

La on-policy distillation solo ve *un token cada vez*.

Primer número semanal. La conclusión del mes pasado era que la on-policy distillation había desplazado al pruning como forma de hacer bueno un modelo pequeño; esta semana cuatro papers independientes han aterrizado sobre la misma limitación, y han tirado del mismo tipo de solución. Además, speculative decoding llega a llama.cpp con unas cifras lo bastante honestas como para enseñar cuándo no ayuda, y un resultado de cuantización que debería preocupar a cualquiera que tenga 4 bits corriendo dentro de un agente.

COMPILADO 23 AGO 2026

4 SECCIONES

~40 CITAS PRIMARIAS

VENTANA 7 DÍAS

DE UN VISTAZO

01

Cuatro papers, un mismo diagnóstico: la señal de la on-policy distillation es por token y, por tanto, demasiado local. Un profesor que puntúa token a token puede parchear un paso malo sin llegar a mostrar nunca el camino correcto completo. Todas las soluciones de esta semana añaden encima un término de horizonte más largo: supervisión a nivel de paso 2608.16333, un residuo de desacuerdo con el verificador 2608.19181 y un reparto distinto del presupuesto de tokens entre profesores 2608.19098. El cuarto explica *por qué* generaliza siquiera — y por qué apilar profesores sale mal 2608.16647.

Llegó speculative decoding para LFM2 en llama.cpp, y las cifras publicadas enseñan exactamente para qué sirve. Se han integrado drafters DSpark para LFM2 #27383 y para bailingmoe3 #27508. La tabla del propio autor promedia $2,12\times$ en una RTX 4070 Laptop — pero eso es $3,22\times$ en un prompt de merge de código con un 66% de aceptación del draft, frente a $1,33\times$ en uno explicativo con un 18%. Speculative decoding es una optimización para salida estructurada, y la media lo esconde.

La cuantización de 4 bits degrada un modo de fallo que ningún benchmark mide. Con interferencia proactiva alta — un valor sobrescrito repetidamente en el contexto — INT4 con bitsandbytes baja Qwen2.5-7B-Instruct del 81,0% al 68,3%, y los errores de intrusión de la misma clave suben del 21,5% al 24,6% de los intentos. INT8, que se suele dar por gratis, tiene una penalización menor pero real en dos de los tres modelos probados. La precisión agregada en benchmarks parece prácticamente intacta todo el rato 2608.18578.

Ornith-1.5 salió como pesos abiertos con licencia MIT, en un MoE 35B-A3B y un 9B denso. Tres mil millones de parámetros activos, 262K de ventana de contexto nativa y un 79 declarado en SWE-bench Verified para el MoE (model card); las conversiones a GGUF aparecieron el mismo día.

CONTENIDO

§1 llama.cpp — decoding, Metal y una saga de reverts

§2 Los papers — el problema de la señal local

§3 Pesos que se publicaron

§4 Qué haría yo

El glosario es ahora una

página permanente

en vez de un apéndice de cada número — cambia poco de una semana a otra, y rehacerlo cada lunes era la forma equivocada. Para este número se han añadido tres términos.

Decoding, Metal y una saga de reverts

135 pull requests integrados entre el 17 y el 23 de agosto, terminando en la build `b10590`. Agrupados por lo que cambian para quien de verdad ejecuta esto.

Llega speculative decoding a los modelos recurrentes

Se integraron drafters DSpark para LFM2 #27383, que además añade rollback parcial para el estado recurrente de LFM2 — la pieza que hace posible especular sobre un modelo que no es puramente atencional — y para bailingmoe3 al día siguiente #27508. Los GGUF de los drafters están publicados, así que esto se puede usar ya, no cuando alguien haga la conversión.

La tabla que publica el autor es inusualmente honesta, y merece leerse como la lección general sobre speculative decoding más que como un dato sobre LFM2. En una RTX 4070 Laptop, con LFM2.5-1.2B-Instruct F16 contra su drafter DSpark de 5 capas:

PROMPT	SPEEDUP	ACEPTACIÓN DEL DRAFT
merge (código)	3,22×	66%
corrección	2,98×	64%
poema	1,54×	23%
resumen	1,53×	24%
explicación	1,33×	18%
media	2,12×	—

Una horquilla de 2,4× entre tipos de prompt, siguiendo casi exactamente a la tasa de aceptación. Si tu carga son ediciones de código y reescritura estructurada, especular es throughput casi gratis. Si es prosa explicativa, estás alojando un segundo modelo a cambio de aproximadamente un tercio más de velocidad. Son cifras del autor en una sola máquina, así que trata las magnitudes como afirmaciones — pero la *forma* es el comportamiento bien establecido de la técnica.

Apple Silicon: KV cuantizada a través de flash attention

Metal ganó un camino para deshacer la cuantización de las KV cache cuantizadas (Q4_0, Q4_1, Q5_0, Q5_1, Q8_0) a F16 contiguo y pasar por encima los kernels de flash attention F16 que ya existían #27390. En modelos MLA, donde V es una vista de K, se salta la dequantización redundante de V. Dos días después ese mismo camino se restringió solo a batches grandes #27438 — la dequantización no es gratis, y con batches pequeños costaba más de lo que devolvía. Publicar y luego medir, visible en el log de commits. Aparte, la dequantización de Q8_0 en Metal pasó a usar tipos empaquetados #27370.

Cuantizar un modelo grande en una máquina que no debería poder

`llama-quantize` ya puede liberar las páginas de memoria de un tensor en cuanto deja de necesitarlo, en vez de mantener el modelo entero residente durante toda la ejecución #22877. Es el tipo de cambio sin glamour que decide si puedes cuantizar un modelo grande con el hardware que tienes o tienes que alquilar algo una tarde.

Server: tus rutas de modelo eran públicas

Con autenticación configurada, `/v1/models` ya no responde a clientes no autenticados #26347. El endpoint devolvía rutas del sistema de ficheros a los modelos — y una ruta como `/home/alex/Downloads/...gguf` filtra un nombre de usuario a cualquiera que alcance el puerto. Si ejecutas `llama-server` con auth activada y delante tienes algo que no sea una red plenamente de confianza, este es el único arreglo de seguridad de verdad de la semana.

Más cosas del server: un preset `dedup-cache-models` #27346 y `/metrics` ahora es accesible mientras el server duerme #27376, lo cual importa si estabas haciendo scraping de una instancia dormida y leías el timeout como una caída.

Arquitecturas nuevas

Entró el soporte de BailingMoE3 #26608, que habilita Ling 3.0 flash con MTP, junto con las variantes de sliding window attention de Granite, `GraniteSWAForCausalLM` y `GraniteMoeSWAForCausalLM` #25505.

Los kernels reordenados Q2_K MMVQ y ESIMD para SYCL se integraron el 21 de agosto, declarando una mejora de generación de tokens de $1,54\times$ a $1,90\times$ según el hardware, medida sobre Qwen35-27B-Q2_K_M #26336. Se revirtieron horas después para dejar el CI en verde #27486, y volvieron a entrar al día siguiente con los parámetros de gate que un conflicto de merge había dejado por el camino #27490.

Los kernels están dentro y el speedup es real, pero si compilas SYCL desde fuente, fija una versión posterior a #27490 — la ventana entre los tres contiene una build que o está rota o le falta la optimización en silencio. Esto es también el argumento de siempre para comprobar el estado de merge en vez de fiarse de un resumen: leída en el momento equivocado, esta historia tiene tres finales distintos.

El problema de la señal local de la destilación

El número 01 cerraba señalando la on-policy distillation como lo que había sustituido al "poda el grande". Cuatro papers enviados dentro de esta misma semana se han puesto a buscarle los bordes, y convergen hasta un punto poco habitual para siete días de arXiv.

El mismo diagnóstico, cuatro veces

La OPD entrena al alumno con sus propias trayectorias generadas, y el profesor puntúa los tokens que el alumno ha producido de verdad. La queja, en cuatro vocabularios distintos, es que esa señal es *por token*, y una señal por token es local: puede marcar un paso concreto como erróneo sin llegar a demostrar nunca el camino correcto completo desde ahí.

Step-Level OPD lo dice de la forma más directa — la OPD a nivel de token "solo puede aportar correcciones fragmentadas a lo largo de una trayectoria errónea del alumno y no puede desplegar un camino de reparación completo y correcto" — e interpola entre OPD y fine-tuning supervisado supervisando trayectorias completas del alumno con granularidad de paso. En los límites de la longitud de paso se reduce a SFT o se aproxima a OPD. Declara 13,4 puntos de tasa de éxito media por encima de la OPD estándar en ALFWorld 2608.16333.

GC-OPD ataca la misma localidad desde el lado del verificador: en tareas de contexto largo, el apoyo del profesor token a token favorece respuestas que son plausibles en local pero se dejan evidencia repartida por

la entrada. El paper primero mide el desacuerdo — las puntuaciones de OPD a nivel de trayectoria se desalinean progresivamente de las recompensas del verificador a medida que crece la entrada — y luego usa el residuo con signo entre ambas como señal de entrenamiento adicional. Post-entrenar los checkpoints oficiales de Qwen3-4B y Qwen3-8B sube sus medias en cinco benchmarks de 29,08 a 40,47 y de 35,12 a 44,65; la OPD estándar llega a 39,31 y 43,56 con el mismo montaje [2608.19181](#).

Open-MOPD encuentra el problema de la localidad en el caso multi-profesor y, muy útil, descarta la explicación obvia. La OPD multi-profesor estándar captura solo el 35,6% del margen disponible frente a un ensemble oráculo con enrutado por dominio sobre SmolLM3-3B-Base — y la causa *no* es conflicto de gradientes entre profesores, sino un mal reparto del presupuesto de optimización por token, provocado por diferencias estructurales de longitud de secuencia entre dominios, deriva en la convergencia y recompensas obsoletas. Reequilibrar ese presupuesto sube la recuperación del margen al 83,4% [2608.19098](#).

El cuarto es el que hay que leer si solo lees uno. Un estudio controlado que varía un factor de generalización cada vez encuentra que **la OPD transfiere el comportamiento de razonamiento del profesor, no sus respuestas**: la dificultad de entrenamiento apenas importa, y los problemas que el profesor nunca resuelve siguen siendo señal útil. Lo que decide la transferencia es la relación de origen — parejas profesor/alumno del mismo origen transfieren entre idiomas, horizontes de razonamiento y dominios, mientras que las de origen distinto se limitan mayormente a ajustarse a la distribución entrenada. Y ese alcance es justo el problema en los montajes multi-profesor: como enrutar prompts a expertos de dominio no consigue confinar la influencia de cada profesor, combinarlos produce un "balancín dependiente de la mezcla" entre sus capacidades [2608.16647](#).

Leídos juntos: la OPD funciona porque es amplia, y los fallos multi-profesor que los otros tres papers están parcheando son esa misma amplitud llegando como efecto secundario. Es un hallazgo real para una semana, y también es el límite de lo que una semana establece — cuatro papers de acuerdo son una hipótesis con un respaldo inusualmente bueno, no un resultado asentado. Fuera de ventana pero parte de la misma conversación, SimpleOPD aborda el caso del desajuste de tokenizador y declara a Intern-S2-Preview subiendo 21,2 puntos en ProofBench hasta 55,2 ([2608.14277](#), 14 de agosto, fuera de ventana).

EL PANEL DE ADVERTENCIA – LA COMPRESIÓN, OTRA VEZ, NO ES NEUTRA EN COMPORTAMIENTO

La sección más útil del número 01 fueron cuatro papers que encontraban modelos comprimidos pasando todos los controles de calidad estándar mientras se comportaban de forma medible distinta. Esta semana esa línea ha tenido su resultado más concreto hasta ahora, y señala por su nombre a la librería que usa casi todo el mundo.

La *interferencia proactiva* es un modo de fallo documentado: cuando un valor se sobrescribe repetidamente en el contexto, recuperar el actual se degrada según se acumulan las sobrescrituras previas. Probando tres niveles de precisión con bitsandbytes sobre Qwen2.5-7B-Instruct, Mistral-7B-Instruct-v0.3 y Phi-3.5-mini-instruct, **INT4 reduce significativamente la precisión con interferencia alta en todos los modelos** — del 81,0% al 68,3% en Qwen —, confirmado con tests de McNemar pareados a $p \leq 2,6 \times 10^{-6}$ y una regresión de efectos mixtos sobre todos los niveles de interferencia. Está ligado mecánicamente a un aumento de los errores de intrusión de la misma clave, del 21,5% al 24,6% de los intentos, y una ablación sitúa el efecto en el backbone del transformer cuantizado y no en la capa de proyección de salida. INT8, "que se suele suponer segura", también arrastra una penalización menor pero real en dos de los tres modelos [2608.18578](#).

La frase que sostiene todo es la última del abstract: este coste cae sobre aplicaciones con contextos largos, actualizables y semánticamente densos *incluso cuando la precisión agregada en benchmarks parece prácticamente intacta*. Un agente que revisa un valor en su propio contexto a lo largo de una ejecución larga es exactamente esa aplicación. El efecto además invierte el signo en una condición de control numérica, así que esto no es un "4 bits es peor" genérico — es específico, y por eso es accionable.

Eficiencia de serving, que ha tenido buena semana

ReCache ataca un problema real y poco comentado: los agentes vuelven a codificar esquemas de herramientas y skills que se repiten entre peticiones en órdenes distintos, lo que anula el prefix caching normal. Hacer que los bloques KV sean invariantes a la composición iguala el rendimiento de invocación densa (82,3% frente a 82,4% de Inv-F1) con un speedup declarado de $3,655 \times$ en time-to-first-token; el framework completo recorta un 92,43% la memoria de tensores KV asignada [2608.19662](#). Si sirves agentes sobre un catálogo fijo de herramientas, este es tu paper de la semana.

FlashPrefill V2 es el paso de "prototipo a desplegable": paged KV cache, continuous batching, FP8 y kernels alineados con FlashAttention-3/4, para poder colocarse como backend de atención en SGLang. Declara hasta $47,26 \times$ y $27,19 \times$ sobre FlashAttention-2 con 128K de contexto en FP8 y BF16 sobre GPUs H20, y $30,49 \times$ frente a una baseline densa alineada con FA3/4 en FP8 [2608.19758](#). Solo prefill — consulta el glosario si esa distinción no te sale automática.

KeysAndValues es el práctico: una librería open source de AWS Labs para hacer fine-tuning de modelos que se co-adapten con una política de sparse attention sobre la KV cache, en una sola A100 de 40 GB, incluyendo una implementación eficiente de H2O con soporte de kernel SDPA dedicado. Entrenar con la

política, en vez de encajarla después, a menudo supera a modelos entrenados con atención exacta 2608.19920.

Dos más que conviene conocer. **WhiteMatter** permite que cada capa de atención atienda a las representaciones de todas las capas de cada token anterior mediante un router que mezcla los estados de capa en un número fijo de canales cacheados — y como ese número de canales fija el tamaño de la cache, la compresión de KV sale de la arquitectura en vez de aplicarse sobre ella. Declara superar a un Transformer normal con un 50% más de capas, conservando casi toda la ganancia con un 50% de compresión de KV cache 2608.18486. Y en el extremo poco glamuroso, integrar SmoothQuant en TorchAO con selección fusionada de GEMM INT8 entre oneDNN, AVX512_VNNI y AMX da hasta 5,8× de throughput extremo a extremo en encoders de la familia BERT sobre Intel Xeon, ya incorporado a PyTorch 2608.18182. Los encoders siguen moviendo mucho ranking y retrieval en producción.

El edge, en serio

FreeToken es la respuesta más directa de la semana a "qué puedo ejecutar realmente". En vez de fijar una estrategia de offloading, mapea de forma continua el cómputo y el estado del modelo sobre lo que la máquina tenga, co-diseñando el layout del modelo, la residencia de expertos, la ejecución CPU–GPU y la reutilización de estado agéntico para modelos MoE. La envolvente que declara: un modelo de 35B en una GPU de portátil de 8 GB, 284B en un sobremesa gaming y el GLM-5.2 de 753B en una sola GPU de workstation, sobre más de 20 modelos MoE y agentes reales de código y uso de herramientas 2608.16157. Son afirmaciones con forma de vendor, de los autores del propio sistema, y el número interesante no es el titular de 753B sino si el caso de 8 GB es usable a un ritmo de tokens sensato — cosa que el abstract no dice.

Como complemento, una caracterización de RAG en el edge sobre una NVIDIA Jetson AGX Thor encuentra que la generación domina el presupuesto con generadores de 7B–8B — alrededor del 90% de la latencia por consulta y el 91% de la energía de GPU — y que una compresión de contexto intermedia da con una región operativa útil, recortando la energía de GPU hasta un 53,2% y la del SoC hasta un 48,2% con pérdida de calidad despreciable. Demasiado suave y dejas energía sobre la mesa; demasiado agresiva y se va la calidad 2608.19535.

§3 · PUBLICACIONES

Pesos que se publicaron

Una semana tranquila, que es lo que parecen la mayoría de las semanas — una familia, dos tamaños, ambos MIT. Solo pesos abiertos.

FECHA	MODELO	POR QUÉ IMPORTA
18 ago	Ornith-1.5-35B-A3B	MoE, ~35B totales pero ~3B activos por token , 262.144 de ventana de contexto nativa ampliable a cerca de 1M con YaRN, licencia MIT. Declara 79 en SWE-bench Verified, 59,6 en SWE-bench Pro, 67,8 en Terminal-Bench 2.1 y 89,2 en GPQA Diamond, promediados sobre cinco ejecuciones según la card. Aquí el número que importa es el de parámetros activos: 3B activos es un coste de decode de portátil colgado de un modelo de clase 35B. Las conversiones a GGUF aparecieron el mismo día.
18 ago	Ornith-1.5-9B	Denso de 9B, ~19 GB en bf16, mismos 262K de contexto y misma licencia MIT, descrito como el miembro ligero de la familia. Declara 70,6 en SWE-bench Verified y 86,4 en GPQA Diamond. Cabe en una GPU de consumo sin cuantizar, cosa que el 35B-A3B no.

Fuera de ventana pero adyacente, y la publicación más interesante si te importa la procedencia: DFM Mimir v1 es un Hierarchical Reasoning Model de 1B entrenado desde cero sobre 161 datasets usando solo datos de post-entrenamiento con permisos, que declara ser competitivo con modelos bastante mayores y estado del arte en danés (2608.13517, 13 de agosto, fuera de ventana). Datos de entrenamiento con licencia limpia y calidad cercana a la frontera es una afirmación que merece seguimiento, porque es justo la que la mayoría de publicaciones de pesos abiertos no puede hacer.

§ 4 · CONCLUSIONES

Qué haría yo

Ordenado por madurez, no por lo interesante que sea la cosa.

ADOPTAR

Si expones `llama-server` con autenticación activada, **actualiza**. #26347 es un cambio pequeño sobre una filtración real: `/v1/models` estaba entregando rutas del sistema de ficheros, y por tanto nombres de usuario, a quien llamara sin autenticar. Nada más de esta semana es urgente; esto sí.

ADOPTAR

Mide tu despliegue de 4 bits con una tarea que sobrescriba valores en el contexto, antes de fiarte de él dentro de un agente. El resultado de interferencia proactiva 2608.18578 es lo bastante concreto como para actuar: monta una prueba en la que una variable se reasigne varias veces y el modelo tenga que decir el valor actual, y ejecútala con la precisión que tienes desplegada. Tu suite de benchmarks no va a pillar esto, por construcción — ese es el argumento central del paper. Si la prueba se degrada, INT8 es el paso barato antes de FP16.

VIGILAR

Speculative decoding: adóptalo para salida estructurada, sáltatelo para prosa. La tabla de DSpark #27383 pone números a algo que se suele decir de forma vaga. Antes de alojar un draft model, mira qué genera de verdad tu carga. Ediciones de código, reescritura estructurada, diffs, extracción: merece la pena. Explicación, resumen, chat: estás pagando memoria por un tercio más de velocidad. Mide la tasa de aceptación, no el throughput — la aceptación es lo que predice si seguirá funcionando cuando cambie la carga.

VIGILAR

Si estás haciendo on-policy distillation multi-profesor, para y lee antes 2608.16647. "Enruta los prompts a expertos de dominio" es el diseño natural, y la evidencia dice que no consigue confinar la influencia de cada profesor, así que las capacidades se canjean entre sí de una forma que el enrutado no arregla. Las parejas profesor/alumno del mismo origen transfieren mucho mejor que las de origen distinto — una restricción barata de respetar en el momento de elegir profesor, y cara de descubrir después.

VIGILAR

Servidores de agentes sobre un catálogo fijo de herramientas: ReCache va por el buen camino.

2608.19662 Un 92,43% menos de memoria KV asignada para esquemas que se repiten en distinto orden es lo bastante grande como para cambiar lo que cabe en una máquina. El código está publicado; las cifras son de los propios autores sobre un benchmark que ellos montaron, así que esto hay que reproducirlo con tu propio conjunto de herramientas antes de acercarlo a producción.

SOLO LABORATORIO

Las soluciones de OPD en sí. SOPD, GC-OPD y Open-MOPD son creíbles y las tres declaran mejoras sobre la OPD estándar en sus propios montajes (2608.16333, 2608.19181, 2608.19098), pero tienen una semana, ninguna se ha reproducido de forma independiente y cada una añade una pieza móvil a un pipeline de entrenamiento. Lo útil ahora es el diagnóstico, no las recetas: si tu alumno destilado se estanca, sospecha que una señal por token está parcheando pasos sin enseñar caminos.

La envoltura de FreeToken. Ejecutar un modelo de 753B en una sola GPU de workstation 2608.16157 es la afirmación más llamativa de la semana y la menos verificable desde el abstract: no se cita ningún ritmo de tokens para las configuraciones más limitadas, y "ejecuta" y "usable" son afirmaciones distintas con 8 GB. Merece probarlo precisamente porque es fácil de refutar uno mismo.

NOTA

Una semana no es una tendencia. Que cuatro papers converjan en la misma limitación de la on-policy distillation es lo más interesante de aquí, y siguen siendo cuatro papers en siete días. Parte de eso es convergencia genuina sobre un problema real; parte es el propio agrupamiento de arXiv, donde una limitación conocida se pone de moda al mismo tiempo. La posición honesta a esta cadencia es que el diagnóstico merece interiorizarse y las soluciones merecen revisarse dentro de un mes, y este número no va a fingir otra cosa para sonar más rotundo.

Fuentes

LLAMA.CPP §1

[Releases de llama.cpp](#) — tags de build hasta [b10590](#) (23 ago).

Los pull requests integrados del 17 al 23 de agosto se enumeraron desde la API de GitHub, no desde un boletín, y cada PR citado abajo se consultó individualmente para confirmar que había hecho merge.

[#27383](#) speculative decoding DSpark para LFM2 · [#27508](#) DSpark para bailingmoe3

[#27390](#) Metal: dequantizar KV cuantizada antes de flash attention · [#27438](#) restringido a batches grandes · [#27370](#) dequantización Q8_0 con tipos empaquetados

[#22877](#) quantize: liberar pesos tras cada capa

[#26347](#) server: endpoints de modelos privados con autenticación

[#27346](#) server: preset dedup-cache-models · [#27376](#) server: /metrics mientras duerme

[#26608](#) BailingMoE3 · [#25505](#) GraniteSWA / GraniteMoeSWA

[#26336](#) Q2_K MMVQ y ESIMD reordenados para SYCL · [#27486](#) revert · [#27490](#) reintegración con el arreglo

[2608.16333](#) Step-Level On-Policy Distillation

[2608.19181](#) Beyond Teacher Likelihood: Group-Calibrated On-Policy Distillation — [código](#)

[2608.19098](#) Open-MOPD: Diagnosing and Fixing Capability Imbalance in Multi-Teacher On-Policy Distillation

[2608.16647](#) Every Coin Has Two Sides: On the Dual Nature of Generalization in On-Policy Distillation

[2608.18578](#) Compress and Forget: bitsandbytes Quantization Amplifies Proactive Interference — [código](#)

[2608.19662](#) ReCache — [código](#)

[2608.19758](#) FlashPrefill V2: Block-Sparse Prefill Attention

[2608.19920](#) Learning how to Forget — [KeysAndValues](#)

[2608.18486](#) WhiteMatter: All-to-All Cross-Layer Connections via KV Mixing

[2608.18182](#) Efficient INT8 Inference of Small NLP Models on Server CPUs

[2608.16157](#) FreeToken: Efficient Edge-Native MoE Serving

[2608.19535](#) From Retrieved Context to Runtime Control: Adaptive Compression for Edge-based RAG

Fuera de ventana, citados como contexto: [2608.14277](#) SimpleOPD (14 ago) · [2608.13517](#) DFM Mimir v1 (13 ago)

[Ornith-1.5-35B-A3B](#) · [Ornith-1.5-9B](#) — las especificaciones y las cifras de benchmark son las que declaran las model cards.

[Daily papers de Hugging Face](#) — nueva en la lista de fuentes esta semana, usada para descubrimiento a lo largo de los siete días; cada paper que sacó se leyó después en su abstract de arXiv antes de citarlo.

Las cifras dadas como speedups, ratios de compresión o resultados de benchmark son las que declaran los propios autores o fabricantes sobre sus propios benchmarks — trátalas como afirmaciones, no como mediciones, hasta que las reproduzcamos. Todo lo fechado fuera del 17–23 de agosto de 2026 va etiquetado como contexto. Los términos están definidos en la [página de glosario](#).



LLM Radar lo publica semanalmente [Montevive](#) · [Feed Atom](#) · ai@montevive.ai · [Privacidad](#)