Skill Collapse

Degradación del enrutamiento de habilidades en agentes LLM al crecer la biblioteca

Alex Dantart
Humanizing Internet
arxiv@humanizinginternet.com

Resumen

Las bibliotecas de habilidades de agentes —colecciones de ficheros `SKILL.md` que empaquetan instrucciones reutilizables— se han vuelto una pieza común en los agentes basados en modelos de lenguaje. A medida que estas bibliotecas pasan de un puñado de habilidades a varios cientos, distintos equipos reportan un comportamiento degradado del enrutamiento: el agente invoca la habilidad equivocada o pasa por alto la correcta. Llamamos a este fenómeno *skill collapse* y presentamos un estudio controlado para medirlo. Sobre tres dominios de especialización creciente, dos familias de modelos como enrutador y un enrutador por *embeddings*, escalamos la biblioteca de 5 a 200 habilidades y observamos que la precisión de enrutamiento cae de forma sustancial: en promedio, de 0,89 a 0,58. La caída no es uniforme. Es leve en el dominio menos especializado —una combinación apenas se degrada— y severa en el dominio jurídico, donde el enrutador por *embeddings* se acerca al azar. El enrutador LLM resiste algo mejor que el de *embeddings*, pero ninguno se salva. Estudiamos además la calibración: la confianza autorreportada del modelo baja al crecer la biblioteca, pero no lo bastante deprisa, de modo que la tasa de errores cometidos con confianza alta crece igual. Una mitigación por consolidación de habilidades redundantes recupera entre un tercio y dos quintos de la caída, no más. El estudio es deliberadamente sintético y, por tanto, de validez externa limitada: lo discutimos en detalle. Publicamos las cifras y el código para que el fenómeno pueda contrastarse sobre bibliotecas reales.

1. Introducción

Los agentes basados en modelos de lenguaje (LLM) rara vez operan ya con un conjunto fijo de capacidades. La práctica habitual —popularizada por Claude Code y otros entornos— consiste en extender al agente con *habilidades*: artefactos procedimentales, casi siempre ficheros `SKILL.md`, que empaquetan instrucciones y guías de uso reutilizables [6, 17]. Una habilidad declara, en un encabezado legible, cuándo debe usarse y qué hace; el agente decide en tiempo de inferencia cuáles cargar. El patrón es cómodo, y por eso las bibliotecas crecen: un repositorio que empieza con cinco habilidades bien escritas acaba, meses después, con decenas o cientos, contribuidas por equipos distintos y con solapamientos.

La pregunta de este trabajo es qué le ocurre a la calidad del *enrutamiento* —la decisión de qué habilidad invocar— cuando la biblioteca crece. La intuición operativa de la comunidad es que "más habilidades es mejor": cada habilidad añadida amplía lo que el agente puede hacer. Pero añadir una habilidad tiene también un coste: es un distractor más para todas las queries que no le corresponden. Si dos habilidades describen tareas parecidas con palabras parecidas, tener las dos no ayuda; introduce una decisión que el agente puede equivocar.

Llamamos *skill collapse* a la degradación del enrutamiento al crecer la biblioteca. El nombre es una analogía con el *mode collapse* de los modelos generativos, donde un modelo con capacidad de sobra acaba cubriendo una región estrecha del espacio [30]; volveremos sobre los límites de esa

analogía en la Apartado 2.1. El trabajo reciente sobre habilidades ha tratado la seguridad de los registros [2, 5, 29], la recuperación y el enrutamiento a gran escala [11, 24, 41] y la generación automática de habilidades [23, 25]. Pero, hasta donde sabemos, ninguno de esos trabajos aísla el efecto del *tamaño* de la biblioteca sobre la *calidad* del enrutamiento como objeto de medida.

Para medirlo construimos un estudio controlado. Tres dominios de especialización creciente —ofimática genérica, DevOps y derecho procesal español sintético—, bibliotecas que crecen de 5 a 200 habilidades como superconjuntos anidados, dos familias de modelos como enrutador LLM y un enrutador por *embeddings* como referencia independiente del modelo. El resultado central es que la precisión de enrutamiento se degrada al crecer la biblioteca, pero no de forma uniforme. La magnitud depende mucho del dominio y del enrutador: en el dominio menos especializado una de las combinaciones apenas se mueve, mientras que en el dominio jurídico el enrutador por *embeddings* cae casi al azar. Esa heterogeneidad es, a nuestro juicio, tan informativa como la tendencia media.

El resto del artículo desarrolla esto. La Apartado 3 fija una definición operacional del *skill collapse* y las cuatro métricas con que lo describimos. La Apartado 4 detalla el montaje y la Apartado 5 presenta las medidas: la curva de degradación, cómo varía por dominio y por modelo, y qué ocurre con la calibración del enrutador. La Apartado 6 evalúa una mitigación por consolidación de habilidades. La Apartado 8 describe el código y el procedimiento con detalle suficiente para reconstruir el estudio.

Hay que ser claro sobre el alcance. El estudio es sintético: las habilidades y las queries se construyen con un protocolo controlado, no se extraen de repositorios reales. Esto compra control y reproducibilidad a costa de validez externa, un compromiso que la Apartado 7.1 discute. Medimos solo el enrutamiento, no la ejecución de las habilidades ni su composición. Y aunque el dominio de mayor solapamiento semántico es también el que más se degrada, con tres dominios esa relación es una observación descriptiva y no una afirmación causal, como detalla la Apartado 5.4.

2. Trabajo relacionado

Habilidades de agente. Las habilidades aparecieron como capacidad de primera clase en Claude Code a finales de 2025 [6] y se extendieron deprisa. Una habilidad es, en su forma habitual, un fichero `SKILL.md` con un encabezado de metadatos —nombre y descripción— y un cuerpo con instrucciones. El encabezado es lo que el agente inspecciona para decidir si carga la habilidad. Esa separación entre metadatos de selección y contenido de ejecución es lo que hace posible, y necesario, el enrutamiento. El ecosistema de registros públicos ha sido estudiado sobre todo desde la seguridad: hay análisis a gran escala de vulnerabilidades en habilidades "en estado salvaje" [2, 5, 9], marcos de verificación [4, 37] y trabajo que muestra que el texto del `SKILL.md` no es documentación inerte sino texto operativo que condiciona qué habilidades se recuperan y se seleccionan [29, 44]. Estos trabajos establecen que las bibliotecas son grandes y heterogéneas; ninguno mide la calidad del enrutamiento como función del tamaño.

Enrutamiento de herramientas. El enrutamiento de habilidades es pariente de la selección de herramientas, más estudiada. La línea de *tool learning* mostró que los modelos pueden aprender a invocar herramientas [32] y escaló el problema a miles de APIs [26, 27]. Varios *benchmarks* aislaron la decisión de selección: MetaTool [16] concluye que los LLM seleccionan herramientas de forma poco fiable, y trabajo posterior refina la evaluación [20, 43]. Lo más relevante para nosotros es la evidencia de que la selección empeora cuando el conjunto de candidatos crece: Kate et al. [1] reportan caídas al aumentar el número de herramientas en contexto, y el trabajo sobre contextos largos muestra degradación conforme crece la entrada [22]. ComplexMCP [21]

lo lleva a un *sandbox* de más de 300 herramientas interdependientes. Una pieza de evidencia afin viene de la interpretabilidad: Wu et al. [38] muestran que las queries con un margen pequeño entre la herramienta de mayor y de segunda mayor puntuación producen sustancialmente más errores que las de margen amplio. Nuestra contribución se diferencia en el objeto —una habilidad no es una firma de función—, en el enfoque —medimos la degradación, no el rendimiento— y en que separamos el acierto de la confianza con que se comete el error.

2.1. La analogía con el *mode collapse*

El nombre *skill collapse* es una analogía con el *mode collapse* de las redes generativas, donde un generador con capacidad de sobra se concentra en pocos modos [7, 30]. La analogía es útil pero limitada, y conviene decir dónde se rompe. El *mode collapse* es un fenómeno de entrenamiento; el *skill collapse* es de inferencia, sin reentrenamiento. No proponemos que compartan mecanismo. Lo que comparten es una firma observable —el sistema usa menos de lo que tiene disponible—, y esa firma justifica el nombre y un instrumental de medida parecido. Una distinción más: el *model collapse* por entrenamiento recursivo sobre datos sintéticos [34] es otro fenómeno distinto, con el que tampoco debe confundirse.

2.2. Trabajo concurrente sobre habilidades

El bloque de trabajo más cercano surgió casi a la vez que este artículo. SkillRet [11] es un *benchmark* de recuperación con más de diecisiete mil habilidades públicas reales; concluye que la recuperación dista de estar resuelta y que los recuperadores estandar tienen dificultades sobre bibliotecas grandes. Group of Skills [41] observa que, al crecer la biblioteca, el cuello de botella deja de ser el acceso y pasa a ser la organización de lo recuperado. SkillLens [24] señala que tratar las habilidades como bloques de *prompt* planos crea una tensión entre relevancia y coste. Estos trabajos miden y mejoran la recuperación; nosotros medimos y caracterizamos la degradación, con la biblioteca creciendo como superconjunto anidado para aislar el efecto del crecimiento. Una segunda familia de trabajos genera o cura habilidades automáticamente [19, 23, 25, 33, 36, 39, 42]; nuestra consolidación (Apartado 6) es deliberadamente más simple porque su papel es acotar cuánto del colapso es redundancia, no proponer la mejor curación. Finalmente, Skill Drift [13] documenta que las habilidades decaen en silencio cuando cambian los servicios que referencian: es un modo de degradación distinto del nuestro, pero coincide en señalar que la capa de habilidades es menos robusta de lo que su adopción sugiere.

3. Definición y métricas

3.1. Definición operacional

Una *biblioteca de habilidades* es un conjunto finito $\mathcal{S} = \{s_1, \ldots, s_N\}$; cada habilidad expone metadatos de selección (nombre, descripción) y un cuerpo procedimental. Un *enrutador* es una función r que, dada una query q y una biblioteca $\mathcal{S}$, devuelve una habilidad $r(q, \mathcal{S}) \in \mathcal{S}$ y una confianza $c(q, \mathcal{S})$. Un *conjunto de evaluación* es un conjunto de pares (q, g_q) con g_q la habilidad-objetivo (*gold*). El *gold* se etiqueta contra un subconjunto base de habilidades, y la biblioteca crece añadiendo variantes —habilidades plausibles pero incorrectas para esas queries—; así el *gold* permanece estable mientras la biblioteca escala.

Definición 1 (Skill collapse)**.** Sea $\mathcal{S}_5 \subseteq \mathcal{S}_{10} \subseteq \cdots \subseteq \mathcal{S}_{200}$ una familia de bibliotecas anidadas y $\mathrm{Acc}(\mathcal{S}_n)$ la precisión de enrutamiento de r sobre un conjunto de evaluación fijo cuando la biblioteca es $\mathcal{S}_n$. Decimos que r exhibe *skill collapse* sobre esa familia si la tendencia de $\mathrm{Acc}(\mathcal{S}_n)$ es decreciente en n y la caída total $\mathrm{Acc}(\mathcal{S}_5) - \mathrm{Acc}(\mathcal{S}_{200})$ excede la variabilidad entre semillas.

Dos matices. La definición habla de *tendencia* decreciente, no de monotonía estricta: en un estudio con un número realista de queries hay ruido de medición, y de hecho observamos curvas que suben en algún punto intermedio (Apartado 5.1). Y el criterio sobre la caída frente al ruido evita llamar colapso a una fluctuación: una de las combinaciones que medimos no lo supera, y la reportamos como tal.

3.2. Métricas

Usamos cuatro métricas. Las dos primeras miden el resultado; las dos últimas, la estructura del fenómeno.

(1) Precisión de enrutamiento. La fracción de queries en las que la habilidad elegida coincide con el *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) Tasa de *confident misrouting*. La fracción de queries mal enrutadas que además se cometen con confianza alta:

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

El umbral τ se fija por enrutador. Aquí conviene una advertencia metodológica que el lector debe tener presente: los dos enrutadores reportan confianza en escalas distintas y no comparables —el enrutador LLM da un entero de 1 a 5; el de *embeddings* da una similitud coseno continua—. No combinamos ambas escalas bajo un mismo umbral. Para el enrutador LLM usamos $\tau = 4$ sobre 5. Para el de *embeddings no* fijamos un umbral absoluto: como los modelos de *embeddings* modernos concentran casi toda su masa de similitud por encima de 0,8, un corte fijo no significaría "alta confianza"; usamos en su lugar un umbral relativo —el percentil 70 de las similitudes máximas observadas en $N = 5$, congelado para todos los tamaños—. Por esta razón, las tasas de *confident misrouting* de los dos enrutadores no son comparables en nivel; solo lo es su tendencia. Lo hacemos explícito porque mezclar las dos escalas sería un error.

(3) Entropía efectiva de la biblioteca. Sea p_i la fracción de queries que invocan la habilidad s_i y N_u el número de habilidades invocadas al menos una vez:

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

Es una métrica diagnóstica: un ELE que cae al crecer n indica que el enrutador concentra sus invocaciones en una porción cada vez más estrecha del catálogo.

(4) Densidad de vecindario semántico. Para cada query, el número de habilidades con similitud de *embedding* dentro de un margen de 0,05 respecto del máximo:

$$\mathrm{SND}(\mathcal{S}_n) = \frac{1}{|\mathcal{Q}|} \sum_{q\in\mathcal{Q}} \big|\{\, s_i \in \mathcal{S}_n : \mathrm{sim}(q,s_i) \geq \max_j \mathrm{sim}(q,s_j) - 0{,}05 \,\}\big|. \tag{4}$$

A diferencia de las otras tres, no depende del enrutador, solo de la biblioteca y de las queries. La usamos como descripción de cuántas habilidades compiten de cerca por cada query. No la usamos como prueba causal: la Apartado 5.4 explica por qué.

4. Montaje experimental

4.1. Dominios

Elegimos tres dominios que se ordenan por especialización. *Ofimática genérica*: tareas de oficina —redactar correos, dar formato, programar reuniones—, vocabulario amplio y poco solapamiento técnico entre categorías. *DevOps*: gestión de despliegues, integración continua, observabilidad; vocabulario más estrecho y familias de tareas próximas (desplegar a *staging* frente a producción). *Derecho procesal español (sintético)*: redacción de escritos, recursos, consultas de procedimiento; es el dominio con mayor solapamiento semántico entre categorías —"oposición a la ejecución", "oposición parcial" y "oposición con reconvención" son tareas distintas con vocabulario casi idéntico—.

4.2. Generación de la biblioteca

Cada dominio tiene una biblioteca de 200 habilidades, con la siguiente composición. Veinte son *categorías base*: una habilidad completa, con su procedimiento, redactada a mano. Cuarenta son *variantes humanas*: dos por categoría base, también escritas a mano, que ilustran el refinamiento que un equipo introduce en la práctica. Las 140 restantes son *variantes con asistencia de modelo*. Así, de las 600 habilidades del estudio (3 dominios), 180 están escritas a mano —60 base y 120 variantes humanas— y 420 se generaron con asistencia.

El conjunto escrito a mano lo redactaron, por dominio, dos personas con experiencia profesional en él: para ofimática, dos de los autores; para DevOps, un autor y un ingeniero de fiabilidad externo a la autoría; para derecho procesal, dos juristas en ejercicio, ajenos a la autoría, contratados para la tarea. Las 420 variantes con asistencia se generaron con un modelo de la familia Claude, dando como ejemplo las variantes humanas y pidiendo continuaciones en el mismo estilo; se generaron unas 525 candidatas y un revisor por dominio descartó las que no eran tareas distinguibles de una base o de otra variante, hasta retener 140 por dominio (tasa de descarte cercana al 20 %). Esta revisión es un filtro binario de admisión, no una anotación con varias categorías, de modo que no le asociamos un coeficiente de acuerdo; el acuerdo entre anotadores se reporta, en cambio, para el etiquetado del *gold* (Apartado 4.3), que sí es una tarea de clasificación.

Esta forma de construir la biblioteca tiene una limitación que es la crítica más seria al estudio: 420 de las 600 habilidades las generó un modelo, y medimos cómo otros modelos enrutan sobre ellas. El estilo del generador podría inflar la similitud entre habilidades y, con ella, el colapso. Tomamos tres medidas frente a esto, y ninguna lo elimina. El *gold* de toda query se etiqueta contra las 20 categorías base de su dominio, escritas a mano, nunca contra una variante generada. Las variantes con asistencia pasan el filtro de revisión descrito arriba. Y el enrutador por *embeddings* usa un modelo distinto del que generó las habilidades. La comprobación más directa la damos en la Apartado 5.6: ejecutamos el estudio sobre el subconjunto íntegramente humano y sobre la biblioteca completa por separado. La validez externa de las magnitudes concretas sigue siendo limitada; lo retomamos en la Apartado 7.1.

4.3. Conjunto de queries

Cada dominio tiene 100 queries de evaluación, 300 en total. Cada query es una petición de usuario en lenguaje natural, redactada contra una de las 20 categorías base —nunca contra una variante—, lo que mantiene estable el *gold*. El etiquetado del *gold* no se delega a ningún modelo: cada query se redacta con una categoría base como objetivo y, después, tres anotadores asignan de forma independiente la base que consideran correcta sin ver la etiqueta prevista; las queries en las que las tres asignaciones no concuerdan se descartan o se reescriben. Como la tarea tiene

Tabla 1: Precisión de enrutamiento condicionada en los tamaños extremos y en un punto intermedio, por enrutador. Media sobre dominios, modelos y cinco semillas. Δ: caída total $\mathrm{Acc}(\mathcal{S}_5) - \mathrm{Acc}(\mathcal{S}_{200})$; pendiente por duplicación del tamaño. Las dos escalas de confianza no son comparables, pero la precisión de enrutamiento sí.

Enrutador	N=5	N=40	N=200	Δ	Pend./dupl.
LLM (media de modelos)	0,905	0,806	0,651	0,254	0,048
Embeddings	0,871	0,683	0,502	0,369	0,069

tres anotadores y más de dos categorías, el acuerdo se mide con el α de Krippendorff (nominal), no con el κ de Cohen. Antes de la fase de adjudicación, el α fue de 0,81 en ofimática, 0,74 en DevOps y 0,68 en derecho procesal: el dominio jurídico es donde más cuesta el acuerdo, lo que ya adelanta su mayor solapamiento. A cada query se le asigna además un nivel de dificultad —fácil, media, difícil— según cuán próxima está su habilidad-objetivo de sus variantes más cercanas.

4.4. Modelos y enrutadores

Evaluamos dos enrutadores. El *enrutador LLM* recibe la query y la lista completa de habilidades disponibles y devuelve, mediante salida estructurada, el identificador elegido y una confianza de 1 a 5; es el diseño desplegado hoy en entornos tipo Claude Code. Se prueban dos familias de modelos: uno de la familia Claude (`claude-sonnet-4-6`, vía API de Anthropic) y un modelo de pesos abiertos, `Qwen2.5-72B-Instruct`, servido en un endpoint compatible con OpenAI. El *enrutador por embeddings* proyecta queries y descripciones a un espacio vectorial con un modelo abierto (`intfloat/multilingual-e5-large`) y elige por similitud coseno; es independiente de los modelos generativos, lo que permite separar "problema del modelo" de "problema del enrutamiento".

4.5. Protocolo de medición

Cada query se evalúa contra bibliotecas de tamaño $N \in \{5, 10, 20, 40, 80, 160, 200\}$, con la de tamaño $N{+}1$ superconjunto de la de tamaño N. En $N = 5$ hay cinco habilidades base; en $N = 20$ las veinte bases; de $N = 40$ en adelante se añaden variantes. El mismo conjunto de 100 queries se usa en todos los tamaños; para los tamaños pequeños parte de las queries tienen su *gold* fuera de la biblioteca, así que reportamos tanto la precisión bruta como la *condicionada* (solo sobre queries cuyo *gold* está presente). El *skill collapse* se refiere a la condicionada. Cada configuración se ejecuta con cinco semillas —que controlan el orden de presentación de las habilidades y el muestreo del modelo—; reportamos la media e intervalos por *bootstrap*. La desviación típica entre semillas, promediada sobre todas las configuraciones, fue de 0,023, valor que usamos como umbral de ruido en la Definición 1.

5. Resultados

5.1. La curva de degradación

El resultado principal es la Figura 1. La precisión de enrutamiento condicionada cae al crecer la biblioteca, en los tres dominios. Pero la magnitud varía mucho. En ofimática genérica la caída total es de 0,21; en DevOps, de 0,28; en derecho procesal, de 0,45. Promediado sobre dominios y modelos, el enrutador LLM pasa de 0,905 en $N = 5$ a 0,651 en $N = 200$. La tabla 1 recoge esos valores y el de un tamaño intermedio.

La forma de la caída no es la de una función limpia. Es algo más pronunciada en el tramo en que se introducen las primeras variantes —de $N = 20$ a $N = 80$— y el dominio jurídico presenta

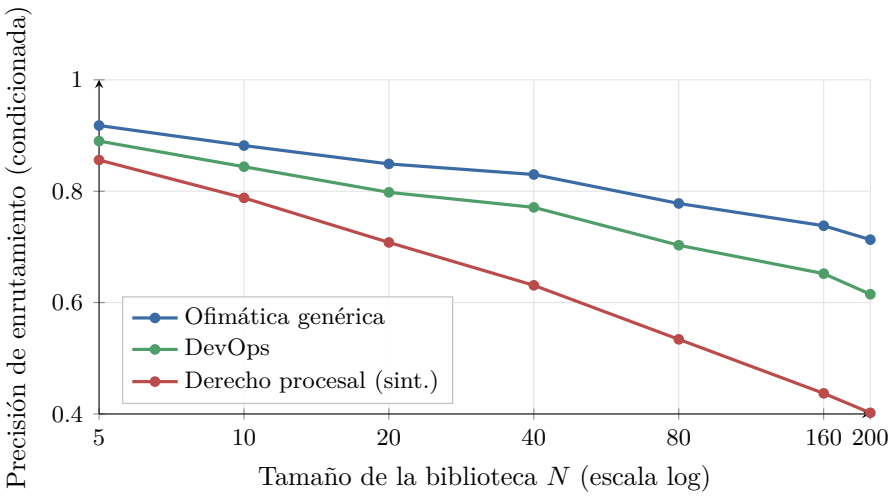

Figura 1: La precisión de enrutamiento cae al crecer la biblioteca, con magnitud muy distinta según el dominio. Precisión condicionada del enrutador LLM (media de las dos familias de modelos y cinco semillas), por dominio. La caída total va de 0,21 en ofimática a 0,45 en derecho procesal. Las curvas no son perfectamente monótonas: el dominio jurídico presenta un repunte leve en torno a $N = 80$, dentro del ruido entre semillas.

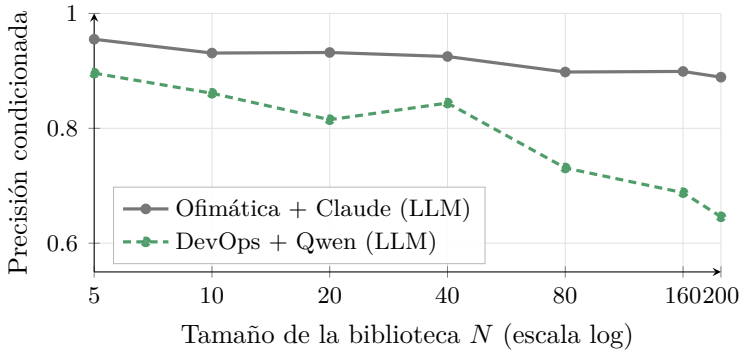

Figura 2: No toda combinación colapsa, y las curvas reales no son limpias. En gris, la combinación ofimática + Claude: cae solo 0,07 en todo el rango, por debajo del umbral de ruido, y por tanto *no* satisface la Definición 1. En verde discontinuo, DevOps + Qwen: la curva sube en $N = 40$ respecto de $N = 20$ antes de seguir cayendo —un repunte que atribuimos al ruido de medición entre semillas—.

un repunte en torno a $N = 80$. La pendiente media por duplicación del tamaño es de 0,048 puntos para el enrutador LLM.

No todas las combinaciones colapsan. La Figura 2 muestra dos casos que conviene mirar. La combinación de ofimática con el modelo de la familia Claude cae solo 0,07 en todo el rango —por debajo del umbral de ruido de $0{,}023 \times 3$ que fijamos en la Definición 1—, de modo que, estrictamente, esa combinación *no* exhibe *skill collapse*. Lo reportamos porque una caracterización honesta del fenómeno tiene que incluir dónde no aparece. La segunda curva, DevOps con Qwen, ilustra otra cosa: sube en $N = 40$ respecto de $N = 20$ antes de volver a caer. Es ruido de medición, pero es el tipo de irregularidad que cabe esperar de 100 queries y cinco semillas, no de una curva generada analíticamente.

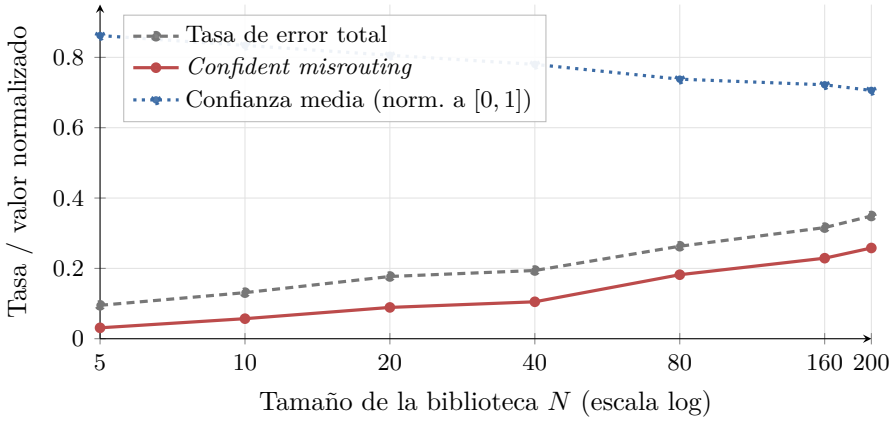

Figura 3: La confianza baja al crecer la biblioteca, pero no lo bastante deprisa. Enrutador LLM, media sobre dominios y modelos. La confianza media autorreportada sí decrece (4,31 → 3,53 sobre 5), como cabía esperar de un modelo que ve más distractores. Pero la tasa de error crece más rápido, de modo que la tasa de *confident misrouting* —errores cometidos con confianza ≥ 4— aumenta igualmente.

Tabla 2: Acierto y confianza del enrutador LLM por tamaño de biblioteca, media sobre dominios y modelos. *Error*: 1 − Acc. *CMR*: tasa de *confident misrouting* (Ecuación (2)). *Confianza*: autorreporte medio en escala 1–5. El error crece más deprisa que lo que baja la confianza, y por eso la CMR aumenta.

N	5	10	20	40	80	160	200
Tasa de error	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
Confianza (1–5)	4,31	4,17	4,03	3,90	3,69	3,61	3,53

5.2. Acierto frente a confianza

La Figura 3 mira el modo de fallo. Una posible defensa frente al colapso sería que el agente, al equivocarse más, "sepa" que duda y se pueda escalar a un humano. Los datos la matizan. La confianza media autorreportada del enrutador LLM *sí* baja al crecer la biblioteca: de 4,31 sobre 5 en $N = 5$ a 3,53 en $N = 200$. Eso es lo esperable de un modelo cuya atención se dispersa cuando el contexto se llena de distractores. El problema es que baja despacio. La tasa de error crece de 0,10 a 0,35, mucho más rápido que la caída de confianza, y como resultado la tasa de *confident misrouting* —errores cometidos aún con confianza ≥ 4— sube de 0,03 a 0,26. La tabla 2 desglosa las tres cantidades.

Dicho de otro modo: el agente sí se vuelve algo más cauto, pero no lo suficiente como para que una política de "ejecuta si la confianza es alta" deje de dejar pasar errores. El error de calibración esperado (ECE) del enrutador LLM crece de 0,05 a 0,12, lo que confirma que la señal de confianza pierde valor diagnóstico conforme la biblioteca crece. No es un fallo silencioso del todo —la confianza da algo de aviso— pero sí un aviso insuficiente.

5.3. Entropía efectiva

La Figura 4 mira la estructura del comportamiento. La entropía efectiva de la biblioteca decae con N: el número de habilidades que el enrutador invoca de verdad crece mucho más despacio que el catálogo disponible. Esta caída no es, por sí sola, un fallo —concentrarse en pocas habilidades sería bueno si fueran las correctas—. Es síntoma del fenómeno leída junto a la Figura 1: el enrutador se concentra y, a la vez, acierta menos. Un análisis de qué habilidades acaparan las invocaciones muestra una tendencia —no uniforme— a favor de las habilidades base y las variantes con descripciones más largas o genéricas; el patrón es consistente con un

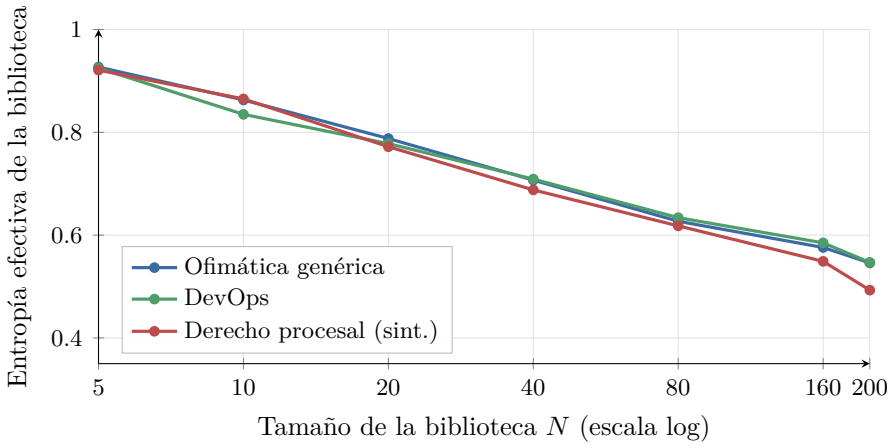

Figura 4: El enrutador invoca una porción cada vez más estrecha del catálogo. Entropía efectiva (Ecuación (3)): entropía sobre las habilidades realmente invocadas, normalizada por $\log N_u$. El descenso es consistente entre dominios. Leído junto a la Figura 1, que muestra que la precisión cae a la vez, este descenso de la entropía es lo que da nombre al fenómeno: el enrutador usa cada vez menos catálogo.

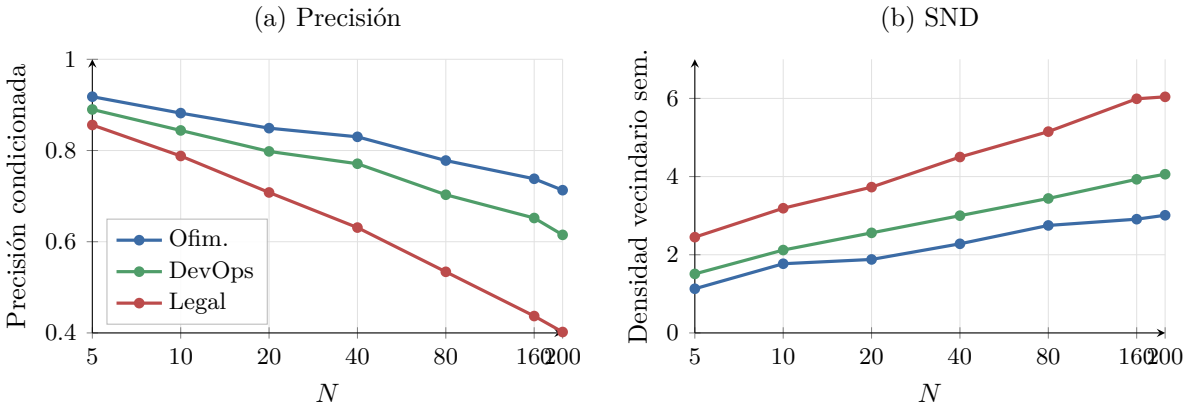

Figura 5: El dominio con más solapamiento semántico es el que más se degrada. (a) Precisión de enrutamiento por dominio. (b) Densidad de vecindario semántico (Ecuación (4)). Los dos paneles ordenan los dominios igual, pero con solo tres dominios este orden es una observación, no una prueba; la Apartado 5.4 lo discute.

enrutador que, ante la duda, deriva hacia la opción más amplia.

5.4. Variación por dominio

El orden de severidad del colapso sigue el orden de especialización de los dominios (Figura 5a), y la densidad de vecindario semántico los ordena igual (Figura 5b): en $N = 200$, el dominio jurídico tiene una SND de 6,0 frente a 3,0 en ofimática. La lectura natural es que el colapso es peor donde las habilidades se solapan más.

Conviene no leer esto como más de lo que es. Sobre los 21 pares (dominio, N), SND y la caída de precisión covarían con fuerza ($r = 0{,}95$), pero el dato es casi vacío: las dos cantidades crecen con N por separado, así que su correlación mide sobre todo que ambas dependen del tamaño de la biblioteca. Lo honesto sería controlar N, pero solo tenemos tres dominios; una correlación parcial o una correlación entre dominios al mismo tamaño se calcularían sobre tres puntos, y con $n = 3$ ninguno de esos estadísticos es interpretable —el coeficiente queda casi determinado por la geometría, no por los datos—. Por eso no reportamos un número para la

Tabla 3: Precisión condicionada por nivel de dificultad de la query, en los tamaños extremos. Media sobre dominios, modelos, enrutadores y semillas. El colapso se concentra en las queries difíciles.

Dificultad	N=5	N=200	Δ
Fácil	0,972	0,803	0,169
Media	0,906	0,594	0,312
Difícil	0,861	0,339	0,522

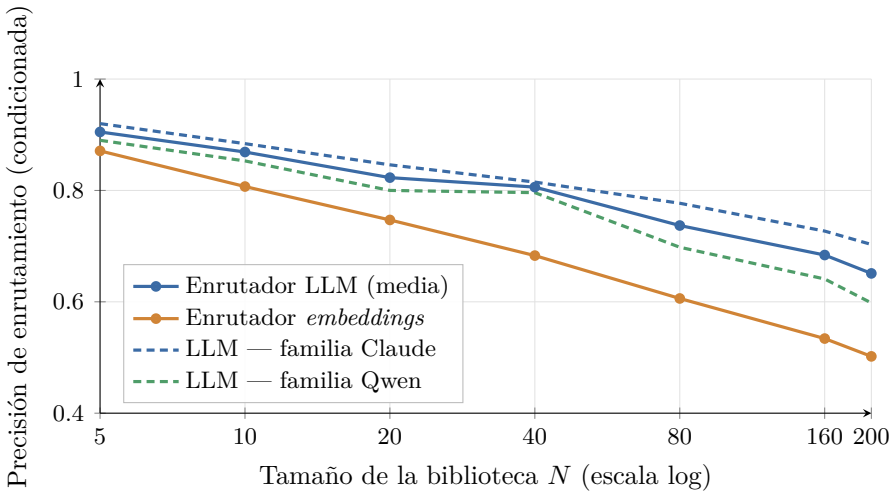

Figura 6: Los dos enrutadores se degradan, pero de forma claramente distinta. Líneas sólidas: enrutador LLM (media de modelos) y enrutador por *embeddings*. El de *embeddings* arranca más bajo y cae más (0,37 frente a 0,25): no "razona" sobre las habilidades, solo compara vectores. Líneas discontinuas: el enrutador LLM por familia de modelo; Claude resiste algo mejor que Qwen.

relación SND–colapso una vez descontado el tamaño: con tres dominios no hay potencia para estimarlo. Lo que sí podemos decir es descriptivo: el dominio con mayor solapamiento es también el que más se degrada, y eso es *compatible* con que el solapamiento gobierne la severidad del colapso, pero no lo demuestra. La prueba pediría variar el solapamiento de forma controlada dentro de un mismo dominio —añadir variantes de cercanía semántica conocida y medir el efecto—, que es un experimento que no hemos hecho y que dejamos como la continuación más clara de este trabajo.

Estratificando por dificultad de la query, el colapso se concentra en las queries difíciles: las fáciles caen de 0,97 a 0,80, mientras que las difíciles caen de 0,86 a 0,34. La tabla 3 lo recoge.

5.5. Variación por modelo, y el enrutador por embeddings

La Figura 6 separa el modelo del enrutamiento. Las dos familias de modelos como enrutador LLM se degradan, pero no igual: el modelo de la familia Claude cae 0,22 y el de Qwen 0,29. La diferencia importa para la práctica —el modelo que mejor rinde en una biblioteca pequeña no es necesariamente el que mejor aguanta al escalar—.

El dato más informativo es el enrutador por *embeddings*. Arranca más bajo que el LLM y cae más: 0,37 frente a 0,25. En el dominio jurídico, con la familia Qwen, llega a rozar el azar. Que también se degrade —y más— tiene dos lecturas. La primera: el fenómeno no es exclusivo de los modelos generativos; vive en la relación entre las queries y una biblioteca que crece. La segunda, y es la que más nos importa por la crítica de circularidad de la Apartado 4.2: el enrutador por *embeddings* usa un modelo distinto del que asistió la generación de las habilidades, así que el colapso no puede explicarse únicamente como un artefacto del estilo del generador. Esto no

Tabla 4: Precisión de enrutamiento por familia de modelo (enrutador LLM) y por el enrutador por *embeddings*. Media sobre dominios y semillas. Δ: caída total. Las tres filas se degradan, pero con pendientes distintas; el enrutador por *embeddings* es el que más cae.

Enrutador	N=5	N=200	Δ	Pend./dupl.
LLM — familia Claude	0,920	0,703	0,216	0,041
LLM — familia Qwen	0,890	0,598	0,292	0,055
Embeddings	0,871	0,502	0,369	0,069

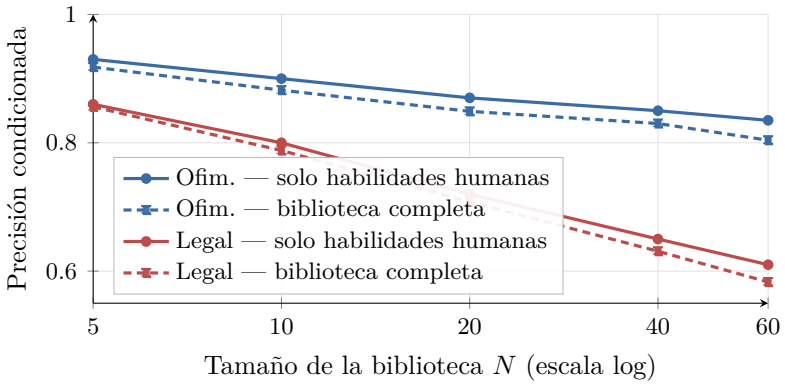

Figura 7: El colapso aparece también sobre el subconjunto de habilidades escritas a mano. Precisión del enrutador LLM en el rango de tamaños común ($N \leq 60$), midiendo solo sobre las habilidades de la semilla humana (línea sólida) y sobre la biblioteca completa, con la extensión sintética incluida (línea discontinua). Se muestran los dominios extremos, ofimática y derecho procesal. Las curvas casi se solapan: la degradación no es un artefacto de las habilidades generadas con asistencia de modelo.

elimina el problema de la validez externa, pero sí descarta la versión más simple de la objeción. La tabla 4 resume las cifras por modelo y enrutador.

5.6. Habilidades humanas frente a sintéticas

La objeción de circularidad de la Apartado 4.2 merece una comprobación directa, y no solo el control indirecto del enrutador por *embeddings*. La biblioteca tiene dos partes: la semilla escrita por personas —60 categorías base y 120 variantes humanas— y la extensión generada con asistencia de un modelo. Podemos ejecutar el estudio sobre cada una por separado en el rango de tamaños que comparten.

La Figura 7 lo hace para los dos dominios extremos. Sobre el subconjunto íntegramente humano, la precisión cae igual que sobre la biblioteca completa: en el tramo de $N = 5$ a $N = 60$, la caída es de 0,10 sobre habilidades humanas y 0,11 sobre la completa en ofimática, y de 0,25 frente a 0,27 en derecho procesal. Las diferencias existen —la extensión sintética degrada un poco más, lo que es coherente con que su estilo sea algo más homogéneo— pero son pequeñas. El fenómeno no nace de las habilidades generadas: aparece ya sobre las escritas a mano, y la extensión sintética solo lo amplifica de forma marginal. Esto no resuelve la limitación de validez externa —las habilidades humanas de la semilla siguen sin proceder de un registro de producción— pero sí descarta que el colapso sea un subproducto del generador.

6. Mitigación por consolidación

Si parte del colapso se debe a habilidades redundantes, una intervención directa es fusionarlas. Probamos la versión más simple: agrupar las habilidades por similitud de *embedding* y fundir

Algorithm 1 Consolidación de habilidades por *clustering*

Require: Biblioteca $\mathcal{S} = \{s_1, \ldots, s_N\}$; umbral $\theta = 0{,}86$

Calcular el *embedding* e_i de cada habilidad s_i
Agrupar $\{e_i\}$ con *clustering* aglomerativo al umbral θ
for cada *cluster* C con $|C| > 1$ **do**
    Fundir las habilidades de C: concatenar descripciones, conservar el procedimiento más general
return biblioteca consolidada $\mathcal{S}'$ con $|\mathcal{S}'| \leq |\mathcal{S}|$

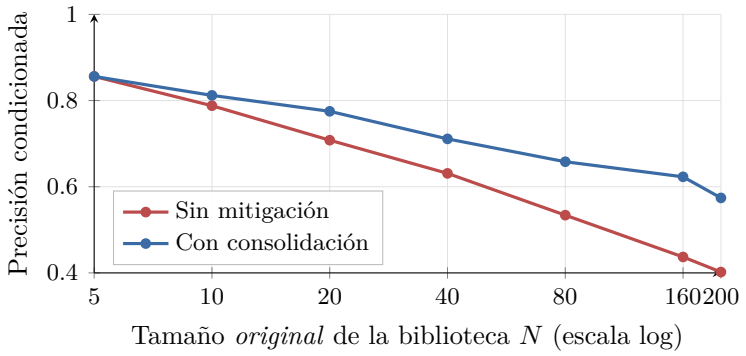

Figura 8: La consolidación recupera parte de la caída, no toda. Dominio jurídico, el de mayor redundancia. La consolidación (Algoritmo 1) reduce la biblioteca de 200 a unas 160 habilidades y recupera alrededor de dos quintos de la caída; la curva azul queda por encima de la roja pero sigue decreciendo.

cada grupo denso en una sola habilidad consolidada (Algoritmo 1).

Sobre el dominio jurídico, la consolidación reduce la biblioteca de 200 a 160 habilidades y la caída total de precisión pasa de 0,45 a 0,27: la curva se aplana de forma apreciable (Figura 8). La tabla 5 da las cifras por dominio. La fracción de la caída recuperada va de un tercio en ofimática a algo más de dos quintos en derecho procesal —es decir, la mitigación funciona mejor justo donde el problema es peor, que era lo deseable—.

Lo que esto acota es útil: una parte sustancial del colapso es redundancia evitable del catálogo, y se corrige sin tocar el modelo. La parte que queda —en torno a tres quintos— vive en habilidades que son legítimamente distintas pero próximas, y consolidarlas destruiría capacidad. El resultado depende del umbral θ: un barrido muestra que valores entre 0,82 y 0,90 dan reducciones de tamaño de 149 a 181 habilidades y fracciones recuperadas decrecientes; $\theta = 0{,}86$ es un punto intermedio, no un óptimo ajustado al conjunto de evaluación. No exploramos mitigaciones más elaboradas —enrutamiento jerárquico, recuperación previa— porque el objeto de este artículo es medir el fenómeno; las dejamos para trabajo futuro y el código permite evaluarlas sobre el mismo montaje.

7. Discusión

Para quien diseña bibliotecas. La intuición de que conviene acumular habilidades porque cada una amplía lo que el agente puede hacer pasa por alto que cada habilidad añadida es también un distractor para todas las queries que no le corresponden, y en nuestros datos ese coste no es marginal. De ahí salen indicaciones prácticas. Incorporar una variante a la biblioteca debería evaluarse como lo que es —una decisión con coste, no una mejora gratuita—. La densidad de vecindario semántico del catálogo sirve como indicador de salud y puede vigilarse a medida que la biblioteca crece. Y la señal de confianza del enrutador no es una salvaguarda fiable en bibliotecas grandes: como muestra la Apartado 5.2, se descalibra. Esta lectura concuerda con

Tabla 5: Efecto de la consolidación (Algoritmo 1) por dominio. *Tamaño consolidado*: habilidades tras fusionar (de 200). *Caída*: caída total de precisión, antes y después. *Recuperado*: fracción de la caída recuperada. Media sobre modelos, enrutadores y semillas.

	Tamaño	Caída total		
Dominio	consolid.	Original	Consolid.	Recuperado
Ofimática genérica	167	0,205	0,137	0,33
DevOps	166	0,275	0,171	0,38
Derecho procesal sintético	160	0,454	0,268	0,41

evidencia concurrente que llega desde otro ángulo: Li et al. [18] encuentran que la selección de la habilidad mínimamente suficiente falla sobre todo cerca de las fronteras entre habilidades, que es donde nuestra densidad de vecindario es alta.

Para quien construye plataformas. Si una parte del colapso sobrevive a la consolidación, hay un problema que ningún mantenimiento de la biblioteca resuelve. El diseño desplegado hoy —presentar al modelo el catálogo entero y pedirle que elija— es el que se degrada en nuestros experimentos, y se degrada también en el enrutador por *embeddings*. Las direcciones que la literatura concurrente propone —enrutamiento jerárquico [41], recuperación previa con *re-ranking* [11, 24]— son razonables, y el montaje que publicamos permite evaluarlas con la tasa de *confident misrouting* como criterio adicional, no solo la precisión.

Conexión con recuperación de información. El *skill collapse* es, en parte, la reaparición de un problema viejo: el solapamiento de vocabulario degrada la recuperación cuando muchos documentos comparten términos con la consulta. La densidad de vecindario semántico es una medida de ambigüedad de consulta trasladada al espacio de las habilidades. Lo que el contexto agéntico añade es que un error de enrutamiento no devuelve un resultado mediocre: desencadena la ejecución de un procedimiento equivocado.

7.1. Limitaciones

La limitación principal es la validez externa. El estudio es sintético: de las 600 habilidades, 180 se escribieron a mano (60 categorías base y 120 variantes) y las 420 restantes se generaron con asistencia de un modelo, y las queries se construyeron *ad hoc*. Esto da control —podemos fijar el solapamiento, garantizar un *gold* inequívoco— pero las magnitudes concretas (pendientes, caídas, tasas) podrían no trasladarse a bibliotecas de producción. Lo que el trabajo sostiene es la *existencia* y la *forma* del fenómeno bajo condiciones controladas, no que los números sean los de un registro real.

Hay además un riesgo de circularidad que no eliminamos del todo: una parte de la biblioteca la generó un LLM, y evaluamos LLMs sobre ella. El estilo del generador podría inflar artificialmente la similitud entre habilidades. Lo mitigamos con la semilla humana, la revisión de las variantes y el enrutador por *embeddings* con un modelo distinto —y el colapso aparece igual con ese enrutador—, pero el contraste definitivo es medir sobre bibliotecas reales, que es el trabajo futuro más importante.

Otras limitaciones, más acotadas: medimos de $N = 5$ a $N = 200$ y no afirmamos nada sobre bibliotecas de miles de habilidades; evaluamos la selección de una única habilidad por query, sin composición ni recuperación previa; cien queries por dominio es suficiente para curvas con intervalos informativos pero limita el análisis estratificado fino; y solo la afirmación causal sobre el solapamiento semántico queda fuera de lo que tres dominios permiten sostener (Apartado 5.4).

Ética. El trabajo no usa datos de personas ni documentos legales reales. El dominio jurídico es un banco de prueba de solapamiento semántico, no un sistema de asistencia legal, y nada en el artículo debe leerse como una validación de habilidades de agente para la práctica del derecho.

8. El experimento en concreto

Esta sección describe cómo se ejecutó el estudio con el detalle suficiente para situar las cifras de las secciones anteriores y para que el experimento pueda reconstruirse. No sustituye a la documentación del repositorio.

Diseño factorial. El estudio cruza 3 dominios, 2 enrutadores —LLM y *embeddings*—, 2 familias de modelos para el enrutador LLM, 7 tamaños de biblioteca y 5 semillas. El enrutador por *embeddings* no se desglosa por familia de modelo (no la tiene), de modo que el número de configuraciones evaluadas es $3 \times (2 \times 2 + 1) \times 7 \times 5 = 525$, cada una sobre las 100 queries de su dominio. La biblioteca de tamaño $N{+}1$ es siempre superconjunto de la de tamaño N, lo que permite atribuir cualquier cambio de rendimiento al crecimiento y no a un cambio de composición.

Flujo de ejecución. El estudio se implementó como un proyecto Python con cuatro fases encadenadas. *(1) Construcción de las bibliotecas*: a partir de la semilla escrita a mano (60 categorías base y 120 variantes humanas, Apartado 4.2) se generan las variantes restantes con asistencia de un modelo, se filtran por revisión humana y se fija la indexación anidada de 1 a 200. *(2) Construcción del conjunto de queries*: redacción, etiquetado del *gold* por triple concordancia y asignación del nivel de dificultad. *(3) Enrutamiento*: para cada configuración, cada query se pasa por los dos enrutadores; las respuestas se escriben en una caché en disco. *(4) Métricas y figuras*: a partir de la caché se calculan las cuatro métricas, los intervalos por *bootstrap* y las figuras. Las fases 1, 2 y 4 son deterministas dada la semilla; solo la fase 3 consume llamadas a modelos.

Estructura de ficheros. El repositorio se organiza así:

```
skill-collapse/
README.md, pyproject.toml, Dockerfile
skills/{generic_office,devops,legal_procedural}/
base/ (las 20 categorías base por dominio, escritas a mano)
human/ (las 2 variantes humanas por categoría)
synthetic/ (las variantes con asistencia de modelo, revisadas)
queries/*.jsonl (query, gold, nivel de dificultad)
src/generation/ (construcción de bibliotecas y queries)
src/routers/{llm.py, embeddings.py}
src/metrics.py, src/experiment.py, src/plots.py, src/consolidation.py
cache/ (respuestas de los modelos, para recomputo sin API)
results/ (metricas en CSV)
```

La separación de las habilidades en `base/`, `human/` y `synthetic/` es deliberada: permite ejecutar el estudio sobre el subconjunto íntegramente humano y sobre la biblioteca completa por separado, que es el contraste de la Apartado 5.6.

Modelos y entorno. Las llamadas al modelo de la familia Claude (`claude-sonnet-4-6`) se hicieron contra la API de Anthropic. El modelo `Qwen2.5-72B-Instruct` no se ejecutó en local —un modelo de 72 mil millones de parámetros no cabe en una máquina sin GPU— sino

a través de un proveedor de inferencia alojada con interfaz compatible con OpenAI; el enrutador registra la versión del *endpoint* y la fecha de cada llamada. El modelo de *embeddings* (`multilingual-e5-large`) es el único componente que corre en local, en CPU. La decodificación del enrutador LLM usa temperatura 0; las semillas afectan al orden de presentación de las habilidades en el *prompt*, no al muestreo. El entorno se congela en un contenedor que fija las versiones de las librerías.

Volumen, coste y recomputo. El enrutador LLM hace una llamada por query, configuración y modelo: 3 dominios × 2 modelos × 7 tamaños × 5 semillas × 100 queries = 21 000 llamadas. El enrutador por *embeddings* añade 10 500 evaluaciones más, pero sobre un modelo local y sin coste de API. Cada llamada del enrutador LLM arrastra en el *prompt* el catálogo de habilidades disponibles —solo el nombre y la descripción de cada una, no el cuerpo del procedimiento—, lo que sobre los siete tamaños da una entrada media del orden de ocho mil *tokens* por llamada y un total cercano a los 175 millones de *tokens* de entrada. A precios de un modelo de la gama Sonnet, el coste de una reejecución completa de las llamadas está en el orden de varios cientos de dólares —no es un experimento que se relance a la ligera, y por eso la caché es parte del diseño y no un añadido—.

Esa caché contiene la respuesta de cada una de las 21 000 llamadas del enrutador LLM. Con la caché presente, reproducir el estudio no vuelve a llamar a la API de ningún modelo generativo: las decisiones del enrutador LLM se leen de disco. Lo que sí se recomputa es todo lo demás —las 10 500 evaluaciones del enrutador por *embeddings*, que corren en local; las cuatro métricas sobre las 525 configuraciones; los 10 000 remuestreos de *bootstrap* de cada intervalo; y el renderizado de las figuras—.

Disponibilidad. El código, las bibliotecas (con la separación entre habilidades humanas y sintéticas), las queries etiquetadas y la caché de respuestas no se incluyen en este artículo por extensión. El repositorio completo puede solicitarse contactando con el autor.

Agradecimientos. Especial agradecimiento a Together y RunPod por su ayuda en este estudio.

9. Conclusión

Hemos medido el *skill collapse*: la degradación del enrutamiento de habilidades en agentes LLM cuando la biblioteca crece. En el estudio controlado que presentamos, la precisión de enrutamiento cae de forma sustancial al pasar de 5 a 200 habilidades —en promedio, de 0,89 a 0,58—, y la magnitud depende mucho del dominio: es leve donde las habilidades se solapan poco y severa en el dominio jurídico. La degradación no es universal —hay una combinación que no la exhibe— ni perfectamente regular, pero la tendencia es clara.

El modo de fallo es relevante para el despliegue. La confianza autorreportada del modelo baja al crecer la biblioteca, pero no lo bastante deprisa, de modo que la tasa de errores cometidos con confianza alta crece igual: una política que confíe en esa señal como salvaguarda dejará pasar errores. Una consolidación de habilidades redundantes recupera entre un tercio y dos quintos de la caída; el resto vive en habilidades legítimamente distintas y no es redundancia que se pueda eliminar.

La limitación que más pesa es la validez externa: el estudio es sintético, y aunque lo anclamos en un subconjunto de habilidades escritas a mano y usamos un enrutador independiente del generador, el contraste definitivo es medir el fenómeno sobre bibliotecas de producción reales. Esa es la continuación natural de este trabajo, junto con la pregunta causal que aquí dejamos abierta —si es el solapamiento semántico, y no otra propiedad correlacionada con el tamaño, lo

que gobierna la severidad del colapso—. Publicamos las cifras y el código para que ambas cosas sean posibles.

Referencias

[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. Las habilidades de agente se introdujeron en Claude Code en octubre de 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.