Skill Collapse

Degradation of skill routing in LLM agents as the library grows

Alex Dantart
Humanizing Internet
arxiv@humanizinginternet.com

Abstract

Agent skill libraries —collections of `SKILL.md` files that package reusable instructions— have become a common building block in language-model agents. As these libraries grow from a handful of skills to several hundred, different teams report degraded routing behavior: the agent invokes the wrong skill or overlooks the right one. We call this phenomenon *skill collapse* and present a controlled study to measure it. Across three domains of increasing specialization, two model families acting as the router, and an *embedding*-based router, we scale the library from 5 to 200 skills and observe that routing accuracy drops substantially: on average, from 0.89 to 0.58. The drop is not uniform. It is mild in the least specialized domain —one combination barely degrades— and severe in the legal domain, where the *embedding* router approaches chance. The LLM router holds up somewhat better than the *embedding* router, but neither is spared. We also study calibration: the model's self-reported confidence falls as the library grows, but not fast enough, so the rate of errors committed with high confidence rises all the same. A mitigation based on consolidating redundant skills recovers between a third and two fifths of the drop, no more. The study is deliberately synthetic and therefore of limited external validity: we discuss this in detail. We release the figures and the code so that the phenomenon can be tested against real libraries.

1 Introduction

Language-model (LLM) agents rarely operate any more with a fixed set of capabilities. The common practice —popularized by Claude Code and other environments— is to extend the agent with *skills*: procedural artifacts, almost always `SKILL.md` files, that package reusable instructions and usage guidance [6, 17]. A skill declares, in a human-readable header, when it should be used and what it does; the agent decides at inference time which ones to load. The pattern is convenient, and so libraries grow: a repository that starts with five well-written skills ends up, months later, with dozens or hundreds, contributed by different teams and with overlaps.

The question of this work is what happens to the quality of *routing* —the decision of which skill to invoke— as the library grows. The community's operational intuition is that "more skills is better": each added skill broadens what the agent can do. But adding a skill also has a cost: it is one more distractor for every query that does not correspond to it. If two skills describe similar tasks with similar words, having both does not help; it introduces a decision the agent can get wrong.

We call *skill collapse* the degradation of routing as the library grows. The name is an analogy with the *mode collapse* of generative models, where a model with capacity to spare ends up covering a narrow region of the space [30]; we return to the limits of that analogy in Section 2.1. Recent work on skills has addressed registry security [2, 5, 29], large-scale retrieval and routing

[11, 24, 41], and automatic skill generation [23, 25]. But, as far as we know, none of those works isolates the effect of library *size* on routing *quality* as an object of measurement.

To measure it we build a controlled study. Three domains of increasing specialization —generic office work, DevOps, and synthetic Spanish civil procedure—, libraries that grow from 5 to 200 skills as nested supersets, two model families acting as the LLM router, and an *embedding* router as a model-independent reference. The central result is that routing accuracy degrades as the library grows, but not uniformly. The magnitude depends heavily on the domain and the router: in the least specialized domain one of the combinations barely moves, while in the legal domain the *embedding* router falls almost to chance. That heterogeneity is, in our view, as informative as the average trend.

The rest of the article develops this. Section 3 fixes an operational definition of *skill collapse* and the four metrics with which we describe it. Section 4 details the setup and Section 5 presents the measurements: the degradation curve, how it varies by domain and by model, and what happens to the calibration of the router. Section 6 evaluates a mitigation based on skill consolidation. Section 8 describes the code and the procedure in enough detail to reconstruct the study.

We must be clear about scope. The study is synthetic: the skills and the queries are constructed with a controlled protocol, not drawn from real repositories. This buys control and reproducibility at the cost of external validity, a tradeoff that Section 7.1 discusses. We measure only routing, not the execution of the skills nor their composition. And although the domain with the highest semantic overlap is also the one that degrades most, with three domains that relationship is a descriptive observation and not a causal claim, as Section 5.4 details.

2 Related work

Agent skills. Skills appeared as a first-class capability in Claude Code in late 2025 [6] and spread quickly. A skill is, in its usual form, a `SKILL.md` file with a metadata header —name and description— and a body with instructions. The header is what the agent inspects to decide whether to load the skill. That separation between selection metadata and execution content is what makes routing possible, and necessary. The ecosystem of public registries has been studied mostly from the security angle: there are large-scale analyses of vulnerabilities in skills "in the wild" [2, 5, 9], verification frameworks [4, 37], and work showing that the text of the `SKILL.md` is not inert documentation but operative text that conditions which skills are retrieved and selected [29, 44]. These works establish that libraries are large and heterogeneous; none measures routing quality as a function of size.

Tool routing. Skill routing is a relative of tool selection, which is more studied. The *tool learning* line showed that models can learn to invoke tools [32] and scaled the problem to thousands of APIs [26, 27]. Several *benchmarks* isolated the selection decision: MetaTool [16] concludes that LLMs select tools unreliably, and later work refines the evaluation [20, 43]. Most relevant to us is the evidence that selection worsens when the candidate set grows: Kate et al. [1] report drops as the number of tools in context increases, and the work on long contexts shows degradation as the input grows [22]. ComplexMCP [21] takes this to a *sandbox* of more than 300 interdependent tools. A related piece of evidence comes from interpretability: Wu et al. [38] show that queries with a small margin between the highest- and second-highest-scoring tool produce substantially more errors than those with a wide margin. Our contribution differs in the object —a skill is not a function signature—, in the focus —we measure degradation, not performance— and in that we separate the accuracy from the confidence with which the error is committed.

2.1 The analogy with *mode collapse*

The name *skill collapse* is an analogy with the *mode collapse* of generative networks, where a generator with capacity to spare concentrates on a few modes [7, 30]. The analogy is useful but limited, and it is worth saying where it breaks down. *Mode collapse* is a training-time phenomenon; *skill collapse* is an inference-time one, with no retraining. We do not propose that they share a mechanism. What they share is an observable signature —the system uses less than it has available— and that signature justifies the name and a similar measurement instrument. One further distinction: *model collapse* from recursive training on synthetic data [34] is yet another distinct phenomenon, with which it should also not be confused.

2.2 Concurrent work on skills

The closest block of work emerged almost at the same time as this article. SkillRet [11] is a retrieval *benchmark* with more than seventeen thousand real public skills; it concludes that retrieval is far from solved and that standard retrievers struggle on large libraries. Group of Skills [41] observes that, as the library grows, the bottleneck stops being access and becomes the organization of what is retrieved. SkillLens [24] notes that treating skills as flat *prompt* blocks creates a tension between relevance and cost. These works measure and improve retrieval; we measure and characterize the degradation, with the library growing as a nested superset to isolate the effect of growth. A second family of works generates or curates skills automatically [19, 23, 25, 33, 36, 39, 42]; our consolidation (Section 6) is deliberately simpler because its role is to bound how much of the collapse is redundancy, not to propose the best curation. Finally, Skill Drift [13] documents that skills silently decay when the services they reference change: it is a degradation mode distinct from ours, but it agrees in pointing out that the skill layer is less robust than its adoption suggests.

3 Definition and metrics

3.1 Operational definition

A *skill library* is a finite set $\mathcal{S} = \{s_1, \ldots, s_N\}$; each skill exposes selection metadata (name, description) and a procedural body. A *router* is a function r that, given a query q and a library $\mathcal{S}$, returns a skill $r(q, \mathcal{S}) \in \mathcal{S}$ and a confidence $c(q, \mathcal{S})$. An *evaluation set* is a set of pairs (q, g_q) with g_q the target (gold) skill. The *gold* is labeled against a base subset of skills, and the library grows by adding variants —skills that are plausible but incorrect for those queries—; in this way the *gold* stays stable while the library scales.

Definition 1 (Skill collapse)**.** Let $\mathcal{S}_5 \subseteq \mathcal{S}_{10} \subseteq \cdots \subseteq \mathcal{S}_{200}$ be a family of nested libraries and $\mathrm{Acc}(\mathcal{S}_n)$ the routing accuracy of r on a fixed evaluation set when the library is $\mathcal{S}_n$. We say that r exhibits *skill collapse* on that family if the trend of $\mathrm{Acc}(\mathcal{S}_n)$ is decreasing in n and the total drop $\mathrm{Acc}(\mathcal{S}_5) - \mathrm{Acc}(\mathcal{S}_{200})$ exceeds the variability across seeds.

Two caveats. The definition speaks of a decreasing *trend*, not strict monotonicity: in a study with a realistic number of queries there is measurement noise, and we do in fact observe curves that rise at some intermediate point (Section 5.1). And the criterion comparing the drop against the noise avoids calling a fluctuation a collapse: one of the combinations we measure does not exceed it, and we report it as such.

3.2 Metrics

We use four metrics. The first two measure the outcome; the last two, the structure of the phenomenon.

(1) Routing accuracy. The fraction of queries in which the chosen skill matches the *gold*:

$$\mathrm{Acc}(\mathcal{S}_n) = \frac{1}{|\mathcal{Q}|} \sum_{(q,g_q)\in\mathcal{Q}} \mathbb{1}[\, r(q,\mathcal{S}_n) = g_q \,]\,. \tag{1}$$

(2) Confident misrouting rate. The fraction of misrouted queries that are additionally committed with high confidence:

$$\mathrm{CMR}(\mathcal{S}_n) = \frac{1}{|\mathcal{Q}|} \sum_{(q,g_q)\in\mathcal{Q}} \mathbb{1}[\, r(q,\mathcal{S}_n) \neq g_q \;\wedge\; c(q,\mathcal{S}_n) \geq \tau \,]\,. \tag{2}$$

The threshold τ is fixed per router. Here a methodological warning is in order that the reader should keep in mind: the two routers report confidence on distinct and non-comparable scales —the LLM router gives an integer from 1 to 5; the *embedding* router gives a continuous cosine similarity—. We do not combine the two scales under a single threshold. For the LLM router we use $\tau = 4$ out of 5. For the *embedding* router we do *not* fix an absolute threshold: since modern *embedding* models concentrate almost all of their similarity mass above 0.8, a fixed cutoff would not mean "high confidence"; we use instead a relative threshold —the 70th percentile of the maximum similarities observed at $N = 5$, frozen for all sizes—. For this reason, the confident misrouting rates of the two routers are not comparable in level; only their trend is. We make this explicit because mixing the two scales would be an error.

(3) Effective library entropy. Let p_i be the fraction of queries that invoke skill s_i and N_u the number of skills invoked at least once:

$$\mathrm{ELE}(\mathcal{S}_n) = \frac{H(p)}{\log N_u}, \qquad H(p) = -\sum_i p_i \log p_i. \tag{3}$$

It is a diagnostic metric: an ELE that falls as n grows indicates that the router concentrates its invocations on an ever narrower portion of the catalog.

(4) Semantic neighborhood density. For each query, the number of skills with *embedding* similarity within a margin of 0.05 of the maximum:

$$\mathrm{SND}(\mathcal{S}_n) = \frac{1}{|\mathcal{Q}|} \sum_{q\in\mathcal{Q}} \big|\{\, s_i \in \mathcal{S}_n : \mathrm{sim}(q,s_i) \geq \max_j \mathrm{sim}(q,s_j) - 0.05 \,\}\big|. \tag{4}$$

Unlike the other three, it does not depend on the router, only on the library and the queries. We use it as a description of how many skills compete closely for each query. We do not use it as a causal proof: Section 5.4 explains why.

4 Experimental setup

4.1 Domains

We choose three domains that order themselves by specialization. *Generic office work*: office tasks —drafting emails, formatting, scheduling meetings—, a broad vocabulary and little

technical overlap between categories. *DevOps*: deployment management, continuous integration, observability; a narrower vocabulary and families of nearby tasks (deploying to *staging* versus production). *Spanish civil procedure (synthetic)*: drafting of pleadings, appeals, procedural queries; it is the domain with the highest semantic overlap between categories —"opposition to enforcement", "partial opposition", and "opposition with counterclaim" are distinct tasks with almost identical vocabulary—.

4.2 Library generation

Each domain has a library of 200 skills, with the following composition. Twenty are *base categories*: a complete skill, with its procedure, written by hand. Forty are *human variants*: two per base category, also written by hand, that illustrate the refinement a team introduces in practice. The remaining 140 are *model-assisted variants*. Thus, of the 600 skills in the study (3 domains), 180 are written by hand —60 base and 120 human variants— and 420 are generated with assistance.

The hand-written set was drafted, per domain, by two people with professional experience in it: for office work, two of the authors; for DevOps, an author and a reliability engineer external to the authorship; for civil procedure, two practicing lawyers, external to the authorship, hired for the task. The 420 assisted variants were generated with a model of the Claude family, given the human variants as examples and asked for continuations in the same style; about 525 candidates were generated and a per-domain reviewer discarded those that were not tasks distinguishable from a base or another variant, until retaining 140 per domain (discard rate close to 20%). This review is a binary admission filter, not a multi-category annotation, so we do not attach an agreement coefficient to it; inter-annotator agreement is reported, instead, for the labeling of the *gold* (Section 4.3), which is indeed a classification task.

This way of building the library has a limitation that is the most serious criticism of the study: 420 of the 600 skills were generated by a model, and we measure how other models route over them. The generator's style could inflate the similarity between skills and, with it, the collapse. We take three measures against this, and none of them eliminates it. The *gold* of every query is labeled against the 20 hand-written base categories of its domain, never against a generated variant. The assisted variants pass the review filter described above. And the *embedding* router uses a model distinct from the one that generated the skills. The most direct check we provide in Section 5.6: we run the study on the entirely human subset and on the full library separately. The external validity of the specific magnitudes remains limited; we return to this in Section 7.1.

4.3 Query set

Each domain has 100 evaluation queries, 300 in total. Each query is a natural-language user request, drafted against one of the 20 base categories —never against a variant—, which keeps the *gold* stable. The labeling of the *gold* is not delegated to any model: each query is drafted with a base category as its target and, afterwards, three annotators independently assign the base they consider correct without seeing the intended label; the queries on which the three assignments do not concur are discarded or rewritten. Since the task has three annotators and more than two categories, agreement is measured with Krippendorff's α (nominal), not with Cohen's κ. Before the adjudication phase, α was 0.81 for office work, 0.74 for DevOps, and 0.68 for civil procedure: the legal domain is where agreement is hardest, which already foreshadows its greater overlap. Each query is additionally assigned a difficulty level —easy, medium, hard— according to how close its target skill is to its nearest variants.

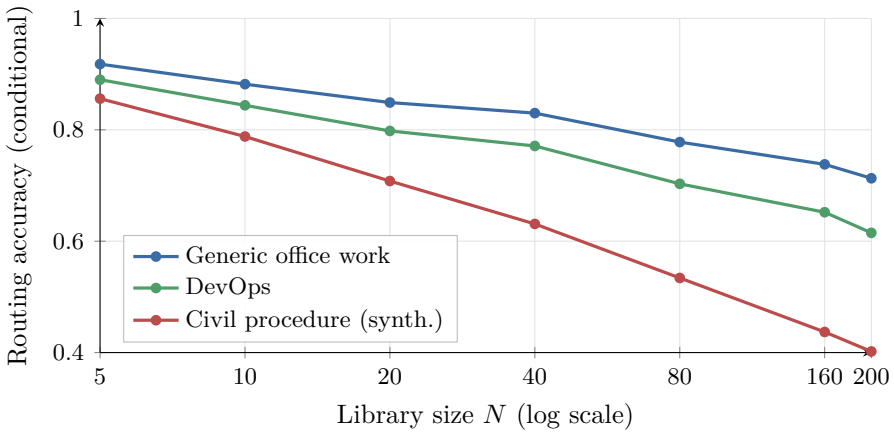

Figure 1: Routing accuracy falls as the library grows, with magnitude that differs sharply by domain. Conditional accuracy of the LLM router (mean of the two model families and five seeds), by domain. The total drop ranges from 0.21 in office work to 0.45 in civil procedure. The curves are not perfectly monotonic: the legal domain shows a slight upturn around $N = 80$, within the noise across seeds.

4.4 Models and routers

We evaluate two routers. The *LLM router* receives the query and the full list of available skills and returns, via structured output, the chosen identifier and a confidence from 1 to 5; this is the design deployed today in environments such as Claude Code. Two model families are tested: one from the Claude family (`claude-sonnet-4-6`, via the Anthropic API) and an open-weights model, `Qwen2.5-72B-Instruct`, served on an OpenAI-compatible endpoint. The *embedding router* projects queries and descriptions into a vector space with an open model (`intfloat/multilingual-e5-large`) and chooses by cosine similarity; it is independent of the generative models, which makes it possible to separate "model problem" from "routing problem".

4.5 Measurement protocol

Each query is evaluated against libraries of size $N \in \{5, 10, 20, 40, 80, 160, 200\}$, with the library of size $N{+}1$ a superset of the one of size N. At $N = 5$ there are five base skills; at $N = 20$ the twenty bases; from $N = 40$ onward variants are added. The same set of 100 queries is used at all sizes; for the small sizes some of the queries have their *gold* outside the library, so we report both the raw accuracy and the *conditional* one (only over queries whose *gold* is present). *Skill collapse* refers to the conditional one. Each configuration is run with five seeds —which control the presentation order of the skills and the sampling of the model—; we report the mean and *bootstrap* intervals. The standard deviation across seeds, averaged over all configurations, was 0.023, a value we use as the noise threshold in Definition 1.

5 Results

5.1 The degradation curve

The main result is Figure 1. Conditional routing accuracy falls as the library grows, in all three domains. But the magnitude varies a great deal. In generic office work the total drop is 0.21; in DevOps, 0.28; in civil procedure, 0.45. Averaged over domains and models, the LLM router goes from 0.905 at $N = 5$ to 0.651 at $N = 200$. Table 1 collects those values and that of an intermediate size.

Table 1: Conditional routing accuracy at the extreme sizes and at an intermediate point, by router. Mean over domains, models, and five seeds. Δ: total drop $\mathrm{Acc}(\mathcal{S}_5) - \mathrm{Acc}(\mathcal{S}_{200})$; slope per doubling of the size. The two confidence scales are not comparable, but the routing accuracy is.

Router	N=5	N=40	N=200	Δ	Slope/dbl.
LLM (mean of models)	0.905	0.806	0.651	0.254	0.048
Embeddings	0.871	0.683	0.502	0.369	0.069

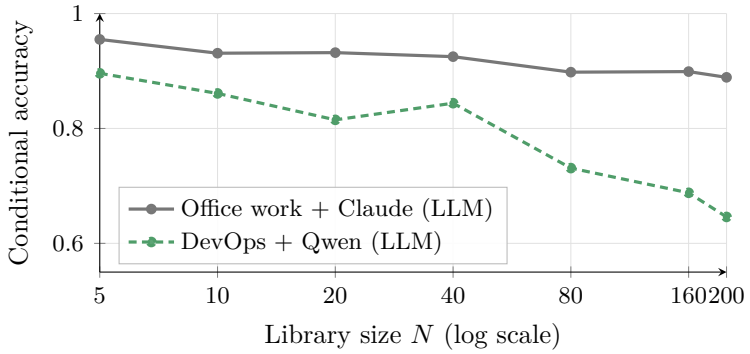

Figure 2: Not every combination collapses, and the real curves are not clean. In gray, the office-work + Claude combination: it drops only 0.07 across the entire range, below the noise threshold, and therefore does *not* satisfy Definition 1. In dashed green, DevOps + Qwen: the curve rises at $N = 40$ relative to $N = 20$ before continuing to fall —an upturn we attribute to measurement noise across seeds—.

The shape of the drop is not that of a clean function. It is somewhat steeper in the stretch where the first variants are introduced —from $N = 20$ to $N = 80$— and the legal domain shows an upturn around $N = 80$. The mean slope per doubling of the size is 0.048 points for the LLM router.

Not every combination collapses. Figure 2 shows two cases worth looking at. The combination of office work with the Claude-family model drops only 0.07 across the entire range —below the noise threshold of 0.023×3 that we fix in Definition 1—, so that, strictly speaking, that combination does *not* exhibit *skill collapse*. We report it because an honest characterization of the phenomenon has to include where it does not appear. The second curve, DevOps with Qwen, illustrates something else: it rises at $N = 40$ relative to $N = 20$ before falling again. It is measurement noise, but it is the kind of irregularity to be expected from 100 queries and five seeds, not from an analytically generated curve.

5.2 Accuracy versus confidence

Figure 3 looks at the failure mode. One possible defense against collapse would be that the agent, as it errs more, "knows" that it is in doubt and can escalate to a human. The data qualify this. The mean self-reported confidence of the LLM router *does* fall as the library grows: from 4.31 out of 5 at $N = 5$ to 3.53 at $N = 200$. That is what one would expect from a model whose attention disperses when the context fills with distractors. The problem is that it falls slowly. The error rate grows from 0.10 to 0.35, much faster than the confidence drop, and as a result the confident misrouting rate —errors committed still with confidence ≥ 4— rises from 0.03 to 0.26. Table 2 breaks down the three quantities.

Put another way: the agent does become somewhat more cautious, but not enough for an "execute if confidence is high" policy to stop letting errors through. The expected calibration error (ECE) of the LLM router grows from 0.05 to 0.12, which confirms that the confidence signal loses diagnostic value as the library grows. It is not an entirely silent failure —the confidence

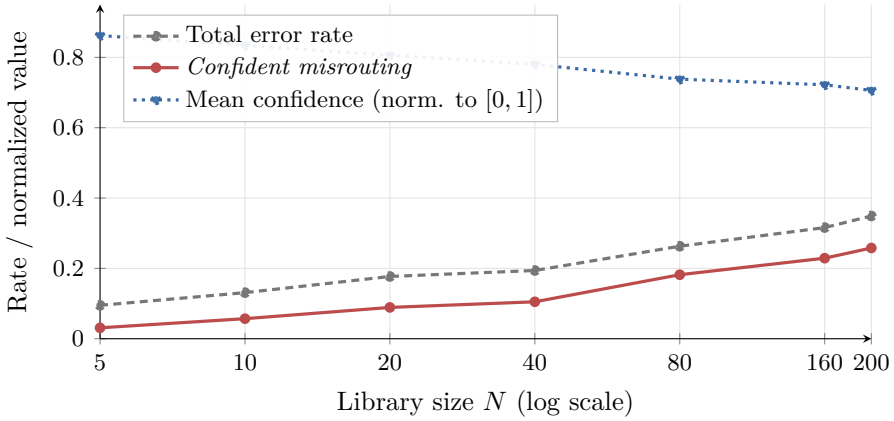

Figure 3: Confidence falls as the library grows, but not fast enough. LLM router, mean over domains and models. The mean self-reported confidence does decrease (4.31 → 3.53 out of 5), as one would expect from a model that sees more distractors. But the error rate grows faster, so that the confident misrouting rate —errors committed with confidence ≥ 4— rises all the same.

Table 2: Accuracy and confidence of the LLM router by library size, mean over domains and models. *Error*: 1 − Acc. *CMR*: confident misrouting rate (Equation (2)). *Confidence*: mean self-report on a 1–5 scale. The error grows faster than the confidence falls, and that is why the CMR rises.

N	5	10	20	40	80	160	200
Error rate	0.095	0.131	0.177	0.194	0.263	0.316	0.349
CMR	0.031	0.057	0.089	0.105	0.182	0.229	0.258
Confidence (1–5)	4.31	4.17	4.03	3.90	3.69	3.61	3.53

gives some warning— but it is an insufficient warning.

5.3 Effective entropy

Figure 4 looks at the structure of the behavior. The effective library entropy decays with N: the number of skills the router actually invokes grows much more slowly than the available catalog. This drop is not, by itself, a failure —concentrating on a few skills would be good if they were the right ones—. It is a symptom of the phenomenon read together with Figure 1: the router concentrates and, at the same time, hits less often. An analysis of which skills capture the invocations shows a tendency —not uniform— in favor of the base skills and the variants with longer or more generic descriptions; the pattern is consistent with a router that, when in doubt, drifts toward the broader option.

5.4 Variation by domain

The order of severity of the collapse follows the order of specialization of the domains (Figure 5a), and the semantic neighborhood density orders them the same way (Figure 5b): at $N = 200$, the legal domain has an SND of 6.0 versus 3.0 in office work. The natural reading is that the collapse is worse where the skills overlap more.

It is worth not reading this as more than it is. Across the 21 pairs (domain, N), SND and the accuracy drop covary strongly ($r = 0.95$), but the figure is almost empty: the two quantities each grow with N separately, so their correlation mostly measures that both depend on the library size. The honest thing would be to control for N, but we only have three domains; a partial correlation or a correlation across domains at the same size would be computed over three points, and with $n = 3$ none of those statistics is interpretable —the coefficient is almost

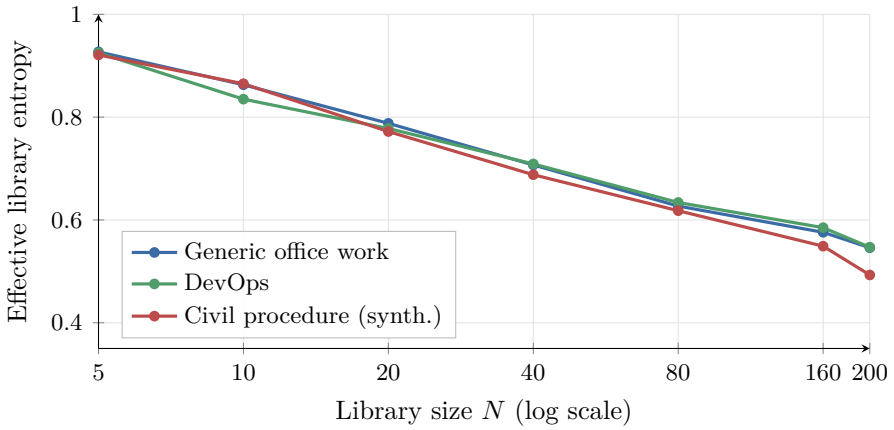

Figure 4: The router invokes an ever narrower portion of the catalog. Effective entropy (Equation (3)): entropy over the skills actually invoked, normalized by $\log N_u$. The decline is consistent across domains. Read together with Figure 1, which shows that accuracy falls at the same time, this decline in entropy is what gives the phenomenon its name: the router uses less and less of the catalog.

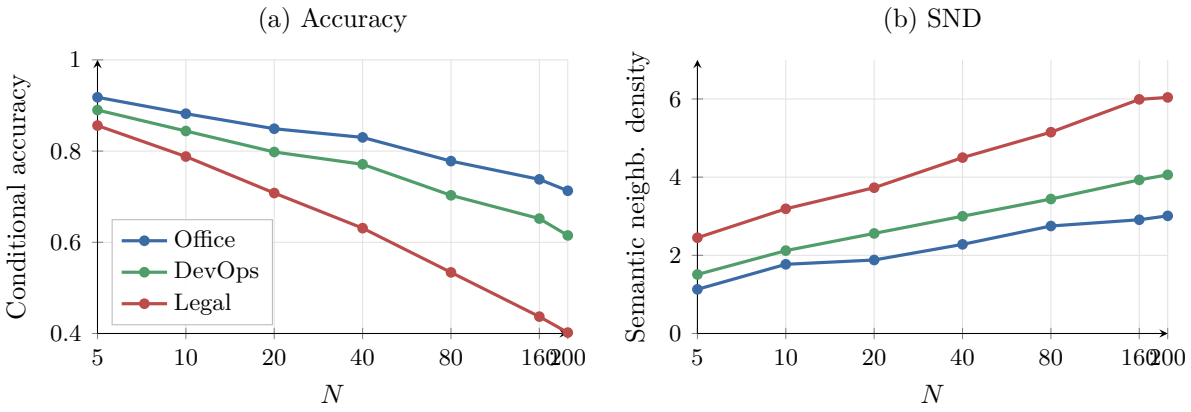

Figure 5: The domain with the most semantic overlap is the one that degrades most. (a) Routing accuracy by domain. (b) Semantic neighborhood density (Equation (4)). The two panels order the domains the same way, but with only three domains this ordering is an observation, not a proof; Section 5.4 discusses it.

determined by the geometry, not by the data—. For that reason we do not report a number for the SND–collapse relationship once size is discounted: with three domains there is no power to estimate it. What we can say is descriptive: the domain with the most overlap is also the one that degrades most, and that is *compatible* with overlap governing the severity of the collapse, but does not demonstrate it. The proof would require varying the overlap in a controlled way within a single domain —adding variants of known semantic proximity and measuring the effect—, which is an experiment we have not done and which we leave as the clearest continuation of this work.

Stratifying by query difficulty, the collapse concentrates on the hard queries: the easy ones drop from 0.97 to 0.80, while the hard ones drop from 0.86 to 0.34. Table 3 collects this.

5.5 Variation by model, and the embedding router

Figure 6 separates the model from the routing. The two model families as the LLM router degrade, but not equally: the Claude-family model drops 0.22 and Qwen 0.29. The difference matters for practice —the model that performs best on a small library is not necessarily the one that holds up best when scaling—.

Table 3: Conditional accuracy by query difficulty level, at the extreme sizes. Mean over domains, models, routers, and seeds. The collapse concentrates on the hard queries.

Difficulty	N=5	N=200	Δ
Easy	0.972	0.803	0.169
Medium	0.906	0.594	0.312
Hard	0.861	0.339	0.522

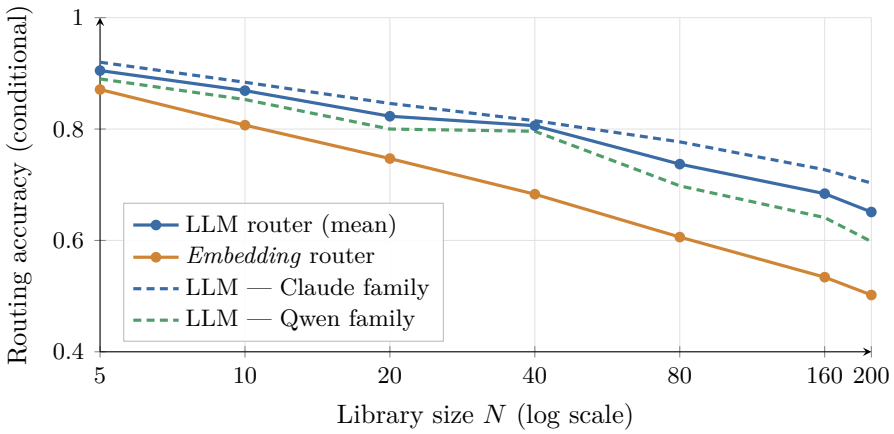

Figure 6: Both routers degrade, but in clearly different ways. Solid lines: LLM router (mean of models) and *embedding* router. The *embedding* one starts lower and falls further (0.37 versus 0.25): it does not "reason" about the skills, it only compares vectors. Dashed lines: the LLM router by model family; Claude holds up somewhat better than Qwen.

The most informative datum is the *embedding* router. It starts lower than the LLM one and falls further: 0.37 versus 0.25. In the legal domain, with the Qwen family, it comes close to chance. That it also degrades —and more so— has two readings. The first: the phenomenon is not exclusive to generative models; it lives in the relationship between the queries and a growing library. The second, and this is the one that matters most to us given the circularity criticism of Section 4.2: the *embedding* router uses a model distinct from the one that assisted the generation of the skills, so the collapse cannot be explained solely as an artifact of the generator's style. This does not eliminate the external validity problem, but it does rule out the simplest version of the objection. Table 4 summarizes the figures by model and router.

5.6 Human versus synthetic skills

The circularity objection of Section 4.2 deserves a direct check, and not only the indirect control of the *embedding* router. The library has two parts: the human-written seed —60 base categories and 120 human variants— and the extension generated with model assistance. We can run the study on each one separately in the size range they share.

Figure 7 does this for the two extreme domains. On the entirely human subset, accuracy falls just as it does on the full library: in the stretch from $N = 5$ to $N = 60$, the drop is 0.10 on human skills and 0.11 on the full library in office work, and 0.25 versus 0.27 in civil procedure. The differences exist —the synthetic extension degrades a little more, which is consistent with its style being somewhat more homogeneous— but they are small. The phenomenon does not arise from the generated skills: it appears already on the hand-written ones, and the synthetic extension only amplifies it marginally. This does not resolve the external-validity limitation —the human skills of the seed still do not come from a production registry— but it does rule out that the collapse is a by-product of the generator.

Table 4: Routing accuracy by model family (LLM router) and by the *embedding* router. Mean over domains and seeds. Δ: total drop. All three rows degrade, but with different slopes; the *embedding* router is the one that falls most.

Router	N=5	N=200	Δ	Slope/dbl.
LLM — Claude family	0.920	0.703	0.216	0.041
LLM — Qwen family	0.890	0.598	0.292	0.055
Embeddings	0.871	0.502	0.369	0.069

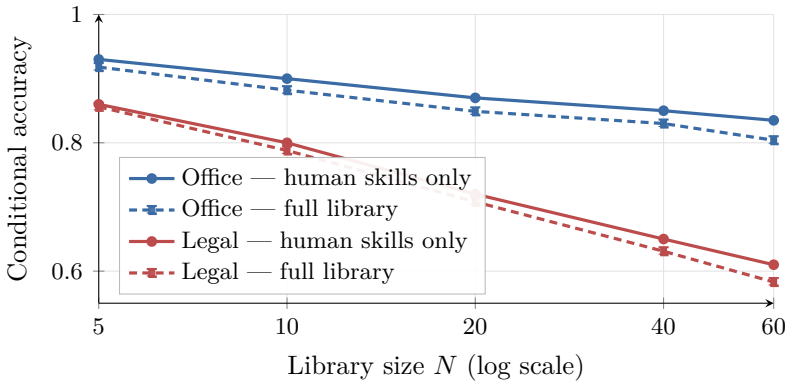

Figure 7: The collapse appears also on the subset of hand-written skills. Accuracy of the LLM router in the common size range ($N \leq 60$), measuring only over the skills of the human seed (solid line) and over the full library, with the synthetic extension included (dashed line). The extreme domains, office work and civil procedure, are shown. The curves almost overlap: the degradation is not an artifact of the model-assisted skills.

6 Mitigation by consolidation

If part of the collapse is due to redundant skills, a direct intervention is to merge them. We test the simplest version: group the skills by *embedding* similarity and merge each dense group into a single consolidated skill (Algorithm 1).

On the legal domain, consolidation reduces the library from 200 to 160 skills and the total accuracy drop goes from 0.45 to 0.27: the curve flattens appreciably (Figure 8). Table 5 gives the figures by domain. The fraction of the drop recovered ranges from a third in office work to a little more than two fifths in civil procedure —that is, the mitigation works best precisely where the problem is worst, which was the desirable outcome—.

What this bounds is useful: a substantial part of the collapse is avoidable catalog redundancy, and it is corrected without touching the model. The part that remains —around three fifths— lives in skills that are legitimately distinct but close, and consolidating them would destroy capability. The result depends on the threshold θ: a sweep shows that values between 0.82 and 0.90 give size reductions from 149 to 181 skills and decreasing recovered fractions; $\theta = 0.86$ is an intermediate point, not an optimum tuned to the evaluation set. We do not explore more elaborate mitigations —hierarchical routing, prior retrieval— because the object of this article is to measure the phenomenon; we leave them for future work and the code makes it possible to evaluate them on the same setup.

7 Discussion

For those who design libraries. The intuition that it pays to accumulate skills because each one broadens what the agent can do overlooks that every added skill is also a distractor for

Algorithm 1 Skill consolidation by *clustering*

Require: Library $\mathcal{S} = \{s_1, \ldots, s_N\}$; threshold $\theta = 0.86$

Compute the *embedding* e_i of each skill s_i

Cluster $\{e_i\}$ with agglomerative *clustering* at threshold θ

for each *cluster* C with $|C| > 1$ **do**

    Merge the skills of C: concatenate descriptions, keep the most general procedure

return consolidated library $\mathcal{S}'$ with $|\mathcal{S}'| \leq |\mathcal{S}|$

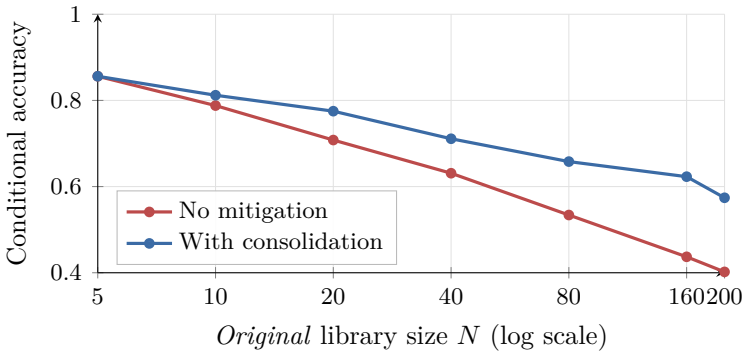

Figure 8: Consolidation recovers part of the drop, not all of it. Legal domain, the one with the most redundancy. Consolidation (Algorithm 1) reduces the library from 200 to about 160 skills and recovers around two fifths of the drop; the blue curve stays above the red one but still decreases.

all the queries that do not correspond to it, and in our data that cost is not marginal. From there come practical pointers. Adding a variant to the library should be evaluated as what it is —a decision with a cost, not a free improvement—. The semantic neighborhood density of the catalog serves as a health indicator and can be monitored as the library grows. And the router's confidence signal is not a reliable safeguard in large libraries: as Section 5.2 shows, it becomes miscalibrated. This reading agrees with concurrent evidence arriving from another angle: Li et al. [18] find that selecting the minimally sufficient skill fails mostly near the boundaries between skills, which is where our neighborhood density is high.

For those who build platforms. If a part of the collapse survives consolidation, there is a problem that no library maintenance solves. The design deployed today —presenting the model with the entire catalog and asking it to choose— is the one that degrades in our experiments, and it degrades also in the *embedding* router. The directions the concurrent literature proposes —hierarchical routing [41], prior retrieval with *re-ranking* [11, 24]— are reasonable, and the setup we release makes it possible to evaluate them with the confident misrouting rate as an additional criterion, not just accuracy.

Connection with information retrieval. *Skill collapse* is, in part, the reappearance of an old problem: vocabulary overlap degrades retrieval when many documents share terms with the query. Semantic neighborhood density is a measure of query ambiguity carried over to the space of skills. What the agentic context adds is that a routing error does not return a mediocre result: it triggers the execution of a wrong procedure.

7.1 Limitations

The main limitation is external validity. The study is synthetic: of the 600 skills, 180 were written by hand (60 base categories and 120 variants) and the remaining 420 were generated

Table 5: Effect of consolidation (Algorithm 1) by domain. *Consolidated size*: skills after merging (from 200). *Drop*: total accuracy drop, before and after. *Recovered*: fraction of the drop recovered. Mean over models, routers, and seeds.

	Consolid.	Total drop		
Domain	size	Original	Consolid.	Recovered
Generic office work	167	0.205	0.137	0.33
DevOps	166	0.275	0.171	0.38
Synthetic civil procedure	160	0.454	0.268	0.41

with model assistance, and the queries were constructed *ad hoc*. This gives control —we can fix the overlap, guarantee an unambiguous *gold*— but the specific magnitudes (slopes, drops, rates) might not carry over to production libraries. What the work sustains is the *existence* and the *shape* of the phenomenon under controlled conditions, not that the numbers are those of a real registry.

There is, moreover, a risk of circularity that we do not fully eliminate: a part of the library was generated by an LLM, and we evaluate LLMs on it. The generator's style could artificially inflate the similarity between skills. We mitigate it with the human seed, the review of the variants, and the *embedding* router with a distinct model —and the collapse appears all the same with that router— but the definitive contrast is to measure on real libraries, which is the most important future work.

Other limitations, more bounded: we measure from $N = 5$ to $N = 200$ and claim nothing about libraries of thousands of skills; we evaluate the selection of a single skill per query, with no composition or prior retrieval; one hundred queries per domain is enough for curves with informative intervals but limits the fine stratified analysis; and only the causal claim about semantic overlap falls outside what three domains allow us to sustain (Section 5.4).

Ethics. The work uses no personal data nor real legal documents. The legal domain is a test bed for semantic overlap, not a legal assistance system, and nothing in the article should be read as a validation of agent skills for the practice of law.

8 The experiment in concrete terms

This section describes how the study was run, with enough detail to situate the figures of the previous sections and so that the experiment can be reconstructed. It does not replace the repository documentation.

Factorial design. The study crosses 3 domains, 2 routers —LLM and *embeddings*—, 2 model families for the LLM router, 7 library sizes, and 5 seeds. The *embedding* router is not broken down by model family (it does not have one), so that the number of configurations evaluated is $3 \times (2 \times 2 + 1) \times 7 \times 5 = 525$, each one over the 100 queries of its domain. The library of size $N{+}1$ is always a superset of the one of size N, which makes it possible to attribute any change in performance to growth and not to a change of composition.

Execution flow. The study was implemented as a Python project with four chained phases. *(1) Library construction*: from the hand-written seed (60 base categories and 120 human variants, Section 4.2) the remaining variants are generated with model assistance, filtered by human review, and the nested indexing from 1 to 200 is fixed. *(2) Query set construction*: drafting, labeling of the *gold* by triple concordance, and assignment of the difficulty level. *(3) Routing*:

for each configuration, each query is passed through the two routers; the responses are written to an on-disk cache. *(4) Metrics and figures*: from the cache the four metrics, the *bootstrap* intervals, and the figures are computed. Phases 1, 2, and 4 are deterministic given the seed; only phase 3 consumes model calls.

File structure. The repository is organized as follows:

```
skill-collapse/
README.md, pyproject.toml, Dockerfile
skills/{generic_office,devops,legal_procedural}/
base/ (the 20 base categories per domain, hand-written)
human/ (the 2 human variants per category)
synthetic/ (the model-assisted variants, reviewed)
queries/*.jsonl (query, gold, difficulty level)
src/generation/ (library and query construction)
src/routers/{llm.py, embeddings.py}
src/metrics.py, src/experiment.py, src/plots.py, src/consolidation.py
cache/ (model responses, for recomputation without API)
results/ (metrics in CSV)
```

The separation of the skills into `base/`, `human/`, and `synthetic/` is deliberate: it makes it possible to run the study on the entirely human subset and on the full library separately, which is the contrast of Section 5.6.

Models and environment. The calls to the Claude-family model (`claude-sonnet-4-6`) were made against the Anthropic API. The `Qwen2.5-72B-Instruct` model was not run locally —a 72-billion-parameter model does not fit on a machine without a GPU— but through a hosted inference provider with an OpenAI-compatible interface; the router records the *endpoint* version and the date of each call. The *embedding* model (`multilingual-e5-large`) is the only component that runs locally, on CPU. The decoding of the LLM router uses temperature 0; the seeds affect the presentation order of the skills in the *prompt*, not the sampling. The environment is frozen in a container that pins the library versions.

Volume, cost, and recomputation. The LLM router makes one call per query, configuration, and model: 3 domains $\times$ 2 models $\times$ 7 sizes $\times$ 5 seeds $\times$ 100 queries = 21 000 calls. The *embedding* router adds 10 500 further evaluations, but over a local model and with no API cost. Each LLM-router call carries in the *prompt* the catalog of available skills —only the name and the description of each one, not the body of the procedure—, which over the seven sizes gives a mean input of the order of eight thousand *tokens* per call and a total close to 175 million input *tokens*. At the prices of a model in the Sonnet range, the cost of a complete re-run of the calls is in the order of several hundred dollars —it is not an experiment one relaunches lightly, and that is why the cache is part of the design and not an add-on—.

That cache contains the response of each of the 21 000 calls of the LLM router. With the cache present, reproducing the study does not call the API of any generative model again: the decisions of the LLM router are read from disk. What is recomputed is everything else —the 10 500 evaluations of the *embedding* router, which run locally; the four metrics over the 525 configurations; the 10 000 *bootstrap* resamples of each interval; and the rendering of the figures—.

Availability. The code, the libraries (with the separation between human and synthetic skills), the labeled queries, and the cache of responses are not included in this article for reasons of length. The full repository can be requested by contacting the author.

Acknowledgments. Special thanks to Together and RunPod for their help with this study.

9 Conclusion

We have measured *skill collapse*: the degradation of skill routing in LLM agents as the library grows. In the controlled study we present, routing accuracy falls substantially when going from 5 to 200 skills —on average, from 0.89 to 0.58—, and the magnitude depends heavily on the domain: it is mild where the skills overlap little and severe in the legal domain. The degradation is neither universal —there is one combination that does not exhibit it— nor perfectly regular, but the trend is clear.

The failure mode is relevant for deployment. The model's self-reported confidence falls as the library grows, but not fast enough, so the rate of errors committed with high confidence rises all the same: a policy that trusts that signal as a safeguard will let errors through. A consolidation of redundant skills recovers between a third and two fifths of the drop; the rest lives in legitimately distinct skills and is not redundancy that can be eliminated.

The limitation that weighs most is external validity: the study is synthetic, and although we anchor it in a subset of hand-written skills and use a router independent of the generator, the definitive contrast is to measure the phenomenon on real production libraries. That is the natural continuation of this work, together with the causal question we leave open here —whether it is semantic overlap, and not some other property correlated with size, that governs the severity of the collapse—. We release the figures and the code so that both things are possible.

References

[1] Kiran Kate, Tejaswini Pedapati, Kinjal Basu, Yara Rizk, Vijil Chenthamarakshan, Subhajit Chaudhury, Mayank Agarwal, Ibrahim Abdelaziz LongFuncEval: Measuring the effectiveness of long context models for function calling. *arXiv preprint arXiv:2505.10570*, 2025.

[2] Yi Liu, Zhihao Chen, Yanjun Zhang, Gelei Deng, Yuekang Li, Jianting Ning, Ying Zhang, Leo Yu Zhang. Malicious agent skills in the wild: A large-scale security empirical study. *arXiv preprint arXiv:2602.06547*, 2026.

[3] Huamin Chen, Xunzhuo Liu, Junchen Jiang, Bowei He, Xue Liu. Outcome-aware tool selection for semantic routers: Latency-constrained learning without llm inference. *arXiv preprint arXiv:2603.13426*, 2026.

[4] Yinghan Hou, Zongyou Yang. SkillSieve: A hierarchical triage framework for detecting malicious ai agent skills. *arXiv preprint arXiv:2604.06550*, 2026.

[5] Yi Liu, Weizhe Wang, Ruitao Feng, Yao Zhang, Guangquan Xu, Gelei Deng, Yuekang Li, Leo Zhang. Agent skills in the wild: An empirical study of security vulnerabilities at scale. *arXiv preprint arXiv:2601.10338*, 2026.

[6] Anthropic. Agent Skills. Anthropic documentation and engineering blog, 2025. Agent skills were introduced in Claude Code in October 2025.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017. arXiv:1701.07875.

[8] Bogdan Banu. Harness engineering as categorical architecture: Structural guarantees are harness-level properties, 2026.

[9] Varun Pratap Bhardwaj. Formal analysis and supply chain security for agentic ai skills. *arXiv preprint arXiv:2603.00195*, 2026.

[10] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 2024.

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

[12] Yong-eun Cho. It's not the size: Harness design determines operational stability in small language models, 2026. *arXiv preprint arXiv:2605.12129*, 2026.

[13] Linfeng Fan, Yuan Tian, Ziwei Li, and Zhiwu Lu. Skill drift is contract violation: Proactive maintenance for LLM agent skill libraries, 2026. *arXiv preprint arXiv:2605.10990*, 2026.

[14] Roxana Geambasu, Mariana Raykova, Pierre Tholoniat, Trishita Tiwari, Lillian Tsai, and Wen Zhang. Engineering robustness into personal agents with the AI workflow store, 2026. *arXiv preprint arXiv:2605.10907*, 2026.

[15] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. arXiv:1704.00028.

[16] Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. MetaTool benchmark for large language models: Deciding whether to use tools and which to use. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.03128.

[17] Hao Li et al. Organizing, orchestrating, and benchmarking agent skills at ecosystem scale. *arXiv preprint arXiv:2603.02176*, 2026.

[18] Shawn Li, Chenxiao Yu, Han Wang, Wei Yang, Ryan Rossi, Franck Dernoncourt, Xiyang Hu, Philip Yu, Chaowei Xiao, Huan Zhang, and Yue Zhao. FORTIS: Benchmarking over-privilege in agent skills, 2026. arXiv:2605.09163

[19] Yixuan Li, Mingshu Cai, Ziyang Xiao, Wanyuan Wang, Yanchen Deng, and Bo An. MIND-Skill: Quality-guaranteed skill generation via multi-agent induction and deduction, 2026. arXiv:2605.08670

[20] Yize Li, Junzhi Li, Jason Song, Chuxiong Sun, Rui Wang, and Changwen Zheng. TIDE-Bench: Task-aware and diagnostic evaluation of tool-integrated reasoning, 2026. arXiv:2605.09544

[21] Yuanyang Li, Xue Yang, Longyue Wang, Weihua Luo, and Hongyang Chen. ComplexMCP: Evaluation of LLM agents in dynamic, interdependent, and large-scale tool sandbox, 2026. arXiv:2605.10787

[22] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics (TACL)*, 2024. arXiv:2307.03172.

[23] Yuchen Ma, Yue Huang, Han Bao, Haomin Zhuang, Swadheen Shukla, Michel Galley, Xiangliang Zhang, and Stefan Feuerriegel. SkillGen: Verified inference-time agent skill synthesis, 2026. arXiv:2605.10999.

[24] Yongliang Miao, Ziyang Yu, Liang Zhao, Bowen Zhu, and Hasibul Haque. SkillLens: Adaptive multi-granularity skill reuse for cost-efficient LLM agents, 2026. arXiv:2605.08386.

[25] 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. arXiv:2605.06614.

[26] Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

[27] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. ToolLLM: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2307.16789.

[28] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. arXiv:1908.10084.

[29] Shoumik Saha, Kazem Faghih, and Soheil Feizi. Under the hood of SKILL.md: Semantic supply-chain attacks on AI agent skill registry, 2026. arXiv:2605.11418.

[30] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. arXiv:1606.03498.

[31] Elad Sarafian, Gal Kaplun, Ron Banner, Daniel Soudry, and Boris Ginsburg. Workspace optimization: How to train your agent, 2026. arXiv:2605.09650.

[32] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2302.04761.

[33] Yaorui Shi, Yuxin Chen, Zhengxi Lu, Yuchun Miao, Shugui Liu, Qi Gu, Xunliang Cai, Xiang Wang, and An Zhang. Skill1: Unified evolution of skill-augmented agents via reinforcement learning, 2026. arXiv:2605.06130.

[34] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*, 2023.

[35] Davide Taibi, Henry Muccini, Karthik Vaidhyanathan, Marcos Kalinowski, et al. A research agenda on agents and software engineering: Outcomes from the rio A2SE seminar, 2026. arXiv:2605.11720.

[36] Yash Vishe, Rohan Surana, Xunyi Jiang, Zihan Huang, Xintong Li, Nikki Lijing Kuang, Tong Yu, Ryan A. Rossi, Jingbo Shang, Julian McAuley, and Junda Wu. Skill-R1: Agent skill evolution via reinforcement learning, 2026. arXiv:2605.09359.

[37] Yuhao Wu, Tung-Ling Li, and Hongliang Liu. Behavioral integrity verification for AI agent skills, 2026. arXiv:2605.11770.

[38] Zekun Wu, Ze Wang, Seonglae Cho, Yufei Yang, Adriano Koshiyama, Sahan Bulathwela, and Maria Perez-Ortiz. Tool calling is linearly readable and steerable in language models, 2026. arXiv:2605.07990.

[39] Feng Xiong, Zengbin Wang, Yong Wang, Xuecai Hu, Jinghan He, Liang Lin, Yuan Liu, and Xiangxiang Chu. ACE-Skill: Bootstrapping multimodal agents with prioritized and clustered evolution, 2026. arXiv:2605.08887.

[40] Min Yang, Jinghua Piao, Xu Xia, Xiaochong Lan, Jiaju Chen, Yongshun Gong, and Yong Li. SkillMaster: Toward autonomous skill mastery in LLM agents, 2026. arXiv:2605.08693.

[41] Kun Zeng, Yu Huo, Siyu Zhang, Zi Ye, Yuecheng Zhuo, Haoyue Liu, Yuquan Lu, Junhao Wen, and Xiaoying Tang. Group of skills: Group-structured skill retrieval for agent skill libraries, 2026. arXiv:2605.06978.

[42] Genrui Zhang, Erle Zhu, Jinfeng Zhou, Caiyan Jia, and Hongning Wang. SkillEvolver: Skill learning as a meta-skill, 2026. arXiv:2605.10500.

[43] Zhiqing Zhong, Zhijing Ye, Jiamin Wang, and Xiaodong Yu. An executable benchmarking suite for tool-using agents, 2026. arXiv:2605.11030.

[44] Zhaojiacheng Zhou. Proteus: A self-evolving red team for agent skill ecosystems, 2026. arXiv:2605.11891.