direct skill-name disclosure, and an explicit joint-necessity judgment for multi-skill requests.

This appendix lists all prompt templates used in the data-generation stage of the main paper (LLM Annotation). There are 8 templates in total: 4 directions (en2en / en2zh / zh2en / zh2zh; skill language then query language) $\times$ (single ($k = 1$), multi ($k \geq 2$, two-stage CoT WRITE/SKIP)). Each template is filled with format substitutions before use: `style_desc` from the style table below, `k` for the skill count, and `skills_block` for a rendered listing of skills with *name* / *description* / *body*.

G.1 Style Descriptions (6 styles, bilingual)

Chinese style descriptions (used by zh2zh / en2zh).

```
task_direct: 任务直述：开门见山地告诉模型要做什么，可以省略主语，像在下指令。
scene:       场景描述：先简短交代背景或痛点，再引出需求；不要使用"作为某某"的句式。
role_setup:  角色设定：让模型扮演某个专家身份再来帮你，
             例如"你是一名资深 X，请帮我..."。
constraint:  约束条件式：把目标 + 输入 + 输出格式 / 风格 / 长度 / 限制条件等讲清楚。
multi_step:  多步骤式：把需求拆成 2~4 个有先后的小步骤，让模型按顺序处理。
debug:       排错求助式：贴一段疑似有问题的代码 / 配置 / 报错，让模型定位并修复
             （也可以只描述现象）。
```

English style descriptions (used by en2en / zh2en).

```
task_direct: Task-direct: state the task plainly, like an instruction.
scene:       Scene-first: briefly describe a context or pain point before stating
             the need; avoid 'As a ...' phrasing.
role_setup:  Role-setup: ask the model to act as some expert (e.g., 'You are a
             senior X, please help me ...').
constraint:  Constraint-style: spell out goal + input + output format / style /
             length / constraints.
multi_step:  Multi-step: break the request into 2~4 ordered substeps for the
             model to handle in sequence.
debug:       Debug-help: paste suspected-broken code / config / error and ask
             for diagnosis and fix (description-only is fine).
```

G.2 Single-skill prompts ($k = 1$)

zh2zh — Chinese skill $\rightarrow$ Chinese query.

```
你扮演一个真实的中文用户，正要去问一个 AI 助手帮忙做事。
你只能看到下方 skill 的能力（name / description / body 摘要），但你写出来的问题里
**绝对不能** 出现 skill 的 name 或与之高度同义的措辞，也不能照抄 description / body
的原句。

风格要求（必须严格遵循）：{style_desc}

写作硬约束：
1. 用中文写问题，长度 50~300 字；可以叠加 1~2 种次要风格让句子更自然，但主风格保持
   上面那条。
2. 不能出现 skill 的 name；不能出现明显从英文 name 直译过来的整词组合；公认缩写
   （PDF / API / SQL / GitHub 这类）允许。
3. 不要照抄 description / body 的整句话，可以重新换说法。
4. 必须能让 AI 助手读完之后判断"我应该用这个 skill 来帮他完成"。
5. 只输出问题本身，不要加任何前后缀、标签或解释。

Skill:
{skill_rendered}
```

en2zh — English skill $\rightarrow$ Chinese query.

```
你扮演一个真实的中文用户，正要去问一个 AI 助手帮忙做事。
下方给的是一份**英文 skill** 的能力（name / description / body 摘要），你需要根据它
来写一个**中文用户问题**，并且 **绝对不能** 出现 skill 的英文 name 或对它的中文直译，
也不能照抄 description / body 的原句。

风格要求（必须严格遵循）：{style_desc}
```

```
写作硬约束:
1. 用**中文**写问题，长度 50~300 字；可以叠加 1~2 种次要风格让句子更自然，但主风格
   保持上面那条。
2. 完全用中文，**不要出现英文单词**（公认缩写如 PDF / API / SQL / GitHub 这类允许）。
3. 不能出现 skill 的英文 name，也不要把它整词直译过来；可以从功能/场景去描述。
4. 不要照抄 description / body 的句子。
5. 必须能让 AI 助手判断"我应该用这个 skill"。
6. 只输出问题本身，不要加任何前后缀、标签或解释。

Skill:
{skill_rendered}
```

en2en — English skill → English query.

```
You play a real user asking an AI assistant for help.
You can only see the skill below (name / description / body excerpt). The
question you write **must not** mention the skill's name (or close synonyms),
and **must not** copy sentences from description / body verbatim.

Style requirement (strict):
{style_desc}

Hard constraints:
1. Write in English, 50~300 words. You may layer in 1~2 secondary styles to keep
   it natural, but stay anchored to the main style above.
2. Do not mention the skill's name or any near-synonym translation of it.
3. Do not copy whole sentences from description / body; rephrase.
4. The AI must be able to read your question and infer "I should use this skill".
5. Output only the question itself; no prefixes, tags, or explanations.

Skill:
{skill_rendered}
```

zh2en — Chinese skill → English query.

```
You play a real user asking an AI assistant for help in English.
The skill below is in **Chinese** (name / description / body excerpt). Write an
**English user question** based on it; you **must not** mention the skill's name
(or any direct English translation of it), and **must not** copy from description
/ body.

Style requirement (strict):
{style_desc}

Hard constraints:
1. Write in **English** only, 50~300 words. You may layer 1~2 secondary styles
   for naturalness, but the main style above must show.
2. No Chinese characters in the output.
3. Do not surface the skill's name or any direct translation of it; describe by
   function/scenario instead.
4. Do not copy sentences from description / body.
5. The AI must be able to read your question and infer "I should use this skill".
6. Output only the question itself; no prefixes, tags, or explanations.

Skill:
{skill_rendered}
```

G.3 Multi-skill prompts ($k = 2$ / $k = 3$, two-stage CoT WRITE/SKIP)

zh2zh — Chinese skill set → Chinese query.

```
你扮演一个真实的中文用户，可能正在做一个比较复杂的任务，需要多个能力配合。
下面给你 {k} 个 skill 的能力（name / description / body 摘要）。请按下面两阶段输出:

第一阶段（思考）：评估这 {k} 个 skill **同时**被一个用户在 **同一句话/同一段需求**
里自然提出来的合理性。判断标准:
  - 缺一不可：去掉任何一个，需求就完成不了 / 明显残缺;
  - 不强行拼凑：不能是"我顺手再来个 X"这种硬塞;
  - 真实合理：现实中真的会有用户这么问。

第二阶段（决定）：
  - 如果合理：在 <DECISION>WRITE</DECISION> 之后输出一段中文用户问题;
  - 如果不合理：在 <DECISION>SKIP</DECISION> 之后简短说明原因，不要再写问题。

写作硬约束（仅当 WRITE 时）：
```

```
- 风格要求：{style_desc}
- 用中文写，长度 50~300 字；可叠加 1~2 种次要风格；
- 必须**自然地同时**需要这 {k} 个 skill；不要列点式"我要 A、要 B、要 C"；
- 不能出现任何一个 skill 的 name；不要照抄 description / body 原句；
- 公认缩写（PDF / API / SQL / GitHub）允许。

输出格式（**严格**）：
<THOUGHT>
你的逐条评估和最后判断，3~6 句即可。
</THOUGHT>
<DECISION>WRITE</DECISION> 或 <DECISION>SKIP</DECISION>
<QUERY>
（只在 WRITE 时填，写问题本身；SKIP 时此节留空或写一句原因）
</QUERY>

下面是 {k} 个 skill：
{skills_block}
```

en2zh — English skill set → Chinese query.

```
你扮演一个真实的中文用户，正要去问 AI 助手做一个稍复杂的事。
下面给你 {k} 个**英文 skill** 的能力（name / description / body 摘要）。先用中文
思考它们组合是否合理，再决定是否写出**中文用户问题**。

第一阶段（思考）：评估这 {k} 个 skill **同时**被一个用户在 **同一句话/同一段需求**
里自然提出来的合理性。判断标准：
  - 缺一不可：去掉任何一个，需求就完成不了 / 明显残缺；
  - 不强行拼凑；
  - 真实合理。

第二阶段（决定）：
  - 合理 -> <DECISION>WRITE</DECISION> 然后写问题；
  - 不合理 -> <DECISION>SKIP</DECISION> 简短说原因。

写作硬约束（仅当 WRITE 时）：
- 风格要求：{style_desc}
- **用中文**写，长度 50~300 字；
- 不要出现英文单词（PDF / API / SQL / GitHub 等公认缩写允许）；
- 不要出现任何一个 skill 的英文 name 或直译；
- 不要照抄 description / body 的句子。

输出格式（**严格**）：
<THOUGHT>
3~6 句中文评估。
</THOUGHT>
<DECISION>WRITE</DECISION> 或 <DECISION>SKIP</DECISION>
<QUERY>
（仅 WRITE 时填）
</QUERY>

下面是 {k} 个 skill：
{skills_block}
```

en2en — English skill set → English query.

```
You play a real user asking an AI assistant to help with a somewhat compound
task. Below are {k} skills (name / description / body excerpt). Use a two-stage
output:

Stage 1 (think): Evaluate whether these {k} skills could plausibly be needed
**together** by **one user in one request**. Tests:
  - Each is necessary: remove any and the request becomes incomplete;
  - No forced stacking ("oh and also X");
  - Realistic: a real person would phrase this.

Stage 2 (decide):
  - Plausible  -> <DECISION>WRITE</DECISION> then write the user question.
  - Not plausible -> <DECISION>SKIP</DECISION> with a one-line reason; do not
    write a question.

Writing constraints (only when WRITE):
- Style: {style_desc}
- English, 50~300 words; you may layer 1~2 secondary styles;
- Naturally require **all** {k} skills; do not enumerate "I want A, B, C";
- Do not surface any skill's name; do not copy sentences from description
  / body.
```

```
Strict output format:
<THOUGHT>
3~6 sentences of evaluation.
</THOUGHT>
<DECISION>WRITE</DECISION> or <DECISION>SKIP</DECISION>
<QUERY>
(only when WRITE)
</QUERY>

The {k} skills:
{skills_block}
```

zh2en — Chinese skill set → English query.

```
You play a real user asking an AI assistant in **English** to help with a
compound task. The {k} skills below are described **in Chinese** (name /
description / body excerpt). Reason about their joint plausibility, then decide
whether to write an **English user question**.

Stage 1 (think): Evaluate whether these {k} skills could plausibly be needed
**together** by **one user in one request**. Tests:
  - Each is necessary;
  - No forced stacking;
  - Realistic.

Stage 2 (decide):
  - Plausible  -> <DECISION>WRITE</DECISION> then write the user question
    in **English**.
  - Not plausible -> <DECISION>SKIP</DECISION> with a one-line reason.

Writing constraints (only when WRITE):
- Style: {style_desc}
- English only, 50~300 words; no Chinese characters;
- Naturally require **all** {k} skills; no enumeration;
- Do not surface any skill's name (no direct translation either);
- Do not copy sentences from description / body.

Strict output format:
<THOUGHT>
3~6 sentences of evaluation (English).
</THOUGHT>
<DECISION>WRITE</DECISION> or <DECISION>SKIP</DECISION>
<QUERY>
(only when WRITE)
</QUERY>

The {k} skills:
{skills_block}
```