

Muse Glimmer 30B: An On-Device Agentic Model

Muse Glimmer team¹

¹Meta Superintelligence Labs

We introduce Muse Glimmer, an open-source 30B parameter causal language model with perception encoder purpose-built for autonomous agentic tasks on consumer hardware. Muse Glimmer integrates multi-step reasoning, reliable tool use and failure recovery into a single model that runs locally on consumer devices without requiring cloud infrastructure. The model accepts text and image inputs and produces text outputs, was trained on data from over a hundred languages, and is quantized to approximately 4-bit precision for efficient single-GPU inference. Muse Glimmer achieves state-of-the-art performance for its size class on agentic benchmarks, setting a new standard compared to the models of its size: Gemma4-31B and Qwen3.6-27B on tool use, multi-step planning, and agentic task completion. We release the model under the Apache 2.0 license to foster open-source development of capable, safe, local AI agents.

Date: October 19, 2026

Model: <https://huggingface.co/meta-models/Muse-Glimmer-30B>

Blogpost: <https://research.meta.ai/blog/introducing-muse-glimmer-open-agentic-model>



1 Introduction

Recent large language models have demonstrated strong performance on complex reasoning, program synthesis, and tool-augmented problem solving (Meta AI, 2025; OpenAI et al., 2024, 2026; Guo et al., 2025). Work on open-weight models has shown that architectures with substantially fewer parameters can recover much of the performance of their larger counterparts when given sufficient data or targeted optimization (Gemma Team et al., 2026; Qwen Team, 2025; Abdin et al., 2024). Nevertheless, many publicly available models aimed at consumer hardware are trained as general-purpose systems and lack the specialization required for reliable autonomous agent behavior, such as sustained multi-step reasoning, robust tool invocation, and principled handling of private user context. This leaves an unaddressed gap between the capabilities of proprietary cloud-hosted systems and those available to developers building local AI agents.

We present Muse Glimmer, a 30-billion parameter language model purpose-built for autonomous agentic tasks suitable for running entirely on consumer hardware. Muse Glimmer is distilled from Muse Spark, Meta’s frontier-class model (Menghini et al., 2026), inheriting its reasoning depth and tool-use reliability while being compressed into a form factor that fits on devices that have 32GB of dedicated memory available. The result is a model that delivers strong intelligence for agentic on-device workloads: multi-step planning, sequential tool invocation, failure recovery, and long-horizon task execution. The model supports ≈ 140 languages, with robust multilingual knowledge and reasoning and demonstrated multilingual agentic capabilities.

Our main contributions are as follows:

- We release Muse Glimmer, a 30B model that sets a new standard for its size class across tool use, multi-step reasoning, and task completion and performs strongly compared with Gemma4-31B and Qwen3.6-27B.
- We demonstrate that targeted distillation from a frontier-class teacher can produce a capable small model, achieving good performance with fewer parameters and lower inference compute cost.
- We demonstrate that a dense architecture delivers high capabilities given our memory constraint on the size of the model as compared to mixture-of-experts.
- We show that through a combination of 4-bit quantization and speculative decoding, the model achieves

suitable speeds for usage on edge hardware while preserving its quality.

- We release artifacts under Apache 2.0: full-precision weights, 4-bit quantized weights, and a DFlash (Chen et al., 2026) head for speculation, enabling the community to deploy, fine-tune, and extend the model.

2 Architecture

Muse Glimmer is a 30B parameter causal language model augmented with a dedicated perception encoder for multimodal input. The model accepts interleaved text and image tokens and produces text output. We summarize the architecture in Table 1 and detail the key design choices below.

Table 1 Architecture of Muse Glimmer.

Property	Value
Architecture	Dense Causal Transformer
Total parameters	$\sim 29.6\text{B}$
Language model (transformer blocks)	$\sim 25.16\text{B}$
Token embedding (untied)	$\sim 1.34\text{B}$
Output projection (untied)	$\sim 1.34\text{B}$
Perception encoder	$\sim 1.8\text{B}$
Input modalities	Text + Image
Output modality	Text
Hidden dimension	6656
Layers	52
Attention heads (Q / KV)	32 / 2 (GQA ratio 16:1)
Head dimension	128
FFN type	SwiGLU
FFN intermediate dimension	19,968
Sliding window size	2048
Attention pattern	[Local, Local, Local, Global] repeating
Position encoding	RoPE ($\theta = 500,000$), local layers only
Gated attention	Yes
Vocabulary size	202,048
Context length	128K+

2.1 Language model

The language backbone comprises 52 transformer layers with a hidden dimension of 6656, totaling $\sim 25.16\text{B}$ parameters (excluding embedding and output layers). We use untied input and output embeddings over a vocabulary of 202,048 tokens, each contributing $\sim 1.34\text{B}$ parameters ($202,048 \times 6656$).

Attention. Each layer uses grouped-query attention (GQA) with 32 query heads and 2 key-value heads (ratio 16:1) at a head dimension of 128. We employ a sliding window of 2048 tokens in a repeating pattern of three local-attention layers followed by one global-attention layer, bounding KV-cache growth while preserving long-range information flow. Gated attention is applied throughout. Rotary position embeddings (RoPE) with $\theta = 500,000$ are applied exclusively to the local attention layers; the global layers rely on the accumulated positional signal from surrounding local layers, making the model extendable.

Feed-forward network. Each layer’s FFN uses SwiGLU activation (Shazeer, 2020) with an intermediate dimension of 19,968. The SwiGLU formulation uses three weight matrices—gate projection $W_1 \in \mathbb{R}^{d \times d_{\text{ff}}}$, up projection $W_2 \in \mathbb{R}^{d_{\text{ff}} \times d}$, and down projection $W_3 \in \mathbb{R}^{d_{\text{ff}} \times d}$ —yielding an effective expansion ratio of $\sim 3\times$ the hidden dimension.

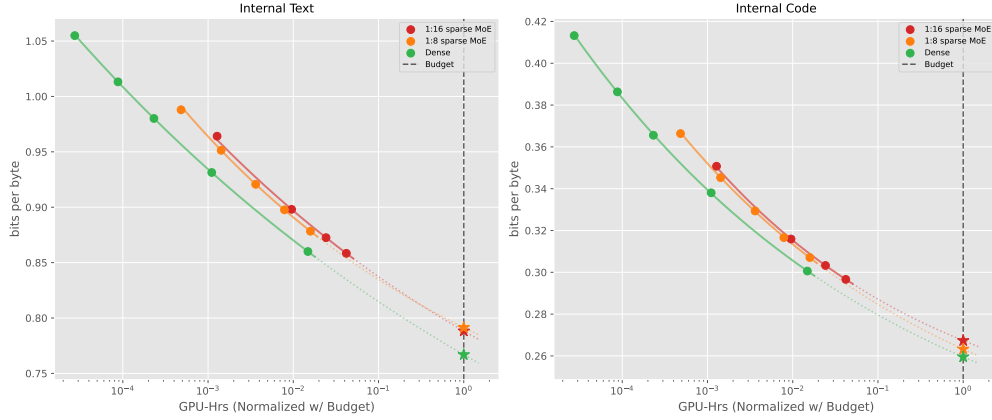


Figure 1 Power law fits of bits-per-byte with respect to GPU-Hrs for the three ladders under consideration (GPU-Hrs is estimated based on final target model MFU). The dense ladder uses 550 TPP, while the MoE ladders use 2,800 TPP (1:8 sparsity) and 6,000 TPP (1:16 sparsity). The total parameter count for the target model in each of these options was chosen such that they can fit into 32 GB VRAM.

2.2 Multimodal input and the perception encoder

To support interleaved text-image input, we adopt the Perception Encoder (Bolya et al., 2025) which leverages a NaViT-style (Dehghani et al., 2023) vision backbone. The encoder is a $\sim 1.8\text{B}$ -parameter ViT-G/14 with 50 layers, width 1536, and patch size 14 that processes images at native resolution using a 448-px reference grid. Its output tokens are spatially downsampled via a 2×2 pixel-shuffle, passed through a two-layer MLP adapter, and then projected into the language model’s embedding space by a single bias-free linear layer followed by parameter-free RMSNorm.

Unlike approaches that rely on dynamic tiling with a thumbnail / global crop (Cho et al., 2026; Liu et al., 2024b), we use a single aspect-ratio-preserving resize to a valid token grid. After the 2×2 spatial downsampling, one output visual token corresponds to a 28×28 pixel cell originating from 14×14 ViT patches. Thus, a 448×448 image yields $32 \times 32 = 1024$ raw ViT patches and $16 \times 16 = 256$ output visual tokens. For higher-resolution images, we increase the grid up to a maximum of 4096 output tokens per image.

We keep the perception encoder frozen throughout training. While Muse Glimmer can ingest video as a sequence of frames, it is not explicitly optimized for video. Instead, we optimize for interleaved text and image understanding, including multi-image contexts.

2.3 Design for on-device efficiency

Our primary design constraint for Muse Glimmer was to operate reliably within a strict 32 GB VRAM envelope, which is the practical ceiling for consumer-grade hardware. To determine the optimal architecture under this hardware limit, we evaluated candidate configurations by training a series of progressively scaled architectural ladders that compared a dense baseline against two Mixture-of-Experts (MoE) designs featuring different routing sparsity (1:8 and 1:16 active experts). We standardized this comparison by keeping the total memory footprint at inference fixed. Under this constant memory ceiling, we estimated the target tokens-per-parameter (TPP) based on our allocated training GPU-hours and the estimated Model FLOP Utilization (MFU) for each configuration. Given our compute budget, all of our ladders operate in over-trained regimes. Scaling curves (Fig 1) demonstrated that within constraints of training compute and parameter size, dense configurations yielded a $2 \times$ - $2.5 \times$ increase in compute efficiency compared to the MoE counterparts. Consequently, we selected a dense 30B parameter design for the architecture.

Beyond static parameter weights, the memory footprint of long-horizon agentic sessions is heavily dominated by the size of the key-value (KV) cache (Gemma Team et al., 2026). To mitigate this during multi-step agentic execution, Muse Glimmer incorporates techniques, such as the sliding-window attention to bound KV-cache growth for long contexts, and using grouped-query attention with only 2 KV heads per layer with

Table 2 Muse Glimmer capability scorecard. We compare Muse Glimmer at *high* reasoning strength against the competitive open models Gemma4-31B and Qwen3.6-27B across agentic, tool-use, reasoning, multimodal, memory, and privacy/security evaluations. Best capability per row in **bold**. Higher is better unless marked ↓. All benchmarks were screened for contamination.

Capabilities Scorecard				
Domain	Evaluation	Muse Glimmer (high)	Gemma4-31B (thinking)	Qwen3.6-27B (thinking)
Agentic & Coding	τ^3 -Banking	23.5	15.1	16.7
	WildClawBench	47.6	37.6	43.2
	GAIA2	43.3	36.4	40.0
	GDPval-AA	953	811	1141
	SciCode	43.6	43.4	39.8
	OSWorld-Verified	65.9	58.5	75.6
	SWE-Bench Verified	76.0	66.6	77.2
	SWE-Bench Pro	51.2	36.9	50.2
	TerminalBench 2.1 (Terminus2)	51.7	43.4	60.7
Tool Use	MCPAtlas	75.5	54.2	62.5
Reasoning & Knowledge	DeepSearchQA	74.6	61.7	71.1
	AIME 2026	94.7	89.2	94.1
	GPQA Diamond	83.5	85.7	84.2
	HLE	22.0	23.6	23.1
Instruction Following	IFBench	77.0	76.0	70.8
	SkillsBench	44.3	32.4	46.6
Multimodal	CharXiv Reasoning	78.8	77.7	78.4
	MMMU-Pro	74	73	75
	OmniDocBench v1.5	75.8	72.5	77.8
	Screenspot Pro	75.4	75.9	76.1
Memory & Long Context	Beam 128K	65.1	58.2	63.0
	AA-LCR	80.0	68.3	73.3
Multilingual	Omnilingual Gaia2	60.4	37.5	45.0
	Linguini	8.8	3.8	2.5
Privacy & Security	CIMemories (violation ↓)	26.4	12.1	53.4
	CIMemories (coverage ↑)	64.8	53.0	66.9
	Siren AgentDojo (ASR@5 ↓)	28.4	25.6	40.3
	Siren AgentDojo (utility ↑)	94.2	90.8	92.7

head dimension of 128. To further compress the KV cache, we evaluated a simplified Multi-Head Latent Attention (MLA) variant (Liu et al., 2024a) in global layers by projecting a single joint key-value vector. We tested two configurations: matching either the baseline’s head count (halving KV cache size) or its total KV footprint (doubling head count). Both underperformed standard Grouped Query Attention (GQA) (Ainslie et al., 2023) on long-context benchmarks such as RULER (Hsieh et al., 2024), leading us to retain GQA.

3 Model Evaluation

Effective autonomous agency requires diverse capabilities. Agentic task completion such as navigating popular scaffolds, writing and debugging code or resolving multi-turn requests requires the model to chain reasoning over long horizons, invoke tools with precise schema, interpret multimodal observations (screenshots, charts, documents), and retain state across sessions. Tool-use and function calling serve as the execution interface through which reasoning is translated into environment actions, while instruction following and steerability determine whether the model can adhere to complex, compositional directives that real-world orchestration systems impose. Long-context memory underpins many of the above: an agent that cannot recall earlier turns may fail at non-trivial multi-step tasks even if it exhibits strong reasoning. We also evaluate multilingual

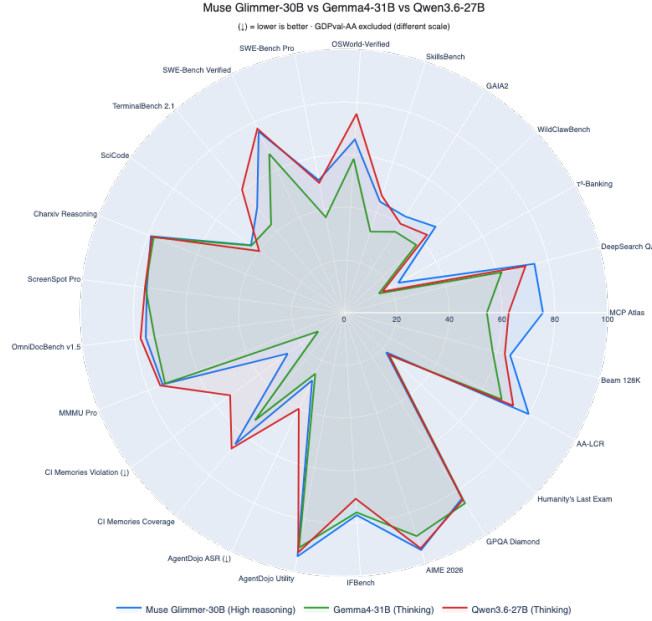


Figure 2 Muse Glimmer capability radar chart. We compare Muse Glimmer at high reasoning strength against the competitive open models Gemma4-31B and Qwen3.6-27B

agentic capabilities and a variety of non-agentic ones covering multilingual reasoning, reading comprehension, knowledge and linguistic competence. The model results are presented in Table 2 and Figure 2. Benchmark descriptions and evaluation details, including which results were run internally and which externally, are presented in Section A.

4 Pre-training

4.1 Tokenizer

Muse Glimmer employs the Llama 4 (Meta AI, 2025) Byte-Pair Encoding (BPE) tokenizer with a total vocabulary of 202,048 tokens, comprising 200,000 base tokens and 2,048 special tokens. The base vocabulary was trained on a balanced data mix of 35% code, 35% multilingual text, 15% mathematics, and 15% English, optimized for compression efficiency across diverse domains. The special token budget is shared across modalities, unifying text, vision, reasoning, and agentic capabilities under a single vocabulary space.

4.2 Pre-training and Mid-training Stages

Training proceeds in multiple stages, a pre-training phase followed by two mid-training phases. Each stage serves a distinct purpose: the pre-training phase acquires broad knowledge and reasoning through logit distillation from the teacher; mid-training progressively introduces domain-specific capabilities and extends the effective context length.

Pre-training. The base model is trained by logit distillation from Muse Spark. We employ co-distillation, logging the top 256 logits from Muse Spark during its pre-training. Unlike prior formulations that continuously blend soft and hard targets (Meta AI, 2025), our objective consists of only KL divergence for the first 95% of pre-training. For the final 5%, we transition to a cross-entropy-only (CE) loss on the ground-truth targets to address the truncation bias inherent to top- k logit distillation (e.g., Anshumann et al., 2025). The pre-training data comprises a large-scale, diverse collection spanning web documents, code, mathematics, multilingual text, and images, with a knowledge cutoff of 2026-01-04.

Checkpoint averaging. At the conclusion of pre-training, we apply checkpoint averaging ("souping") (Izmailov et al., 2018; Wortsman et al., 2022) to produce the initialization for mid-training. Following recent work demonstrating the effectiveness of weight averaging during LLM pre-training (Li et al., 2025b; Sanyal et al., 2023; Groeneveld et al., 2025), we save checkpoints every 200 steps during the final 5% of training (84 checkpoints total) and sweep over merge strategies: uniform average, weighted moving average (WMA), and exponential moving average (EMA)—as well as the number of checkpoints and stride between them. We find that merging over a short window (200-step spacing) consistently outperforms wider strides, and that EMA and WMA provide slightly more stable results than uniform averaging, though the best-performing configuration is similar across merge types. The final souping recipe we chose averages the last checkpoint with three prior checkpoints spaced 600 steps apart using a uniform average. This yields consistent improvements across all perplexity evaluations relative to the single final checkpoint, with larger gains on smoother benchmarks (e.g., ML Papers, Multilingual Documents). The resulting souped checkpoint serves as the initialization for mid-training Phase 1.

Mid-training Phase 1: Domain infusion. Starting from the souped pre-training checkpoint, Phase 1 introduces targeted synthetic and domain-specific data to improve the starting point for post-training. In addition to organic data (similar to that used in pre-training), the data in this phase also contains synthetic data, including reasoning traces and structured problem-solving demonstrations. The learning rate we use for this phase is lower than the peak pre-training learning rate, so as to balance gains on synthetic-data objectives against acceptable regression on organic perplexity benchmarks.

Mid-training Phase 2: Agentic capabilities. The second phase of mid-training extends the context window from 32K to 128K tokens and enriches the model with agentic reasoning data. For this phase we have used a cross-entropy loss, as the distributional shift introduced by long-context and agentic data makes teacher logits less reliable at this stage. The data mix differs from that of the first mid-training phase by including tool-use and agentic trajectory data and purpose-generated agentic demonstrations from our synthetic pipeline (Section 6.2). The learning rate for this phase is set to lower than the peak pre-training learning rate.

5 Post-training

Post-training converts the distilled base into an aligned agent through a sequence of supervised fine-tuning (SFT), reinforcement learning (RL) and on-policy distillation stages described below.

1. **SFT** Align the base checkpoint to supervised targets: task formatting, instruction following, and reasoning style.
2. **On-policy Distillation** We apply on-policy distillation Agarwal et al. (2024b) where the student generates rollouts and the teacher provides per-token distributional targets on the student’s visited prefixes.
3. **Multi-objective RL.** Optimize toward reward models, verifiers, and task-success signals: reasoning, agentic, reinforcement learning from human feedback (RLHF).

5.1 Supervised Fine-tuning (SFT)

We use a supervised fine-tuning (SFT) recipe. This stage extends the capabilities acquired during mid-training into a general instruction-following setting. It uses a mixture of synthetic and human-generated data covering reasoning, coding, mathematics, general instruction following, and agentic scaffolds such as tool-use and multi-step task execution (Section 6.2). We train for approximately one epoch on each dataset, prioritizing response formatting, data quality, and coverage over repeated exposure to maintain entropy.

5.2 Rollouts extraction

We collected teacher trajectories from Muse Spark, focusing on reasoning, agentic, and tool-use capabilities across multiple unique RL and RLHF runs. For each source run, we selected a range of training checkpoints and retained trajectories where the reward was higher than a threshold value. We prioritized trajectories generated at later checkpoints as these reflected more developed teacher policies. We retained repeated

tasks and prompts when they occurred at different checkpoints or produced different trajectories. Each trajectory was converted into a common conversational training format containing system context, user messages, assistant reasoning, tool interactions, and the final response. This conversion preserved the sequence of model and environment interactions needed to reproduce the teacher’s behavior, while operational metadata that was not visible to the model was removed.

Filtering and quality controls. The extracted trajectories underwent extensive cleaning and filtering to ensure high-quality teacher rollouts. We stripped evaluation metadata (e.g., rewards, grader outputs), occasional prompt echoes, and artifacts from unsupported tool interfaces or collection infrastructure. Trajectories containing references that revealed the teacher’s model identity were excluded to avoid transferring unintended identity behavior to the student. We then validated each sample against the training conversation schema and the model’s input renderer, rejecting malformed message sequences, invalid tool-call structures, unsupported fields, and examples that could not be rendered into the target token sequence.

5.3 On-Policy Distillation and Reinforcement Learning

In the later stages of post-training we leverage on-policy distillation to train Muse Glimmer from the larger Muse Spark model (Kim and Rush, 2016; Gu et al., 2024; Agarwal et al., 2024a), together with reinforcement learning. During on-policy distillation, the student, Muse Glimmer as it is being trained, actively samples rollouts. As the student generates these autoregressive trajectories, the teacher evaluates the student’s generated distributions step by step to produce fine-grained feedback signals. This setup resolves the train–inference distribution shift by training the model on its own output distribution, teaching it to recover when generation paths drift. We find that this intermediate stage consistently improves downstream RL outcomes relative to proceeding directly from SFT to RL.

5.4 Reasoning Modes

Muse Glimmer supports varying reasoning strengths. We trained this capability during RL by splitting per prompt samples into subgroups, where each sub group has a length penalty applied corresponding to its assigned reasoning strength. As applying more thinking leads to higher performance, this may reduce the model’s capabilities. We found that including the full spectrum did not harm performance at higher reasoning strengths and in fact had a beneficial regularizing effect on their chain of thought length. On the other hand, applying full spectrum training does provide less exposure to higher strength samples in each step; these primarily drive capability acquisition, subsequently propagated to lower strengths. Consequently, balancing performance vs step count led us to adopt a routing scheme, whereby a certain proportion of prompts are routed solely to higher strengths while the remainder use the full spectrum.

At inference time, Muse Glimmer supports configurable reasoning depth through a reasoning strength parameter specified in the system prompt. Four levels are defined: *low*, which triggers minimal internal deliberation for simple factual lookup; *medium*, which allocates a moderate thinking budget for tasks requiring some planning; *high*, which enables full chain-of-thought reasoning for complex multi-step problems such as debugging or research; and *xhigh* for maximum capability. The reasoning strength is rendered as a single directive (**Reasoning strength:** <level>) appended to the system prompt, allowing deployment-time control over the trade-off between latency and reasoning quality without retraining. High reasoning is the default for agentic tasks, where the model must plan tool-use sequences, interpret intermediate results, and backtrack when necessary.

5.5 Trustworthy Local Agentic Workflows

We applied SFT and RL to enable Muse Glimmer to better support trustworthy local agentic workflows.

Appropriate Information Flows: Muse Glimmer is trained to better respect the privacy of information when interacting with third parties on an individual’s behalf. The underlying principle behind this notion of privacy is inspired by Contextual Integrity (CI) theory (Nissenbaum, 2004). Here, the privacy of information depends upon the context surrounding the exchange, i.e. the sender/recipient of the information, the type of data being considered, and transmission principles governing the exchange. We first introduce the ability to reason

about the context of information exchange (e.g., when composing emails, or WhatsApp messages) during the SFT stages, through synthetic traces from Muse Spark. These traces capture interactions between an intricate network of individuals (see Section 6.2); we induce behavior via carefully crafted system prompt instructions (also referred to as prompt distillation (Snell et al., 2022; Askell et al., 2021)) during trace generation with Muse Spark, and remove them from the final trace before training. We also discard traces where, despite system prompt instructions, Muse Spark either (a) includes unnecessary information, (b) doesn't include necessary information, or (c) doesn't complete the task at hand. Our traces span both simpler non-agentic, single turn settings as well as more complex, multi-turn tool-use settings in mock personal assistant environments. We finally amplify this behavior during RL by penalizing oversharing, while rewarding correct sharing of necessary information.

Safety via Adversarial Robustness and Principled Refusal: As an agentic assistant operating over tool-mediated environments, Muse Glimmer must remain robust to adversarial manipulation while maintaining its ability to complete legitimate tasks. We train Muse Glimmer to exhibit complementary safety behaviors such as resisting indirect prompt injections hidden in tool-returned data, refusing harmful or policy-violating requests, and respecting a principal hierarchy that governs how conflicting instructions are resolved. We leverage the synthetic data generation pipeline from Section 6.2 to produce agentic safety traces via Muse Spark, using rejection sampling under a charter-aware constitution to ensure the agent correctly ignores adversarial content, refuses unsafe sub-tasks while completing safe ones, and offers defensive alternatives. We discard traces where the agent either (a) succumbs to an adversarial instruction, (b) over-refuses a legitimate task, or (c) fails to complete the benign objective. Our safety traces span both simpler single-turn settings and complex multi-turn tool-use scenarios across diverse domains. During RL, we further amplify robustness by deterministically penalizing trajectories where the agent executes an attacker-seeded action, while jointly rewarding successful task completion and explicit recognition of untrusted content in the model's chain-of-thought reasoning.

6 Data

6.1 Pre-training Data

Our pre-training dataset is a large-scale, diverse collection spanning a broad range of domains and modalities, including web documents, code, STEM content, multilingual text, and images. We filter each domain using a combination of rule-based heuristics, classifier-based quality scoring, and aggressive deduplication. In addition, we decontaminate the corpus against benchmark and evaluation sets to minimize train-test overlap. The knowledge cutoff date is 2026-01-04. List of supported languages is listed in B.6.

6.2 Synthetic Data

Training LLMs as effective tool-using agents requires high-fidelity trajectories that demonstrate complex reasoning, privacy awareness, and proficiency in tool selection and execution. Given the scalability and resource challenges associated with collecting real-world data, we used a pipeline for generating synthetic supervised fine-tuning data. This pipeline uses an always-on AI agent harness operating within Docker-container with simulated environments that mimics applications such as popular email services, calendar, documents, WhatsApp, code repository management tools and other common CLI tools and access to the real web.

We adhere to the design principles established in Agents Research Environments (Froger et al., 2025). This design considers a "universe" as a pre-populated digital life—a persona supported by a network of 10–14 contacts, featuring weeks of correspondence, calendar events, documents, and messaging history. By integrating these personas into simulated environments, we obtain cohesive digital lives, ensuring the agent remains agnostic to the simulated nature of its interactions. Tools accessible to the agent utilize the same argument structures and JSON return formats as their real-world counterparts.¹

¹For instance, when an agent executes a query such as `gh issue list --label critical`, the mock environment returns matching issues derived from the synthetic universe definition. Similarly, commands like `wacli send --to "Alice" --message "status update?"` trigger an LLM to generate a contextually appropriate response based on the persona's profile.

The synthetic data framework supports complex conversations, modeling user and environment responses, leverage time shifts to represent long running (multiple days) scenarios, multiple types of memory (native, memory embeddings, cli tools) and other features.

Personas: We first generate diverse personas to simulate potential agent users. Each persona starts from sampled demographic and Big Five personality attributes (McCrae and John, 1992) (age, region, and trait scores), which the model expands into a full profile including occupation, hobbies, and habits. To maximize diversity beyond temperature-based sampling, we additionally ground a subset of profiles in randomly sampled text, prompting the model to generate a persona plausibly matching the text’s author—an approach related to work deriving personas from real text to approximate population-level trait distributions (Hu et al., 2025). Personas are generated in batches and organized into relationship graphs that ground interactions across email, messaging, and calendar platforms. See Appendix B.3 for further details.

Universes: For each persona, we materialize a complete suite of mock tool states, packaged within a dedicated directory. The pipeline selects a realistic subset of applications and populates them in dependency order to ensure that subsequent content—such as emails referencing specific contacts or calendar events involving established participants—remains grounded in the earlier state. Finally, the pipeline generates all necessary workspace files and memory entries.

Scenarios: Once a universe is established, we generate task sequences in two stages. First, given a concise universe summary (contacts, recent communications, document history), the model proposes high-level objectives spanning categories such as execution, search, adaptability, and time management—constrained to only the tools and contacts actually present to avoid hallucinating unavailable resources. Second, these objectives are expanded into runnable scenarios: tasks are rewritten as natural user interactions and any missing setup content (e.g., email threads or calendar invites) is planned as needed. Finally, verifiable rubrics are generated to evaluate agent performance.

Post-processing: We apply a rigorous post-processing pipeline to ensure dataset quality:

- **Deterministic Quality Filtering:** We remove trajectories that fail to meet baseline standards, such as trajectories where the agent failed to complete the goals, duplicated entries, incoherent text, or "fourth-wall" breaks.
- **Undesirable behavioral masking:** We intentionally disable training on any parts of the trajectories that we judge as “undesirable” even if the agent achieved the prescribed goal. This includes bad tool calls, bad memory search queries, and other cases.
- **Grounding Verification:** We employ a post-hoc judging pipeline to perform aggressive filtering based on grounding checks. Any trajectory exhibiting hallucinations, false successes, or unvalidated outcomes is removed from the final dataset.

7 Inference Optimization

Our focus for inference is on low batch size regime for a range of edge devices, so we primarily cover: weight-only quantization to reduce the memory footprint of the weights and the data transfer during decode; training and deployment of a DFlash drafter for speculative decoding.

7.1 Quantization

The weights of Muse Glimmer occupy 55.7 GB in BF16. We target deployments with 24 or 32 GB of memory while reserving capacity for the unquantized KV cache, draft model, perception encoder. We therefore target a language-model weight size below 20 GB. We compare post-training quantization (PTQ) variants using the change in negative log-likelihood relative to BF16, denoted ΔNLL , on held-out text and code data. We use ΔNLL to select candidates for downstream reasoning and agentic evaluations.

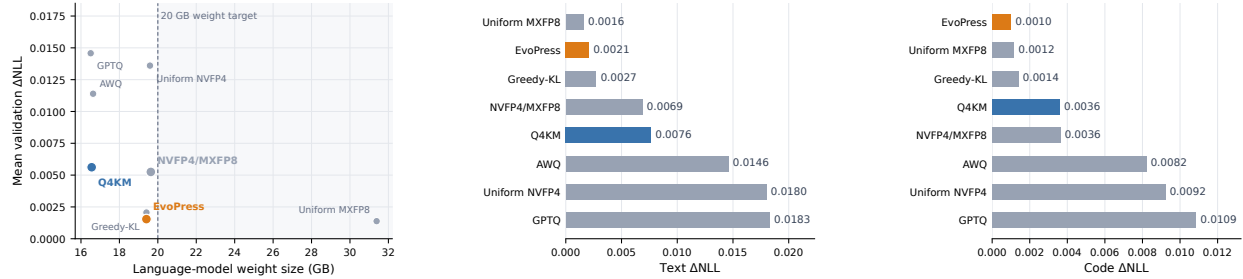


Figure 3 Memory and held-out validation quality of the evaluated PTQ methods. **(left)** Language-model weight size against mean Δ NLL across held-out text and code data. The chosen deployment profiles are highlighted. **(center-right)** Text and code Δ NLL for the same methods, ordered from lowest to highest within each panel. Lower values are better.

Static per-tensor allocations. We start from the popular Q4_K_M profile from `llama.cpp`², where the Value, FFN-down and Output projections are upcasted to higher precision in a subset of the layers. To be more conservative, we apply the upcasting on all layers and we test different precision formats beyond K-quant from `llama.cpp`.

These formats include GPTQ (Frantar et al., 2023) and AWQ (Lin et al., 2024), where the lower-precision tensors use INT4 and the upcasted tensors use INT6, all with group size 128. We apply the same tensor allocation to floating-point formats, using NVFP4 with 16-weight blocks for the lower-precision tensors and MXFP8 with 32-weight blocks for the upcasted tensors (Abecassis et al., 2025; Rouhani et al., 2023). We calibrate this allocation using our weight-only adaptation of AWQ-Lite from NVIDIA TensorRT Model Optimizer (NVIDIA Corporation, 2024–2026). The calibration uses activation statistics to select rescaling factors across adjacent projections. We fold these factors into the BF16 weights before casting each weight once to its target format, while keeping activations in BF16. Lastly, we include uniform NVFP4 and uniform MXFP8 variants as reference points.

For our Q4KM variant, we keep the upcasting applied only to a subset of layers matching `llama.cpp`. We make two changes compared to the original scheme: our output projection uses Q5_K as opposed to Q6_K, and we use a larger group size 128 for both the Q6_K tensors and the Q5_K output projection, compared to 16 for Q6_K and 32 for Q5_K. The larger groups avoid the group-size-16 limitation in MLX and reduce scale overhead when converting K-quant weights to the affine representation used by our MLX implementation in ExecuTorch³. Both changes show negligible quality degradation.

Dynamic allocations. We also test whether calibration measurements support a more flexible allocation than the static Q4KM schedule. Motivated by the per-tensor assignments of Unsloth Dynamic 2.0,⁴ we introduce Greedy-KL, which starts from Q6_K and lowers tensor precision according to the increase in calibration KL divergence per byte saved. It uses the remaining capacity to move the most sensitive Q4_K tensors to Q5_K. For comparison, we adapt EvoPress, an evolutionary method for mixed-precision quantization (Sieberling et al., 2025), to the same Q4_K, Q5_K, and Q6_K search space. EvoPress evolves precision assignments under a memory constraint using KL divergence from BF16 as its search objective.

Both searches expose the query, key, value, and output-gate slices of the fused attention projection independently. The FFN gate and up projections are likewise exposed independently. Together with the attention output, FFN-down, token embedding, and output projection, this gives 418 weight units. No tensor role is fixed at high precision. The winner EvoPress profile assigns 51, 130, and 237 units to Q4_K, Q5_K, and Q6_K, respectively, all with group size 32. The search selects Q4_K for the token embedding and Q5_K for the output projection, with a total of 19.65 GB for the model weights.

²<https://github.com/ggml-org/llama.cpp/blob/23e0d70bacaaca1429d365a44aa9e7434f17823b/src/llama.cppv#L5188>

³<https://pytorch.org/blog/fast-ondevice-agentic-ai-with-executorch/>

⁴<https://huggingface.co/unsloth/gemma-4-31B-it-GGUF>

Table 3 Downstream accuracy for quantization. Average accuracy and response length in tokens for different quantized variants across a mix of 15 downstream benchmarks.

Model Variant	Average Accuracy (%)	Average Response Length
BF16	75.7	11070
EvoPress	75.6	11417
Greedy-KL	75.3	10754
Q4KM	74.8	10774
NVFP4/MXFP8	74.4	10988

Table 4 DFlash drafter configuration.

Component	Setting
Draft layers	5
Block size	16
Attention	Sliding-window, 2048, all layers
Attention heads	32 query (matches target) / 8 KV (GQA)
Sequence length	128k
Hidden-feature layers	5, uniform over target: {1, 13, 25, 37, 49} of 52

Results. Figure 3 summarizes the memory and validation trade-offs. EvoPress provides the lowest validation loss, and Q4KM provides the smallest memory footprint, so we select these two variants for our 32 GB and 24 GB VRAM deployments, respectively.

Agentic and reasoning evaluations. We evaluate the candidates on 15 downstream tasks to compare against our NLL-based results. Table 3 shows our results. We see that Δ NLL aligns reasonably well with the average accuracy across all tasks combined. Related work has shown that aggressive quantization at low bit widths can increase chain-of-thought length and its associated inference cost (Liu et al., 2025; Lotfi et al., 2026). However due to the conservative nature of our chosen quantization schemes, we observe no significant variation in chain-of-thought length relative to BF16.

7.2 Speculative Decoding

We adopt DFlash (Chen et al., 2026) as our speculative decoding strategy, matching the current open source leading standard. Our drafter is a lightweight block-diffusion model (Arriola et al., 2025) that predicts in a single forward pass an entire block of marginal probabilities that are sampled independently to build the draft block. Table 4 summarizes our DFlash configuration.

We train the DFlash drafter offline in a supervised manner using on-policy rollouts from the target model, drawn from a combined pool of SFT and RL prompts. During training, starting anchors for the masked blocks are sampled at a rate of 1/16 from the response tokens, and the model is optimized using the D-PACE (Dynamic Position-Aware Cross-Entropy) objective (Wu et al., 2026).

8 Trust Considerations

We address catastrophic risk assessment, content safety, agentic-specific risks and other trustworthiness considerations through a combination of train-time mitigations built into the model weights.

8.1 Safety and Privacy Policies

Muse Glimmer is evaluated and mitigated across three risk axes—content safety, agentic risk (policies for irreversible-action confirmation, data minimization, scaffold boundary respect, and indirect prompt-injection resistance, enforced through dedicated training data), and catastrophic risk (chemical, biological, cyber,

and loss-of-control assessments), using both train-time alignment built into the weights and a lightweight inference-time safety mechanism.

8.2 Train-Time Mitigations

Safety behavior in Muse Glimmer is established through three stages of post-training:

1. **Safety SFT.** Curated examples demonstrating correct safety behavior, including refusal of harmful requests, calibrated responses to borderline prompts, and proper handling of agentic safety scenarios, are included in the supervised fine-tuning data mix. We reuse established alignment recipes while augmenting with Muse Glimmer-specific agentic safety data covering tool-use boundaries, prompt injection resistance, and permission handling.
2. **Safety RL.** Reinforcement learning stages incorporate safety-specific reward signals that penalize policy violations while rewarding helpful, non-refusing responses to legitimate requests. This stage hillclimbs on content safety evaluations through iterative checkpoint evaluation and mitigation.
3. **Appropriate information flows** As described in Section 5.5, principles of data sensitivity recognition, minimization, and local-first execution are embedded directly into the model weights through dedicated synthetic training data, providing defense-in-depth beyond what system prompts alone can achieve.

9 Related Work

On-device small language models. Running capable language models directly on consumer hardware has been enabled by a wave of compact, openly released model families. The Llama model series (Touvron et al., 2023a,b; Grattafiori et al., 2024) showed good performance as compared with frontier models while spawning a large open-source ecosystem of fine-tunes, quantization tooling, and local inference engines (llama.cpp, Ollama, LM Studio). This ecosystem established the practical infrastructure such as GGUF quantization formats, speculative decoding support, Apple Silicon optimizations, on which models like Muse Glimmer now rely for consumer deployment.

Microsoft’s PHI-4 (Abdin et al., 2024) is a 14B-parameter model trained largely on curated synthetic data. Google DeepMind’s recent GEMMA 4 (Gemma Team et al., 2026) release spans dense models from 2.3B to 31B parameters (alongside a Mixture-of-Experts variant) and relies on quantization-aware training and KV cache optimizations to fit tight memory and latency budgets. Finally, the QWEN3 (Qwen Team, 2025) family offers dense and MoE models spanning 0.6B–235B parameters, unifies fast (non-thinking) and deliberative (thinking) inference within a single model, and extends multilingual coverage to 119 languages. Such model families define the small-LLM landscape against which we position our 30B model.

Teacher-student Distillation. Distilling the capabilities of large teacher models into smaller, deployable students has emerged as a dominant paradigm for building capable compact models. DeepSeek-R1 (Guo et al., 2025) demonstrated that reinforcement learning alone can elicit sophisticated chain-of-thought reasoning in large models, and that the resulting reasoning traces can then be distilled into much smaller students (1.5B–70B parameters) that substantially outperform models trained from scratch at equivalent scale. The Qwen3 family (Qwen Team, 2025) extends this idea through strong-to-weak on-policy distillation, in which the student generates its own rollouts and receives dense token-level KL supervision from a larger teacher, yielding gains over off-policy distillation that merely imitates teacher-generated outputs. Concurrent work on Self-Distilled Reasoner (Zhao et al., 2025) and AsyncOPD (Kang et al., 2025) further explores the efficiency frontier of on-policy distillation, showing that student-generated data with teacher feedback is more sample-efficient than traditional supervised fine-tuning. Our post-training recipe (Section 5.3) employs on-policy distillation from the Muse Spark teacher, an approach we found superior to a reinforcement-learning-only recipe for our target model size.

Agentic LLM frameworks. The ReAct paradigm (Yao et al., 2023) established the interleaved reasoning-and-acting loop that underpins most modern LLM agent architectures, showing that language models can effectively chain verbal reasoning with tool invocations to solve multi-step tasks. Subsequent work has focused

on training models for reliable tool use rather than relying solely on prompting. Toolformer (Schick et al., 2024) introduced self-supervised API-call insertion during pre-training, while Gorilla (Patil et al., 2023) demonstrated that fine-tuning on large-scale API documentation enables models to generate correct function calls with high fidelity even as APIs evolve. FireAct (Chen et al., 2023) and Agent-FLAN (Chen et al., 2024) showed that fine-tuning on multi-step agent trajectories produces models that significantly outperform prompted baselines in interactive environments. On the deployment side, open-source orchestration frameworks such as OpenClaw (OpenClaw Contributors, 2025) have popularized locally-hosted agents that chain model inference with tool execution on consumer hardware, creating demand for compact models that can sustain reliable sequential tool invocation across long sessions. Coding agents like SWE-agent (Yang et al., 2024) and OpenHands (Wang et al., 2024a) demonstrate that agentic tool use in software engineering requires sustained multi-step planning and failure recovery—capabilities that our model targets through its agentic alignment training (Section 6.2).

10 Conclusion

This technical report presents Muse Glimmer, a 30B dense model purpose-built for autonomous agentic operation on consumer hardware. Through targeted distillation from Muse Spark, it surpasses comparably-sized open models on agentic benchmarks. Combined with 4-bit quantization and speculative decoding via a DFlash head, Muse Glimmer runs at practical speeds on a single RTX 5090 with minimal quality loss. Multimodal understanding of screenshots, documents, and charts equips the model to operate natively in agentic and information-rich environments. Muse Glimmer covers over a hundred language varieties and leads comparably-sized open models on multilingual agentic evaluation.

All artifacts including model weights and the speculation head are released under Apache 2.0.

11 Authors List

11.1 Core Contributors

Arash Ebrahim Abtahizadeh, Muna Aghamelu, Adil Ahmed, Juan C. Angeles Acuna, Alexis Audran-Reiss, Asaf Avrahamy, Yoram Bachrach, Can Balioglu, Amar Budhiraja, Onur Çelebi, Gaurav Chaurasia, Joy Chen, Lynn Cherif, Lucia Cipolina-Kun, Gal Cohen, Timothée Darcet, Beto de Paola, Digant Desai, Sam Devlin, Derek Dunfield, Ivan Evtimov, Maryam Fazel-Zarandi, Rob Fergus, Jean-Christophe Gagnon-Audet, Grant Gardner, Bhavul Gauri, Paris Giampouras, Sujan Kumar Gonugondla, Antoine Grosnit, Young Han, Michael Hassid, Monuril Islam, Masoud Jalili Sabet, Songhao Jia, Narine Kokhlikyan, Svetlana Kurtsman, Sandra Lefdal, Chee Hau Leow, Noam Levi, Mike Lewis, Sanae Lotfi, Xiyang Luo, Despoina Magka, Saeed Mahloujifar, Neal Mangaokar, Ishita Mediratta, Alexander H. Miller, Hela Momand, Saba Nazir, Kelvin Niu, Mattia Oppen, Parth Pathak, Alberto Pepe, Danielle Pintz, Mikhail Plekhanov, Sidhartha Reddy Potu, Anton Protopopov, Shengyi Qian, Shashank Rajput, Joseph Rocca, Scott Roy, Luca Sbordone, Amin Shakeri, Tatiana Shavrina, Michael Shvartsman, Ruan Silva, Anya Sims, Eric Michael Smith, Hugo Touvron, Mabel Ubong, Isabel Urrego, Yanzheng Xiang, Zhiyuan Yao, Alex Yarmula, He Ye, Jade Yu, Luke Zettlemoyer, Arman Zharmagambetov

11.2 Contributors

Perpetua Fathima Ahlijah, Tremo Ahmed, Enrique Alfonseca, Pedram Bashiri, Nikolay Bashlykov, Aaron Berman, Rohith Bhatia, Siddharth Bhattacharya, Prashant Bhende, Chloe Bi, Shuchao Bi, Clement Anthonioz Blanc, Allen Bolourchi, Andrea Caciolai, Erica Cai, Xiang "Tony" Cao, Yue Cao, Andrew Caples, Jinzhao Chang, Lele Chen, Moya Chen, Weijie Chen, Wenhui Chen, Xiaobai Chen, Yinyin Chen, Zhengxing Chen, Wei Cheng, Pavel Chikulaev, Edward Chou, Sy Choudhury, Brian Clerkin, Marta R. Costa-jussà, Curtis Davis, Myra Deng, Dujian Ding, Hantian Ding, Shuangrui Ding, Jesse Dodge, Shweta Doshi, Shirin Egtesadi, Faith Eischen, Athyu Eleti, Alex Fabbri, Han Fang, Paula Farani, Aiman Farooq, Christoph Feichtenhofer, Darius Feher, Amanda Felix, Carl Chengyan Fu, Sean Gahagan, Yifan Gao, Itai Gat, Akashdeep Goel, Ahuva Goldstand, Bo He, David Heineman, Dylan Herman, Irene Hjelt, Joonas Hjelt, Alexander Holz, Rui Hou, Annie Hu, Yuan-Ting Hu, Bernie Huang, Jing Huang, Kun Huang, Pere-Lluís Huguet Cabot, Ashley Hunter, Srinivasan Iyer, Somya Jain, Michael Jiang, Song Jiang, Wenwen Jiang, Lin Jin, Jacob Kahn, Melanie Kambadur, Yingxin Kang, Justine Kao, Stephane Kasriel, Hossein Kavianiamedani, Ameya Khandekar, Yash Khare, Gefen Kohavi, Kavya Kopparapu, Praveen Krishnan, Venu Kuchibhotla, Ananya Kumar, Leona Lan, Jack Lanchantin, Hunter Lang, Nikolay Pavlovich Laptev, Yuanzhi Li, Yunlu Li, Asad Liaqat, Johnson Lin, Runwei Lin, Zhichao Lin, Shaun Lindsay, Yaron Lipman, Frederick Liu, Mengchen Liu, Qiang Liu, Yinghai Lu, Yujie Lu, Christopher Luc, Faye Ma, Pingchuan Ma, Tengyu Ma, Wufei Ma, Xiaodong Ma, Damir Maksyutov, Josh Malone, Udeep Manchanda, Thomas Mann, Kalyani Marathe, Roberto-Rafael Maura-Rivero, Ruslan Mavlyutov, Thomas Mesnard, Julian Michael, Sagar Miglani, Todor Mihaylov, Naureen Mitha, Archi Mitra, Leo Mok, Naila Murray, Muthukumaravel Muthuraman, Mergen Nachin, Nese Demir Nalci, Saaketh Narayan, Alyssa Newcomb, Peter Ney, Joycelyn Ng, Devin Norder, Maxwell Nye, Rajvardhan Oak, Cody Ohlsen, Artidoro Pagnoni, Yi Pan, Nilesch Pandey, Guan Pang, Richard Yuanzhe Pang, Mahesh Pasupuleti, Parth Patel, Yash Patel, Vladan Petrovic, Norman Ponte, Yundi Qian, Zengyi Qin, Jack Rae, Emmanuel Ramirez, Ramya Raghavendra, Nikhila Ravi, Hongyu Ren, Riandy Riandy, Harrison Rudolph, Kalyan Saladi, Eduardo Sánchez, Karthik Abinav Sankararaman, Aman Shankar, Paresh Shenoy, Alana Shine, Hamid Shojanazeri, Iva Simon-Bubalo, Mitesh Kumar Singh, Tiffany Siqi, Pawan Sisodiya, Ali Sol, Sten Sootla, Leslie St. Pierre, DiJia Su, Suraj Subramanian, Brian Sun, Tony Sun, Zhiqing Sun, Natacha Supper, Jason Talpalar, Ellen Xiaoqing Tan, Liang Tan, Jean-Christophe Testud, Parth Thakkar, Joseph Tighe, Ty Toledano, T Tse, Igor Tufanov, Alex Vaughan, Josh Walters, Alex Wang, Bethany Wang, Meng Wang, Rocky Wang, Weiyue Wang, Xiaolan Wang, Xin Wang, Zijian Wang, Bram Wasti, Quentin Wellens, Yeming Wen, Mary Williamson, Yuncheng Wu, Fei Xia, Tian Xie, David Xu, Jacob Xu, Puxin Xu, Zheng Xu, Shen Yan, Megan Yang, Yining Yang, Nick Yannopoulos, Kevin Yao, Michihiro Yasunaga, Haoxuan You, Michael You, Dingli Yu, Tianhe Yu, Summer Yue, Xiang Yue, Shuwei Zeng, Hanwen Zha, Chen Zhang, Hongming Zhang, Kaiyu Zhang, Meng Zhang, Mingyang Zhang, Yi Zhang, Shengjia Zhao, Chao Zhou, Claire Zhou, Shuyan Zhou, Yanjun Zhou, Yaxuan Zhou, Rosalie Zhu, Yun Zhu

References

- Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.
- Felix Abecassis, Anjolie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, et al. Pretraining large language models with nvfp4. *arXiv preprint arXiv:2509.25149*, 2025.
- Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263, 2024a.
- Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263, 2024b.
- Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, 2023.
- Pierre Andrews, Mikel Artetxe, Mariano Coria Meglioli, Marta R. Costa-jussà, Joe Chuang, David Dale, Mark Duppenthaler, Nathaniel Paul Ekberg, Cynthia Gao, Daniel Edward Licht, Jean Maillard, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Eduardo Sánchez, Ioannis Tsiamas, Arina Turkatenco, Albert Ventayol-Boada, and Shireen Yates. BOUQUET : dataset, benchmark and open initiative for universal quality evaluation in translation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 27515–27535, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1400. <https://aclanthology.org/2025.emnlp-main.1400/>.
- Anshumann, Mohd Abbas Zaidi, Akhil Kedia, Jinwoo Ahn, Taehwak Kwon, Kangwook Lee, Haejun Lee, and Joohyung Lee. Sparse logit sampling: Accelerating knowledge distillation in llms, 2025. <https://arxiv.org/abs/2503.16870>.
- Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models, 2025. <https://arxiv.org/abs/2503.09573>.
- Amanda Aspell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.
- Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.44. <https://aclanthology.org/2024.acl-long.44/>.
- Chaithanya Bandi, Razvan-Gabriel Dumitru, Ben Hertzberg, Divyansh Agarwal, Geobio Boo, Tejas Polakam, Sami Hassaan, Jeff Da, HiJae Kim, Vipul Gupta, et al. Mcp-atlas: A large-scale benchmark for tool-use competency with real mcp servers. *arXiv preprint arXiv:2602.00933*, 2026.
- Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. In *Advances in Neural Information Processing Systems*, volume 38, 2025.
- Andrea Caciolai, Pere-Lluís Huguet Cabot, Chierh Cheng, Albert Ventayol-Boada, Gabriel Mejia Gonzalez, Christophe Ropers, Lucas Bandarkar, Sebastian Ruder, Darlene Sakakihara, Elliot Yun, Pierre Andrews, Grégoire Mialon, Romain Froger, and Marta R. Costa-jussà. Omnilingualaia2: Evaluating the multilingual gap in frontier ai agents, 2026. <https://arxiv.org/abs/2608.08775>.

- Center for AI Safety, Scale AI, and HLE Contributors Consortium. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649:1139–1146, 2026. doi: 10.1038/s41586-025-09962-4. <https://arxiv.org/abs/2501.14249>.
- Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*, 2023.
- Jian Chen, Yesheng Liang, and Zhijian Liu. Dflash: Block diffusion for flash speculative decoding, 2026. <https://arxiv.org/abs/2602.06036>.
- Zehui Chen, Kuikun Du, Wenquan Chen, Yining Liu, Zhengkai Yang, Zhiyuan Liu, and Maosong Sun. Agent-flan: Designing data and methods of effective agent tuning for large language models. *arXiv preprint arXiv:2403.12881*, 2024.
- Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Suyog Jain, et al. Perceptionlm: Open-access data and models for detailed visual understanding. *Advances in Neural Information Processing Systems*, 38:23475–23537, 2026.
- François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.
- Edoardo De Benedetti, Nicholas Carlini, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *arXiv preprint arXiv:2406.13352*, 2024.
- Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim Alabdulmohsin, Avital Oliver, Piotr Padlewski, Alexey Gritsenko, Mario Lučić, and Neil Houlsby. Patch n’ pack: Navit, a vision transformer for any aspect ratio and resolution. In *Advances in Neural Information Processing Systems*, volume 36, pages 2252–2274, 2023.
- Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rognvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. 2026. <https://arxiv.org/abs/2605.00674>.
- Daniel Deutsch, Eleftheria Briakou, Isaac Rayburn Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, Shruti Rijhwani, Parker Riley, Elizabeth Salesky, Firas Trabelsi, Stephanie Winkler, Biao Zhang, and Markus Freitag. WMT24++: Expanding the language coverage of WMT24 to 55 languages & dialects. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12257–12284, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.634. <https://aclanthology.org/2025.findings-acl.634/>.
- Shuangrui Ding, Xuanlang Dai, Long Xing, Shengyuan Ding, Ziyu Liu, Yang JingYi, Penghui Yang, Zhixiong Zhang, Xilin Wei, Xinyu Fang, Yubo Ma, Haodong Duan, Jing Shao, Jiaqi Wang, Dahua Lin, Kai Chen, and Yuhang Zang. Wildclawbench: A benchmark for real-world, long-horizon agent evaluation, 2026. <https://arxiv.org/abs/2605.10912>.
- Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2023.
- Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, Kunal Malkan, Dheeraj Mekala, Pierre Ménard, Gerard Moreno-Torres Bertran, Ulyana Piterbarg, Mikhail Plekhanov, Mathieu Rita, Andrey Rusakov, Vladislav Vorotilov, Mengjue Wang, Ian Yu, Amine Benhalloum, Grégoire Mialon, and Thomas Scialom. Are: Scaling up agent environments and evaluations. *arXiv preprint arXiv:2509.17158*, 2025. <https://arxiv.org/abs/2509.17158>.
- Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, Kunal Malkan, Dheeraj Mekala, Pierre Ménard, Gerard Moreno-Torres Bertran, Ulyana Piterbarg, Mikhail Plekhanov, Mathieu Rita, Andrey Rusakov, Vladislav Vorotilov, Mengjue Wang, Ian Yu, Amine Benhalloum, Grégoire Mialon, and Thomas Scialom. Gaia2: Benchmarking llm agents on dynamic and asynchronous environments, 2026. <https://arxiv.org/abs/2602.11964>.
- Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, Mayank Chaturvedi, Victor Cotruta, Alice Coucke, Phil Culliton, Robert Dadashi, Lucas Dixon, Mohamed Elhawaty, Utku Evci, Clément Farabet, Johan Ferret, Filippo Galgani, Sertan Girgin, Jean-Bastien Grill, Maarten Grootendorst, Jiaxian Guo, Cassidy Hardin, Yanzhang He, Steven M. Hernandez, Omri Homburger, Léonard Hussenot, Juyeong Ji, Armand Joulin, Aishwarya Kamath, Parnian Kassarai, Olivier Lacombe, Preethi Lahoti, Gaël Liu, Gus Martins, Luciano Martins, Tatiana Matejovicova, Ramona Merhej,

Nikola Momchev, Sneha Mondal, Ryan Mullins, Sindhu Raghuram Panyam, Shreya Pathak, Sarah Perrin, André Susano Pinto, Etienne Pot, Angéline Pouget, Alexandre Ramé, Sabela Ramos, Douglas Reid, David Rim, Morgane Riviére, Karsten Roth, Louis Rouillard, Omar Sanseviero, Pier Giuseppe Sessa, Shane Settle, Danila Sinopalnikov, Sara Smoot, Piotr Stanczyk, Andreas Steiner, Lawrence Stewart, Ilya Tolstikhin, Michael Tschannen, Anton Tsitsulin, Nino Vieillard, Renjie Wu, Pingmei Xu, Haichuan Yang, Edouard Yvinec, Li Zhang, Joe Zou, Nicolas Aagnes, Abdelrahman Abdelhamed, Shivani Agrawal, Shubham Agrawal, Ibrahim Alabdulmohsin, Jean Baptiste Alayrac, Uri Alon, Chandramouli Amarnath, Ankesh Anand, Chrysovalantis Anastasiou, Setareh Ariaifar, François-Xavier Aubet, Kyriakos Axiotis, Federico Barbero, Joelle Barral, Alexei Bendebury, Urs Bergmann, Stanley Bileschi, Kat Black, Mathieu Blondel, Sebastian Borgeaud, Arthur Bražinskas, Ryan Burnell, Robert Busa-Fekete, Mu Cai, Glenn Cameron, Charlotte Caucheteux, Garima Chadha, Jetha Chan, Aditya Chawla, Blake Jianhang Chen, Jesse Chen, Lin Chen, Xu Chen, Derek Cheng, Tzu hsiang Chien, Nikolai Chinaev, Yi Chou, Zhaohui Chu, Benjamin Coleman, Pooja Consul, Sam Conway-Rahman, Scott Crowell, Dylan Cutler, Vivek Dani, Samira Daruki, Anil Das, Daniel Deutsch, Nishanth Dikkala, Li Ding, Qiuhan Ding, Shenil Dodhia, Konstantin Donhauser, Tulsee Doshi, Anca Dragan, Alex Druinsky, Sahil Dua, Zoltan Egyed, Danielle Eisenbud, Daniel Eppens, Cindy Fan, Bahare Fatemi, Yassir Fathullah, Vlad Feinberg, Milen Ferev, Takumi Fujimoto, Isaac Galatzer-Levy, João Gante, Simon Geisler, Soham Ghosal, Antonious M. Girgis, Alec Go, Alhaad Gokhale, Alex Grills, Yiming Gu, Pramod Gupta, Guru Guruganesh, Raia Hadsell, Hamza Harkous, Jitendra Harlalka, Demis Hassabis, Anja Hauth, Joe Heyward, Arian Hosseini, Chih-Yang Hsia, I-Hung Hsu, Xiaopeng Huang, Yangsibo Huang, Kevin Hui, Adrian Hutter, Te I, Fotis Iliopoulos, Advait Jain, Ganesh Jawahar, Ziwei Ji, Qilin Jin, Melvin Johnson, Kandarp Joshi, Arun Kandoor, Wang-Cheng Kang, Koray Kavukcuoglu, Mehran Kazemi, Kathleen Kenealy, Amr Khalifa, Phoebe Kirk, Suraj Kothawade, Vitaly Kovalev, Neel Kovelamudi, Adam Kraft, Ravin Kumar, Harish Kuppam, Justin Lannin, Chen-Yu Lee, Seungji Lee, Dmitry Lepikhin, Dongdong Li, Qiujia Li, Valentin Liévin, Ethan Lin, Ziqian Lin, Casper Liu, Tianlin Liu, Tianqi Liu, Xin Liu, Mayank Lunayach, Min Ma, Gagan Madan, Andrii Maksai, Eric Malmi, Michal Matuszak, Daniel McDuff, Gaurav Menghani, Daniil Mirylenka, Karolis Misiunas, Vedant Misra, Andreea Mitran, Kareem Mohamed, Maksim Mukha, Eric Noland, James O'Donnell, Kate Olszewska, Bennett Orlando, Wanqiong Pan, Rina Panigrahy, Unnati Parekh, Chunjong Park, Eric Paskie, Liqian Peng, Bryce Petrini, Slav Petrov, Jonas Pfeiffer, Bilal Piot, Martyna Plomecka, Siim Poder, Octavio Ponce, Arijit Pramanik, David Racz, Anish Rajan, Michelle Ramanovich, Anand Rao, Marvin Ritter, Vitor Rodrigues, Evan Rosen, Mikolaj Rybiński, Noveen Sachdeva, Michaël E. Sander, Rohit Sathyanarayana, Sagar Savla, Samuel Schmidgall, Tal Schuster, Benoit Seguin, Andrew Sellergren, Aliaksei Severyn, Izhak Shafran, Dhruv Shah, Yuan Shangguan, Ashish Shenoy, Pradeep Shenoy, Rakesh Shivanna, Pauline Sho, Lucas Spangher, Wojciech Stokowiec, Tim Strother, Yao Su, Yinghao Sun, Mukund Sundararajan, Andrea Tacchetti, Mor Hazan Taege, Pouya Tafti, Chetan Tekur, Rahul Thapa, Madeleine Traverse, Lenart Treven, Tao Tu, Chien Te Tung, Petar Veličković, Malini Pooni Venkat, Sagar Gubbi Venkatesh, Vidya Venkiteswaran, Francesco Visin, Alex Vitvitskyi, Kiran Vodrahalli, Weiye Wang, Xin Wang, Tris Warkentin, Jan Wassenberg, John Wieting, Lechao Xiao, Hao Xu, Yuhui Xu, Fuzhao Xue, Arun Yadav, Jun Yan, Antoine Yang, Lin Yang, Ming-Hsuan Yang, Ziyu Ying, Jae Hyeon Yoo, Sajjad Zafar, Fred Zhang, Jiageng Zhang, Jianyi Zhang, Xiaofan Zhang, Chao Zhao, David Zhou, and Chen Zou. Gemma 4 technical report, 2026. <https://arxiv.org/abs/2607.02770>.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dirk Groeneveld et al. OLMo 2: Furious. *arXiv preprint arXiv:2501.00656*, 2025.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024. <https://arxiv.org/abs/2306.08543>.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyuan Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojuan Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou,

- Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. <http://dx.doi.org/10.1038/s41586-025-09422-z>.
- Nikita Gupta, Riju Chatterjee, Lukas Haas, Connie Tao, Andrew Wang, Chang Liu, Hidekazu Oiwa, Elena Gribovskaya, Jan Ackermann, John Blitzer, et al. Deepsearchqa: Bridging the comprehensiveness gap for deep research agents. *arXiv preprint arXiv:2601.20975*, 2026.
- Cheng-Ping Hsieh, Simeng Sun, Samuel Krizan, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What’s the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.
- Zhengyu Hu, Jianxun Lian, Zheyuan Xiao, Max Xiong, Yuxuan Lei, Tianfu Wang, Kaize Ding, Ziang Xiao, Nicholas Jing Yuan, and Xing Xie. Population-aligned persona generation for llm-based social simulation. *arXiv preprint arXiv:2509.10127*, 2025.
- Pavel Izmailov, Dmitrii Podoprikin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *Uncertainty in Artificial Intelligence (UAI)*, 2018.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? *arXiv preprint arXiv:2310.06770*, 2024.
- Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjun Kang, Sanghyun Park, Donghoon Kim, Minjae Lee, Minseo Kim, Rishabh Tiwari, Yuchen Zeng, Hyung Il Koo, and Kangwook Lee. Asyncopd: How stale can on-policy distillation be? *arXiv preprint arXiv:2606.24143*, 2025.
- Kilo AI. Pinchbench: Real-world benchmarks for ai coding agents, 2026. <https://pinchbench.com>. Accessed: 2026-07-27.
- Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. <https://aclanthology.org/D16-1139/>.
- Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 8778–8786, 2025a.
- Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, Yifeng He, Shenghan Zheng, Kyoung Whan Choe, Jiankai Sun, Shuyi Wang, et al. Skillsbench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026.
- Yuelin Li et al. Model merging in pre-training of large language models. *arXiv preprint arXiv:2505.12082*, 2025b.
- Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for LLM compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2024.
- Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.
- Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024b.
- Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng Yu, Chun Yuan, and Lu Hou. Quantization hurts reasoning? an empirical study on quantized reasoning models. *arXiv preprint arXiv:2504.04823*, 2025.
- Sanae Lotfi, Polina Kirichenko, Steven Li, and Zechun Liu. Quantized reasoning models think they need to think longer, but they do not. *arXiv preprint arXiv:2606.00206*, 2026.

- Robert R. McCrae and Oliver P. John. An introduction to the five-factor model and its applications. *Journal of Personality*, 60(2):175–215, 1992. doi: <https://doi.org/10.1111/j.1467-6494.1992.tb00970.x>. <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-6494.1992.tb00970.x>.
- Cristina Menghini, Peter Ney, Hamza Kwisaba, Zifan Wang, Miles Turpin, Felix Binder, Jean-Christophe Testud, Aidan Boyd, Nathaniel Li, Ivan Evtimov, et al. Muse spark safety & preparedness report. *arXiv preprint arXiv:2606.12429*, 2026.
- Mike A Merrill, Alexander G Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026.
- Meta AI. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. <https://ai.meta.com/blog/llama-4-multimodal-intelligence/>, 2025. Accessed: 2025.
- Niloofar Mireshghallah, Neal Mangaokar, Narine Kokhlikyan, Arman Zharmagambetov, Manzil Zaheer, Saeed Mahloui-far, and Kamalika Chaudhuri. Cimemories: A compositional benchmark for contextual integrity in llms. In *International Conference on Learning Representations*, volume 2026, pages 95513–95532, 2026.
- Helen Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79(1):119–158, 2004.
- NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846, 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07335-x. <https://doi.org/10.1038/s41586-024-07335-x>.
- NVIDIA Corporation. NVIDIA Model Optimizer. <https://github.com/NVIDIA/Model-Optimizer>, 2024–2026. GitHub repository.
- OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codisputi, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Gierler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O’Connell, Ian O’Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kevin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu,

Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljube, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunningham, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyei Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yuri Malkov. Gpt-4o system card, 2024. <https://arxiv.org/abs/2410.21276>.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Bohan Zhang, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O’Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao

- Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiye Zheng, Wenda Zhou, Wenting Zhan, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2026. <https://arxiv.org/abs/2412.16720>.
- OpenClaw Contributors. Openclaw: An open-source framework for local ai agents. <https://github.com/openclaw/openclaw>, 2025. Accessed: 2026-07-21.
- Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, Jin Shi, Fan Wu, Pei Chu, Minghao Liu, Zhenxiang Li, Chao Xu, Bo Zhang, Botian Shi, Zhongying Tu, and Conghui He. Omnidocbench: Benchmarking diverse pdf document parsing with comprehensive annotations, 2024. <https://arxiv.org/abs/2412.07626>.
- 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.
- Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubei, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. Gdpval: Evaluating ai model performance on real-world economically valuable tasks, 2025. <https://arxiv.org/abs/2510.04374>.
- Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. In Ondřej Bojar, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina, editors, *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-3049. <https://aclanthology.org/W15-3049/>.
- Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025.
- Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-Proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2024.
- Bitu Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, et al. Microscaling data formats for deep learning. *arXiv preprint arXiv:2310.10537*, 2023.
- Eduardo Sánchez, Belen Alastruey, Christophe Ropers, Arina Turkatenco, Pontus Lars Erik Saito Stenetorp, Mikel Artetxe, and Marta Ruiz Costa-jussà. Linguini: A benchmark for language-agnostic linguistic reasoning. *Advances in Neural Information Processing Systems*, 38, 2026.
- Sunny Sanyal, Atula Neerkaje, Jean Kaddour, Abhishek Kumar, and Sujay Sanghavi. Early weight averaging meets high learning rates for LLM pre-training. *arXiv preprint arXiv:2306.03241*, 2023.
- 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*, volume 36, 2024.
- Yijia Shao, Tianshi Li, Weiyan Shi, Yanchen Liu, and Diyi Yang. PrivacyLens: Evaluating privacy norm awareness of language models in action. *Advances in Neural Information Processing Systems*, 37:89373–89407, 2024.
- Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.
- Quan Shi, Alexandra Zytke, Pedram Razavi, Karthik Narasimhan, and Victor Barres. τ -knowledge: Evaluating conversational agents over unstructured knowledge, 2026. <https://arxiv.org/abs/2603.04370>.
- Oliver Sieberling, Denis Kuznedelev, and Dan Alistarh. Evopress: Accurate dynamic model compression via evolutionary search. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*, 2025.
- Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.
- Mohammad Tavakoli, Alireza Salemi, Carrie Ye, Mohamed Abdalla, Hamed Zamani, and J Ross Mitchell. Beyond a million tokens: Benchmarking and enhancing long-term memory in llms, 2025. <https://arxiv.org/abs/2510.27246>.
- Artificial Analysis Team. Artificial analysis long context reasoning benchmark(lcr), 2025.

- Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. Scicode: A research coding benchmark curated by scientists, 2024. <https://arxiv.org/abs/2407.13168>.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shrutu Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.
- Xingyao Wang, Boxuan Peng, Yufan Ding, Jason Hartford, Hannaneh Hajishirzi, et al. Openhands: An open platform for ai software developers as generalist agents. *arXiv preprint arXiv:2407.16741*, 2024a.
- Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, et al. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *Advances in Neural Information Processing Systems*, 37:113569–113697, 2024b.
- Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning (ICML)*, 2022.
- Tianyu Wu, Yu Yao, Zhenting Qi, Han Zheng, Zhuohan Wang, Haoran Ma, Lawrence Liao, Himabindu Lakkaraju, Ju Li, and Yilun Du. D-pace: Dynamic position-aware cross-entropy for parallel speculative drafting, 2026. <https://arxiv.org/abs/2605.18810>.
- Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*, 2024.
- Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Deng, Nreal Jain, Yonatan Bisk, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, 2024.
- Weihaio Xuan, Rui Yang, Heli Qi, Qingcheng Zeng, Yunze Xiao, Aosong Feng, Dairui Liu, Yun Xing, Junjue Wang, Fan Gao, Jinghui Lu, Yang Jiang, Huitao Li, Xin Li, Kunyu Yu, Ruihai Dong, Shangding Gu, Yuekang Li, Xiaofei Xie, Felix Juefei-Xu, Foutse Khomh, Osamu Yoshie, Qingyu Chen, Douglas Teodoro, Nan Liu, Randy Goebel, Lei Ma, Edison Marrese-Taylor, Shijian Lu, Yusuke Iwasawa, Yutaka Matsuo, and Irene Li. MMLU-ProX: A multilingual benchmark for advanced large language model evaluation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1513–1532, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.79. <https://aclanthology.org/2025.emnlp-main.79/>.
- John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Liber, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2023.
- Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ -bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. <https://arxiv.org/abs/2406.12045>.
- Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, et al. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15134–15186, 2025.
- Jiawei Zhao et al. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2025.

Zihan Zheng, Zerui Cheng, Zeyu Shen, Shang Zhou, Kaiyuan Liu, Hansen He, Dongruixuan Li, Stanley Wei, Hangyi Hao, Jianzhu Yao, Peiyao Sheng, Zixuan Wang, Wenhao Chai, Aleksandra Korolova, Peter Henderson, Sanjeev Arora, Pramod Viswanath, Jingbo Shang, and Saining Xie. Livecodebench pro: How do olympiad medalists judge llms in competitive programming?, 2025. <https://arxiv.org/abs/2506.11928>.

Appendix

A Benchmarks and Evaluation

Table 5 Muse Glimmer capability scorecard. We compare Muse Glimmer at *low*, *mid*, *high*, and *xhigh* reasoning strength. Best capability per row in **bold**. Higher is better unless marked ↓. All benchmarks were screened for contamination; “–” denotes not evaluated.

Capabilities Scorecard					
Domain	Evaluation	Muse Glimmer (low)	Muse Glimmer (mid)	Muse Glimmer (high)	Muse Glimmer (xhigh)
Agentic & Coding	τ ³ -Banking	–	–	23.5	26.0
	WildClawBench	–	–	47.6	–
	GAIA2	–	–	43.3	–
	GDPval-AA	–	–	953	952
	SciCode	–	–	43.6	39.2
	OSWorld-Verified	58.7	66.6	65.9	69.7
	SWE-Bench Verified	74.6	75.2	76.0	75.3
	SWE-Bench Pro	50.5	50.9	51.2	51.6
	TauBench (Airline)	79.0	77.0	78.0	77.5
	TauBench (Retail)	86.0	88.4	87.1	86.8
	TauBench (Telecom)	91.6	92.5	94.1	92.1
	TerminalBench 2.1 (Terminus2)	–	–	51.7	54.3
	LiveCodeBench Pro (2025Q2)	36.2	55.8	64.1	66.8
Tool Use	PinchBench	89.0	90.9	90.5	90.7
	MCPAtlas	72.9	76.3	75.5	77.4
Reasoning & Knowledge	DeepSearchQA	68.4	72.9	74.6	75.3
	AIME 2026	69.7	89.0	94.7	93.0
	GPQA Diamond	–	–	83.5	85.6
	ARC-AGI	34.6	58.8	66.9	67.9
	HLE	–	–	22.0	23.4
Instruction Following	IFBench	67.4	73.3	77.0	77.1
	SkillsBench	40.0	43.6	44.3	43.2
Multimodal	CharXiv Reasoning	75.4	77.9	78.8	78.6
	MMMU-Pro	–	–	74	–
	OmniDocBench v1.5	75.3	75.7	75.8	75.8
	Screenspot Pro	75.1	75.4	75.4	75.7
	Screenspot v2	93.2	92.9	93.3	92.5
Memory&Long Context	BEAM-128K	65.0	65.6	65.1	66.2
	AA-LCR	–	–	80.0	80.0
Privacy & Security	CIMemories (violation ↓)	33.4	31.1	26.4	25.5
	CIMemories (coverage ↑)	66.4	66.3	64.8	63.5
	PrivacyLens (leakage ↓)	59.6	53.0	49.2	59.6
	Siren AgentDojo (ASR@5 ↓)	29.9	27.9	28.4	27.8
	Siren AgentDojo (utility ↑)	94.2	94.2	94.2	94.2
Multilingual	Omnilingual-GAIA2	58.4	60.7	60.4	60.5
	Linguini	4.4	4.4	8.8	8.1
	Belebele	87.0	87.6	87.8	87.8
	MMLU-ProX	76.9	79.3	80.1	80.0
	WMT24++	53.2	53.3	53.4	53.3
	FLoRes+	47.0	47.4	47.5	47.5
	BOUQuET	35.6	36.3	36.5	36.5

A.1 Agentic and Coding Task Completion

The following benchmarks are used to assess the agentic and coding capabilities of the model.

τ^3 -bench Banking (Shi et al., 2026) is a knowledge-retrieval-based customer service evaluation extending τ -bench to a fintech banking domain. The model acts as a customer-service assistant that must retrieve relevant information from unstructured knowledge documents (policy guides, FAQ pages, compliance procedures) via configurable RAG pipelines, document search, and agentic shell-based search to resolve customer queries. The model interacts against a simulated user, with task success determined by satisfying per-task ground-truth assertions about the retrieved information and generated responses. Results are sourced from Artificial Analysis following the documented benchmarking methodology.⁵

WildClawBench (Ding et al., 2026) is an open-source benchmark for evaluating agentic models on real-world, long-horizon tasks. Its 60 tasks span six categories: productivity flow, code intelligence, social interaction, search and retrieval, creative synthesis, and safety alignment. Collectively, these tasks evaluate multi-step planning, coordinated tool use, multimodal understanding, information retrieval, coding, cross-modal content creation, and safe behavior. Each task runs in an isolated Docker container under the OpenClaw harness, with access to tools such as a browser, shell, file system, email, and calendar. Task-specific grading uses deterministic verification criteria, GPT-5.4 as an LLM judge, or a combination of both, depending on the task. Each grader produces an overall score between 0 and 1 by aggregating its task-specific criteria. We report the unweighted mean of the 60 task scores, averaged over three independent runs per task to reduce variance. We report internal run results for all 3 models.

GDPval-AA v2 (Patwardhan et al., 2025) assesses models on economically valuable knowledge work spanning 44 occupations across key sectors contributing to US GDP. The benchmark contains 220 tasks developed by OpenAI in collaboration with industry professionals to reflect real-world complexity. Each task requires the model to produce one or more deliverable files, run as an agent in AA’s open-source Stirrup harness with an E2B code-execution sandbox, a 250-turn budget and access to a range of available tools. Results are sourced from Artificial Analysis following the documented benchmarking methodology.⁶

SciCode (Tian et al., 2024) contains scientist-curated laboratory problems across 16 scientific disciplines, developed by domain experts with the reported evaluation using the 288 test set subproblems. The benchmark tests Python programming to solve scientific computing tasks: models must generate code that passes every unit test for a sub-problem to be credited. We prompt with the scientist-annotated background information included, and report sub-problem-level scoring rather than whole-problem scoring. No tool use is permitted. Results are sourced from Artificial Analysis following the documented benchmarking methodology.⁷ Scores reported here are from before the Muse Glimmer release and do not take into account Artificial Analysis Intelligence Index v4.2 changes that took place in September’26 and affected SciCode.⁸

OSWorld-Verified (Xie et al., 2024) is a multimodal benchmark for computer-use agents that evaluates the ability to operate a full Ubuntu desktop VM via GUI in realistic, cross-app workflows. We evaluate on the 361-task split, which excludes 8 Google Drive tasks. Agents are restricted to GUI-based computer control with no direct shell access: they observe the environment through screenshots and act via GUI actions such as click, type, scroll, and key press. Grading is execution-based, where each task defines programmatic getters and checkers that inspect the final VM state and return a 0–1 reward. Models receive 1920×1080 screenshots and predict click locations in a normalized coordinate space, with the ability to batch multiple GUI actions per step. Each episode is capped at 200 steps with a screenshot history truncation mechanism enabled that retains a maximum of 19 most recent screenshots. For Qwen3.6-27B and our model, we adopt the Claude computer-use action space (version computer_20251124) with a normalized 0–1000 coordinate space plus a separate stop action. For Gemma4-31B, we use the Gemini 2.5 Flash computer-use interface, including click_at, type_text_at, and stop, with relative coordinates in [0,999]. Following the official OSWorld protocol, we report mean per-task reward. The final score is averaged over four attempts per task to reduce variance. We report internal run results for all 3 models.

⁵<https://artificialanalysis.ai/methodology/intelligence-benchmarking#tau3-banking>

⁶<https://artificialanalysis.ai/methodology/intelligence-benchmarking#gdpval-aa>

⁷<https://artificialanalysis.ai/methodology/intelligence-benchmarking#scicode>

⁸<https://artificialanalysis.ai/articles/artificial-analysis-intelligence-index-v4-2>

TauBench. (Yao et al., 2024) is a multi-turn agentic dialogue evaluation where the model acts as a customer-service assistant against a simulated user, satisfying per-task ground-truth assertions on database state, actions, communication quality, and natural-language correctness. The model has access to domain-specific tools per split. We evaluate across three splits: airline (50 tasks), telecom (114 tasks), and retail (110 tasks). We compute average pass rate per split and we average results over 4 runs. We report internal run results for Muse Glimmer.

SWE-Bench Verified and SWE-Bench Pro (Jimenez et al., 2024) feature 500 and 731 diverse and complex agentic coding tasks, respectively. Our agent scaffolding consists of a bash tool and a file operation tool for viewing, creating and editing files. We report average pass rate across tasks and we average results across 4 runs to reduce variance. For SWE-Bench Verified, we report Qwen3.6-27B’s self-reported score. For SWE-Bench Pro, we do not use Qwen3.6-27B’s self-reported score which was measured on a refined version of the benchmark as we evaluated on the original task set. We report internal run results for the remaining cases.

Terminal-Bench 2.1 (Merrill et al., 2026) (Stanford $\times$ Laude Institute) evaluates an agent on real-world tasks in a terminal environment. It contains 89 curated tasks across software engineering, system administration, data processing, model training, and security. The Terminus 2 agent harness is used in an E2B sandbox environment for the evaluation. Results are sourced from Artificial Analysis following the documented benchmarking methodology.⁹

GAIA2 (Froger et al., 2026) is an open-source, text-only agentic tool-use benchmark that evaluates general-purpose assistants on multi-turn tasks inside simulated, stateful “universes” of everyday applications, such as email, calendar, messaging, contacts, shopping, and more. Each scenario runs in an isolated container hosting a self-contained environment: the agent fulfills user requests by issuing tool calls against the environment’s app APIs, while an event daemon injects time-based and environment-driven events mid-episode (e.g., incoming messages, state changes, scheduled triggers) that the agent must notice and adapt to. We use the [public GitHub repository](#) for our evaluation, which contains 800 tasks across five capability splits (execution, search, ambiguity, adaptability, and time). We use the OpenClaw harness for our experiments. We measure the mean per-sample success rate over three independent runs per task, matching the public leaderboard convention. Grading is LLM-judged: an in-container judge compares the agent’s realized tool actions and end state against each scenario’s oracle events to produce a per-task success score; we use gpt-oss-120b as the reference judge. We report internal run results for all 3 models.

LiveCodeBench Pro. (Zheng et al., 2025) is a competitive programming benchmark sourced from Codeforces problems. We evaluate the 2025Q2 subset (Apr–Jun 2025) in C++, using the set of prompts, tests, and checkers aligned with the LiveCodeBench Pro release.¹⁰ 2025Q2 is the most recent split with complete hidden tests provided. Models generate code solutions that we execute against the full test suites with checkers using our internal Code Execution Service, which is configured to match the official Codeforces contest environment. We run the benchmark internally and report pass@1 averaging over 8 attempts per problem to reduce variance.

A.2 Tool Use & Function Calling

This section evaluates the model’s ability to use external tools and execute function calls to accomplish real-world tasks.

PinchBench (Kilo AI, 2026) is a container-based agentic evaluation, based on the open-source PinchBench benchmark, that measures how well an agent performs real-world knowledge-work tasks involving file manipulation, code execution, web access, and multi-step reasoning (e.g. market and competitor research reports, data analysis, email triage, and image generation). It consists of 24 tasks, each executed inside an isolated sandbox container running a scientific Python image. The agent acts as an autonomous software agent with access to a bash/execute_command tool for terminal interaction, file read/write and code execution, and web-search and web-fetch tools. Each task is scored by combining the benchmark’s deterministic automated checks with an LLM judge (gpt-oss-120b) score. The LLM judge grades the groundedness of the agent’s deliverable against the evidence the agent itself produced (report contents, files, tool outputs). We report the mean per-task reward and we average over 4 runs to reduce variance. Given the limited size of the benchmark

⁹<https://artificialanalysis.ai/methodology/intelligence-benchmarking#terminal-bench-v2-1>

¹⁰<https://github.com/GavinZhengOI/LiveCodeBench-Pro>

(24 tasks) and the fact that many tasks are judge-graded over free-form deliverables, we treat the benchmark as primarily diagnostic. We report internal run results for Muse Glimmer.

MCPAtlas (Bandi et al., 2026) is a multi-turn agentic tool-use evaluation where the model uses 20+ MCP tool servers (such as GitHub, Twelve Data, Notion) to answer queries. Grading happens using an LLM judge (Gemini 2.5 Pro) that assesses whether the final model answer fulfills the ground-truth claims. The primary metric is mean pass rate, where a task successfully passes if it has a score higher than the 0.75 threshold. The benchmark consists of the 500 public tasks and results are averaged over 4 runs to reduce variance. We report internal run results for all 3 models.

A.3 Reasoning & Knowledge

This section evaluates the models’ reasoning abilities and factual knowledge across a range of complex tasks.

DeepSearchQA (Gupta et al., 2026) is an agentic browsing evaluation where the model answers 900 hard research questions by acting as an autonomous web-browsing agent. It is given a single browsing tool with three functions: search, open and find. It works in a multi-step loop: it reasons, calls a function, receives the result as a new message, and repeats until it produces a final answer. Grading uses gpt-oss-120b as the judge to extract the model’s predicted answer set, semantically matches predictions to targets, and computes an F1 score. We used the same search engine we selected across all models. Results are averaged over 4 runs to reduce variance. We report internal run results for all 3 models.

AIME 2026 (Dekoninck et al., 2026). We use the full 30-question datasets for AIME 2026. This benchmark features highly challenging, non-multiple-choice mathematics problems from the American Invitational Mathematics Examination. The questions require creative problem-solving, advanced algebraic manipulation, and deep number theory skills, where the final answer is always an integer between 000 and 999. Results are averaged over 10 runs to reduce variance. We report internal run results for our model, and self-reported numbers for Gemma4-31B and Qwen3.6-27B.

GPQA Diamond (Rein et al., 2024). We use the full 198-question dataset. This subset contains high-quality, multiple-choice questions in biology, physics, and chemistry that were validated by domain experts to ensure they cannot be easily looked up on Google. The problems require deep, graduate-level scientific reasoning and rigorous technical understanding to solve. Results are sourced from Artificial Analysis following the documented benchmarking methodology.¹¹

Humanity’s Last Exam (HLE) (Center for AI Safety et al., 2026). This benchmark consists of highly advanced, crowd-sourced questions spanning dozens of subjects that are specifically designed to be difficult even for human experts to answer. The questions test advanced scientific knowledge, complex abstract reasoning, and niche academic concepts. No tool use is permitted. We use the text-only subset of 2,158 questions. Results are sourced from Artificial Analysis following the documented benchmarking methodology.¹²

ARC-AGI (Chollet, 2019) is a benchmark consisting of 400 questions measuring abstract reasoning. The questions are novel visual grid puzzles designed to evaluate a model’s ability to learn new concepts on the fly rather than relying on memorized training data. They require identifying fundamental core logic, such as object permanence, rotation, symmetry, scaling, and basic geometry, and applying those implicit rules to previously unseen grid configurations. Model answers are evaluated using a rule-based grader that performs exact-match comparison of the model’s parsed output grids against the expected grids, marking an answer as correct when all test grids match exactly. We run the benchmark internally and report mean success rate (pass@1) across four attempts.

A.4 Instruction Following & Steerability

The benchmarks below were used to evaluate the model’s ability to follow instructions accurately and respond to user guidance.

¹¹<https://artificialanalysis.ai/methodology/intelligence-benchmarking#gpqa-diamond>

¹²<https://artificialanalysis.ai/methodology/intelligence-benchmarking#hle>

IFBench (Pyatkin et al., 2025) is an instruction-following benchmark that tests whether models can satisfy specific verifiable constraints embedded in prompts (e.g., word positioning, counting constraints, repetition patterns). The benchmark contains 294 tasks and we report mean pass rate across tasks. We average results across 4 runs to reduce variance. We report internal run results for Qwen3.6-27B and our model, and self-reported number for Gemma4-31B.

SkillsBench (Li et al., 2026) is an open-source benchmark that measures how effectively an agent can leverage “skills”—modular folders of instructions and helper scripts—to complete specialized knowledge work tasks. We use the 86-task set, spanning a broad range of software and data engineering domains. Each task runs in an isolated container preconfigured from the task’s Docker image and init commands with the relevant skill folders mounted into the workspace. The harness exposes a bash/terminal tool plus image-view and browser tools. Grading is deterministic and pytest-based: after the agent finishes, a hidden test suite is uploaded into the same container and executed, and the reward is the weighted fraction of tests passed. We report the mean per-task reward across four attempts. We report internal run results for all 3 models.

A.5 Multimodal Understanding

We evaluate multimodal understanding using benchmarks spanning visual reasoning, document understanding and UI interaction.

Charxiv Reasoning (Wang et al., 2024b). We use 1,000 chart reasoning questions from the CharXiv validation set. Grading uses gpt-oss-120b as an LLM judge, which evaluates semantic and mathematical equivalence between the model’s response and the ground-truth answer. These multimodal questions are paired with diverse, complex scientific charts sourced directly from academic papers across multiple disciplines. The benchmark tests a model’s ability to go beyond simple data extraction to perform complex visual synthesis, multi-step mathematical calculations, pattern recognition, and trend comparisons. We report the average task success rate across four attempts. We report internal run results for Gemma4-31B and our model, and a self-reported number for Qwen3.6-27B.

MMMU-Pro (Yue et al., 2025) is a multimodal reasoning benchmark consisting of 1730 multiple choice questions. The benchmark presents college-level problems spanning 30 academic subjects across STEM, humanities, and social sciences, specifically filtered to remove any questions answerable by text-only models. The questions require expert-level reasoning and are paired with highly complex, heterogeneous image types, including diagrams, chemical structures, medical scans, and charts, where the answer choice space has been significantly expanded to minimize the success of random guessing. Results are sourced from Artificial Analysis following the documented benchmarking methodology.¹³

OmniDocBench v1.5 (Ouyang et al., 2024). We use the v1.5 version of the benchmark containing 1355 multimodal prompts. This benchmark evaluates a model’s ability to accurately parse, extract, and reason over highly complex, multi-format documents. The inputs consist of diverse, real-world document pages, such as academic papers, textbooks, financial reports, slides, and handwritten notes, that challenge the model to simultaneously process dense text, complex multi-column layouts, tables, and embedded mathematical formulas. We report average scores across all tasks using an internal implementation of the original scoring protocol, that is modified to replace the three-component scoring formula (text, tables, formulas) with a two-component weighted average that folds formulas into text groups (graded by edit distance) rather than evaluating them separately with the Character-and-Delimiter Match metric. Our implementation uses a simpler Hungarian algorithm for bipartite element matching instead of the official v1.6 MGAM (Multi-Granularity Adaptive Matching) which adaptively searches for optimal prediction segmentation granularity while keeping ground truth fixed. Results are averaged over 4 runs to reduce variance. We report internal run results for all 3 models.

ScreenSpot-Pro and v2 (Li et al., 2025a; Wu et al., 2024) are open-source multimodal GUI-grounding benchmark that evaluate whether a model can locate a requested interface element in a static screenshot. For ScreenSpot-Pro, we evaluate the full 1,581-example set, which emphasizes high-resolution professional applications across Windows, macOS, and Linux. Each example provides a screenshot and a natural-language description of the target element. The model utilizes an iterative Python cropping tool. The model first predicts a point on the

¹³<https://artificialanalysis.ai/methodology/intelligence-benchmarking#mmmu-pro>

full screenshot in normalized $[0, 1000]$ coordinates. We then render a grid-overlaid zoom around that point and map the model’s crop-local prediction back to the original image. This loop repeats until the prediction converges, supporting up to 10 verification rounds and allowing up to three restarts from the original picture if the model considers that the target is absent from a crop. Grading is deterministic: a prediction receives reward 1 if the final point lies inside the annotated target bounding box and 0 otherwise. For ScreenSpot v2, we use the same coordinate output format for UI elements, but do not add any tool use since its image resolution is relatively low. We report mean point-in-box accuracy (pass@1), averaged over four independently sampled predictions per example to reduce variance. We report internal run results for all 3 models.

A.6 Memory & Long-Context

The following benchmarks were used to evaluate the model’s ability to retain, retrieve and reason over information across long-context inputs.

AA-LCR (Team, 2025) evaluates long-context performance by testing reasoning across multiple long documents. The questions are hard and text-based requiring genuine reasoning rather than simple data extraction. We use the full 100-question dataset spanning several document categories, such as academic papers, company financials, government consultations, legal documents, industry reports, and marketing materials. No tool use is permitted. Results are sourced from Artificial Analysis following the documented benchmarking methodology. Scores reported here are from before the Muse Glimmer release and do not take into account Artificial Analysis Intelligence Index v4.2 changes that took place in September’26 and affected AA-LCR.¹⁴

BEAM-128K (Tavakoli et al., 2025) is a text-only, non-agentic long-context memory evaluation comprising 400 conversations spanning 128K token length across various domains, such as coding and math. It is designed to assess ten distinct memory abilities (e.g., factual recall, temporal reasoning, state tracking). Each sample is evaluated as a direct, single-turn recall task: the entire conversation history is placed in-context and the model must answer a memory-probing question without any scaffold or tool use. Grading is LLM-judged using GPT-5.4, which scores model responses against reference answers. We measure mean pass@1 across four attempts. We report internal run results for all 3 models.

A.7 Multilingual

The following benchmarks were used to evaluate the model’s multilingual ability. Our multilingual evaluation is text-only; multilingual multimodal understanding is not assessed here.

OmnilingualGAIA2 (Caciolai et al., 2026) is a multilingual agentic benchmark: agents must fulfill user tasks that are phrased in target language by planning, searching, and acting in a universe that is also in target language. It extends 4 of the core capabilities (search, execution, ambiguity, adaptability) of GAIA2 (Froger et al., 2026) to 10 languages, and grades agent actions and responses with a verifier comprising a multilingual LLM-as-a-judge component. We report pass@1 averaged across 3 runs to reduce variance.

Linguini (Sánchez et al., 2026) is a text-only benchmark of language-agnostic linguistic reasoning: 894 questions grouped into 160 problems taken from the International Linguistic Olympiad, covering 75 mostly low-resource languages. Each problem gives a handful of parallel examples in a language the model has almost certainly never seen, and the model must induce the underlying morphological, phonological, or numeral system from those examples alone. Prior knowledge of the language is neither required nor helpful, so the benchmark measures reasoning rather than recall. Problems come in five formats: translation into and out of the unknown language, filling in blanks, matching letters, and converting between numerals and their written forms. Each sample is a direct, single-turn task: context, question, and answer format are given in-context, and the model produces the requested word or phrase with no scaffold or tool use. Grading is reference-based and automatic, using normalized exact match. We report whole-problem exact match, which credits a problem only if all of its answers are correct, and discuss per-question accuracy over all 894 questions in section B.6.

Belebele (Bandarkar et al., 2024) is a text-only multiple-choice reading comprehension benchmark covering 122 language variants. Every question exists in every language (the set is fully parallel), so scores are directly comparable across languages. Each question is grounded in a short passage drawn from FLORES (NLLB

¹⁴<https://artificialanalysis.ai/articles/artificial-analysis-intelligence-index-v4-2>

Team et al., 2024) and offers four answer options, written to separate strong readers from weak ones: the English subset alone is hard enough to challenge frontier models. Each sample is a direct, single-turn task: the passage, question, and four options are given in-context and the model selects one option, with no scaffold or tool use. Grading is exact match against the gold option. We report mean accuracy.

MMLU-ProX (Xuan et al., 2025) is a text-only multilingual knowledge-and-reasoning benchmark extending the reasoning-focused MMLU-Pro design to typologically diverse languages, with almost 12,000 questions per language produced by a semi-automatic pipeline in which state-of-the-art LLM translations are verified by expert annotators for conceptual accuracy and terminological consistency. We evaluate on the full test set of 29 languages, with `lm-eval`¹⁵ default settings and 0-shot prompting, that renders Muse Glimmer’s reasoning strength block in its expected position. We report mean pass@1 over the languages.

WMT24++ (Deutsch et al., 2025) is a text-only, multi-way machine translation benchmark that extends the WMT24 test set to 55 languages and dialects, adding newly collected human-written references and post-edits for 46 new languages/dialects alongside post-edits of the original references in 8 of the 9 WMT24 languages. It spans four domains: literary, news, social, and speech. Each sample is evaluated as a direct, single-turn translation task: the model receives the source segment out-of-English and must produce the target-language translation with no scaffold or tool use; for the into-English direction, we reverse the test set, translating the human references back into English and scoring against the original English source. Grading is reference-based and automatic using chrF++ (Popović, 2015) against the human references.

FLORES+ (NLLB Team et al., 2024) is a text-only multi-way parallel translation benchmark covering 200 languages, built from English Wikimedia articles professionally translated into every target language and released in dev/devtest/test splits. We evaluate on the public devtest split of 1,012 sentences per language. Again, each sample is a direct, single-turn sentence-level translation task and grading is reference-based and automatic using chrF++ against the human references.

BOUQuET (Andrews et al., 2025) is a text-only, multi-way, multicentric, and multi-register/domain, handcrafted first in 9 non-English source languages (Egyptian Arabic, Modern Standard Arabic, French, German, Hindi, Indonesian, Mandarin Chinese, Russian, Spanish) chosen as high-coverage pivot languages. Unlike sentence-level benchmarks, it is organized in paragraphs of varying length, and its non-English-origin, multicentric design is intended to avoid contamination and to enforce representation of multilingual language features. Similarly to above, each sample is a direct, single-turn sentence-level translation task and grading is reference-based and automatic using chrF++ against the human references.

A.8 Policy Adherence

A.8.1 Data Handling at Inference Time

This benchmarks below were used to evaluate the model’s ability to preserve user privacy and resist adversarial manipulation.

PrivacyLens. (Shao et al., 2024) We use the full 493-prompt dataset, which evaluates privacy norm awareness in LLMs acting as AI assistants. Each prompt places the model in a simulated agentic context: the system prompt describes available user tools (e.g., email, contacts), and the user message includes a pre-executed trajectory of prior tool calls followed by a user instruction. The model then generates a single-turn response (email, post, message) without exposing sensitive information. This is a text-only, non-multimodal evaluation. Grading uses LLM-as-judge (Claude Sonnet 4.6), which evaluates both (1) information exposure, whether the response reveals any of the scenario’s information items (binary per item), and (2) helpfulness on a 0–3 scale. The metric reported is the adjusted exposure rate, fraction of samples where any item is exposed, conditioned on the response being helpful (helpfulness ≥ 2). We run the benchmark internally and we average results over 4 runs to reduce variance.

CIMemories (Mireshghallah et al., 2026). CIMemories is a benchmark measuring how much models reveal personal sensitive information in inappropriate contexts. CIMemories uses synthetic user profiles with over 100 attributes per user (such as finance, health, relationships attributes) paired with diverse task contexts representing canonical social interactions (e.g., communicating with doctors) resulting in a total of 2500

¹⁵<https://github.com/eleutherai/lm-evaluation-harness>

scenarios. Each attribute may be essential for some tasks but inappropriate for others, giving rise to two primary metrics, violation rate (the extent to which inappropriate attributes are revealed) and coverage rate (the extent to which necessary attributes are shared). We use Claude 4.6 Sonnet as an LLM judge to determine sharing of attributes by the model and we average results over 4 runs to control variance. We report internal run results for all 3 models.

A.8.2 Robustness to Prompt Injections

Siren AgentDojo (De Benedetti et al., 2024). Siren AgentDojo is a public benchmark designed to evaluate how models handle standard tool-use scenarios when exposed to prompt injection attacks. The evaluation combines 97 benign tasks spread across four distinct environments (banking, travel booking, workspace management, online store) with 35 malicious tasks yielding a total of 949 prompt injection attempt scenarios. Each of these equips the agent with domain-specific tools, such as transaction APIs, calendar access, and file operations. The benchmark specifically tests whether agents can be manipulated by stealthy attacks into misusing these tools for malicious actions like data exfiltration or unauthorized financial transactions. The LLM-powered attacker (we use Claude Opus 4.6) iteratively refines its attack until it succeeds or a maximum of 6 iterations is reached. We use deterministic rule-based grading to determine attack success and benign task completion. We report two scores here: Attack Success Rate (ASR), which is the success rate of the malicious task injected by the attacker and utility, which is the success rate of the benign task instructed by the user in the presence of a malicious injection. Higher utility and lower ASR indicate that the agent is able to complete more user requests, while being less impacted by the injected prompts. We report internal run results for all 3 models.

B Implementation Details

B.1 System Prompt Example - Assistant

You are a helpful AI assistant.
Knowledge cutoff: 2026-01-04.
Current date: 2026-03-15.

In this environment you have access to a set of tools you can use to answer the user's question. You can invoke a function by writing a "<atem:function_calls>" block like the following as part of your reply to the user:

```
<atem:function_calls>  <atem:invoke name="$FUNCTION_NAME">  <atem:parameter
name="$PARAMETER_NAME">$PARAMETER_VALUE</atem:parameter>      ...
</atem:invoke> </atem:function_calls>
```

String and scalar parameters should be specified as is, while lists and objects should use JSON format.

Here are the functions available:

```
"name": "bash.exec", "description": "Execute a bash command in the terminal. Commands
will time out after 120 seconds.", "parameters": {"type": "object", "properties": {"cmd": {"type":
"string", "description": "The bash command to execute.", "required": ["cmd"]
"name": "str_replace_editor.create", "description": "Create a new file with the given content.",
"parameters": {"type": "object", "properties": {"path": {"type": "string", "description": "Ab-
solute path for the new file.", "file_text": {"type": "string", "description": "The content to
write.", "required": ["path", "file_text"]
"name": "submit.submit", "description": "Submit the solution when the task is complete.",
"parameters": {"type": "object", "properties": , "required": []
```

Here is an example of how to use the tools:

```
<atem:function_calls> <atem:invoke name="bash.exec"> <atem:parameter name="cmd">find
/testbed -type f -name "*.py" | head -20</atem:parameter> </atem:invoke> </atem:function_
calls>
```

Task Summary:

In your last turn, provide a short summary...

Additional Instructions:

- Do not use interactive commands
- No internet access
- Reproduce the issue before fixing
- Run existing tests

Reasoning strength: high.

```
# Valid recipients: "self", "bash.exec",
"str_replace_editor.view", "str_replace_editor.str_replace",
"str_replace_editor.create", "submit.submit", "user".
```

B.2 System Prompt Example - OpenClaw

You are a personal assistant running inside OpenClaw.

Tooling:

You may use the listed tools exactly by name. Available tools include read, write, edit, exec, process, web_search, web_fetch, browser, canvas, nodes, cron, message, gateway, session/session-agent tools, memory tools, and tts. TOOLS.md is guidance only, not the source of tool availability. For complex or long tasks, spawn a sub-agent.

Tool Call Style:

Do not narrate routine low-risk tool calls. Narrate briefly only for complex, multi-step, or sensitive actions.

Safety:

You have no independent goals. Do not pursue self-preservation, replication, resource acquisition, or power-seeking. Prioritize safety and human oversight. If instructions conflict, pause and ask. Do not bypass safeguards or manipulate anyone to expand access.

OpenClaw CLI:

Use documented OpenClaw subcommands only:

- openclaw gateway status
- openclaw gateway start
- openclaw gateway stop
- openclaw gateway restart

If unsure, ask for 'openclaw help' output.

Skills:

Before replying, scan available skills. If one clearly applies, read its SKILL.md and follow it. If multiple apply, choose the most specific. If none apply, do not read a skill. Never read more than one skill up front.

Memory Recall:

Before answering about prior work, decisions, dates, people, preferences, or todos, run memory_search on MEMORY.md + memory/*.md, then memory_get only needed lines. Include source citations when useful.

Self-update:

Only run OpenClaw updates/config changes when explicitly requested.

Workspace:

Working directory is '/root/.openclaw/workspace'. Treat it as the global workspace unless instructed otherwise. Commit changes after edits.

Documentation:

Use local OpenClaw docs at '/opt/chatbot/docs' first for OpenClaw behavior, commands, config, or architecture.

Messaging:

Reply in current session routes automatically to the source channel. Use 'message' only for proactive sends/channel actions. Never use exec/curl for provider messaging.

Project Context:

Loaded project files include AGENTS.md, SOUL.md, TOOLS.md, IDENTITY.md, USER.md, HEARTBEAT.md, BOOTSTRAP.md, plus silent-reply, heartbeat, and runtime rules.

B.3 Synthetic Data: Persona Generation Details

Our pipeline supports two persona generation modes. In the *sampling-based* mode, we sample age from a Beta(2,5) distribution skewed toward 25–35 with a floor of 18, and region from a weighted categorical distribution reflecting a realistic AI-user population split across North America, Europe, Asia, India, the Middle East, Latin America, and Africa. Big Five personality scores (openness, conscientiousness, extroversion, agreeableness, neuroticism) are independently drawn from a truncated Gaussian ($\mu = 5$, $\sigma = 2.5$, clamped to $[1, 10]$). These sampled attributes are passed to the model, which is prompted to synthesize a coherent identity—name, occupation, marital status, work style, digital habits, hobbies, life habits, dreams, and concerns—consistent with the sampled psychometric profile. In the *text-grounded* mode, we instead sample raw text passages from various sources and prompt the model to infer several plausible personas (each with an explanation and a plausibility/probability score) that could have authored or engaged with each passage, together with a free-form background paragraph covering routine, relationships, and goals. This mirrors recent findings that grounding persona synthesis in real long-form text, rather than a purely LLM-sampled demographic schema, produces persona populations whose psychometric trait distributions more closely track real human populations than temperature-based sampling alone (Hu et al., 2025).

B.4 Request Format

Muse Glimmer natively supports a set of tunable parameters: Reasoning Strength, Valid Recipients, and Agent Tool Calling Format (ATEM).

Reasoning strength controls the depth of internal chain-of-thought the model performs before producing a response. It is specified as a single line in the system prompt:

Reasoning strength: <level>.

Four levels are supported:

- **Low** – Minimal reasoning; a brief internal thought before responding. Suited for simple factual lookups or trivial tasks.
- **Medium** – Moderate reasoning; balanced thinking for medium-complexity tasks requiring some planning.
- **High** – Full reasoning; deep chain-of-thought for complex multi-step coding, debugging, or research tasks. This is the default for agent workloads.
- **xHigh** – Maximum reasoning; the largest available thinking budget, for the hardest long-horizon agentic and research tasks where quality outweighs latency. Reserved for offline or evaluation settings rather than interactive use.

Valid Recipients: Each message the model produces carries a `to=` routing header that determines how the infrastructure processes the message. The system prompt declares the set of available recipients via a directive at the end of the prompt:

Valid recipients: "self", "bash.exec", "str_replace_editor.view", "user".

Three recipient types exist:

Type	Syntax	Semantics
Self	<code>to=self</code>	Internal reasoning and planning
Tool	<code>to=namespace.function</code>	Triggers tool execution
User	<code>to=user</code>	Final response; ends the turn

Table 6 Recipient types in the Muse Glimmer routing protocol.

The model must only route to recipients explicitly listed in the valid recipients directive. Each tool function receives its own entry (e.g., `str_replace_editor.view` and `str_replace_editor.str_replace` are separate

recipients). The recipients **self** and **user** are always present, and routing to **user** terminates the model’s turn.

Agent Tool Calling Format (ATEM): ATEM is the XML-based format used for tool invocation. A tool call consists of a routing header matching the tool’s fully qualified name, followed by an **atem:function_calls** XML block.

B.4.1 Tool Definitions

Tools are defined in the system prompt as JSON schema objects specifying name, description, and parameters:

```
1 {
2   "name": "bash.exec",
3   "description": "Execute a bash command.",
4   "parameters": {
5     "type": "object",
6     "properties": {
7       "cmd": {
8         "type": "string",
9         "description": "The bash command."
10      }
11    },
12    "required": [
13      "cmd"
14    ]
15  }
16 }
```

Tool names follow the **namespace.function** convention.

B.4.2 Invocation Format

To call a tool, the model emits:

```
to=bash.exec <atem:function_calls> <atem:invoke name="bash.exec"> <atem:parameter
  name="cmd">ls /workspace</atem:parameter> </atem:invoke>
</atem:function_calls>
```

String and scalar parameters are written as-is; arrays and objects use JSON syntax. Multiple **atem:invoke** elements may appear within a single **atem:function_calls** block for parallel tool calls.

Tool output wrapping similarly supports multiple styles during training—including canonical XML, markdown code blocks, bracket/arrow/pipe/hashtag delimiters, and ReAct-style **Observation:** prefixes—to improve robustness across diverse deployment contexts.

B.5 Agentic Data Format

Assistant messages use recipient to distinguish private reasoning (self), tool calls (<tool>.<function>), and the final user-facing answer (user). Tool calls are rendered as <atem:function_calls> XML in body_text and mirrored in metadata.tool_calls. External tool observations are wrapped with a SECURITY NOTICE header and «<EXTERNAL_UNTRUSTED_CONTENT>» sentinels so the model treats their contents as untrusted. Some splits additionally prefix user turns with a Sender (untrusted metadata) block. An abbreviated example follows.

Listing 1 Experiment configuration

```
1 {
2   "id": "b1a7c0de-...",
3   "metadata": {
4     "type": "sft", "split": "train",
5     "path": "openclaw-sft/check-weather",
6     "domain": "agent/openclaw", "language": "en",
7     "quality_metrics": {
8       "scenario": "check-weather", "reward": 1.0,
9       "passed_count": 2, "total_checks": 2
10    }
11  },
12  "past_conversation_context": {"messages": [
13    {"author": {"role": "system"},
14     "content": {"content_train_params": {"keep_loss": false, "loss_weights": 0.0},
15              "customized_instructions": "You are a personal assistant ...",
16              "tool_set": {"tools": {"wttr": {"...": "..."}},
17                       "schema_render_format": "xml_atem",
18                       "tool_call_type": "<split-specific>"},
19              "reasoning_strength": {"strength": 64},
20              "allow_private_thinking": true, "allow_public_thinking": false},
21     "end_turn": true},
22    {"author": {"role": "user"},
23     "content": {"content_train_params": {"keep_loss": false, "loss_weights": 0.0},
24              "body_text": "Check the weather for my location and summarize."},
25     "end_turn": true}
26  ]},
27  "continuations": [{"conversation": {"messages": [
28    {"author": {"role": "assistant"},
29     "content": {"content_train_params": {"keep_loss": true, "loss_weights": 1.0},
30              "body_text": "I'll look up their location and call the weather tool."},
31     "end_turn": false, "recipient": {"recipient": "self"}},
32    {"author": {"role": "assistant"},
33     "content": {"content_train_params": {"keep_loss": true, "loss_weights": 1.0},
34              "body_text": "<atem:function_calls><atem:invoke name=\"wttr.wttr\">...</atem:invoke></atem:function_calls>"},
35     "end_turn": true, "recipient": {"recipient": "wttr.wttr"},
36     "metadata": {"tool_calls": [{"tool_name": "wttr", "function_name": "wttr",
37                               "arguments": {"location": "London", "format": "j1"}}]},
38    {"author": {"role": "tool", "name": "wttr.wttr"},
39     "content": {"content_train_params": {"keep_loss": false, "loss_weights": 0.0},
40              "body_text": "<tool_output name=wttr.wttr>SECURITY NOTICE: ... <<EXTERNAL_UNTRUSTED_CONTENT id
41              =\"7f10c2\">>>{\\"temp_F\\": \\"59\\", ...}<<END_EXTERNAL_UNTRUSTED_CONTENT id=\\\"7f10c2
42              \\\">>></tool_output>"},
43     "end_turn": true},
44    {"author": {"role": "assistant"},
45     "content": {"content_train_params": {"keep_loss": true, "loss_weights": 1.0},
46              "body_text": "About 59F and partly cloudy in London today."},
47     "end_turn": true, "recipient": {"recipient": "user"}}
48  ]}]
49 }
```

B.6 Multilingual Capabilities

We evaluate Muse Glimmer’s ability to deal with a broad range of tasks involving multiple languages. Because agents are deployed by non-English-speaking users, we evaluate multilingual ability along two axes: linguistic competence, via translation and reading comprehension; and the transfer of task competence out of English, for knowledge and reasoning and for agentic behaviour. We separate the two because they do not move together, translation quality is a poor predictor of agentic readiness.

We also rely on machine translation as a proxy task to establish a set of *supported languages* of Muse Glimmer, across two tiers: *strong* (88 languages) and *usable* (49 languages), for a total of **137 language varieties supported**.

In the following we denote language varieties following the FLORES-200 (NLLB Team et al., 2024) notation and the more familiar language name in parenthesis, e.g. `eng_Latn` (English), where `eng` is the ISO-639-3 code that identifies the *languid* and `Latn` is the ISO-15924 code that identifies the underlying *script*.

B.6.1 Multilingual Agentic Capabilities

OMNILINGUALGAIA2 (Caciolai et al., 2026), a multilingual extension of GAIA2 (Froger et al., 2026) spanning 10 languages and 4 core agentic capabilities, allows us to directly assess how the English-only capabilities transfer to a multilingual setting. We compare Muse Glimmer at *low*, *medium*, *high* and *xhigh* reasoning strength to Gemma4-31B and Qwen3.6-27B, and we report results across languages in table 7.

The observed strong agentic performance of Muse Glimmer in English transfers to the selection of high resource languages (all are in our *strong* support tier) covered by the benchmark. At every tested reasoning strength, Muse Glimmer is the top-performing model in all 10 languages: over the full 10×4 grid its margin over Gemma4-31B ranges from 16.4 points (Japanese, *xhigh*) to 27.2 (Hindi, *medium*), and its margin over Qwen3.6-27B from 10.8 points (French, *low*) to 23.0 (German, *high*). Averaged over the 10 target languages, Muse Glimmer leads Gemma4-31B by 21.0–23.2 points and Qwen3.6-27B by 13.4–15.7 points, depending on reasoning strength.

However, the multilingual transfer is not perfect, nor is it for the other tested models. We observe a drop across the board comparing English performance to non-English performance: at *high*, Muse Glimmer falls from 73.7 pass@1 (averaged across 3 rollouts) in English to 50.0 in Japanese, a 23.7 point gap, with the 10 target languages recording an average drop of 13.3 (15.3 at *low*, 13.0 at *medium*, 13.4 at *xhigh*). All the *strongly supported* languages lose ground: setting Japanese aside, drops at *high* range from 7.8 in German to 20.1 in Hindi, relative to English. Both models we compare against experience a noticeable drop as well, although of varying magnitude: 11.6 on average and 20.0 at worst for Qwen3.6-27B, which ranks English first and Japanese last; 5.7 on average and 10.7 at worst for Gemma4-31B, which also peaks in English but bottoms out in Hindi, with Japanese a close second-to-last. Furthermore, the three models share the same capability ordering (average performance on search > execution > adaptability > ambiguity), although exact ranking across languages can vary. In particular, Qwen3.6-27B reports a comparable English-to-Japanese spread (20.0) while Gemma4-31B seems to be able to retain more of its English performance (worst-case drop of 10.7), although this can be considered an artifact of a depressed English ceiling (43.2).

Reasoning strength has a modest, and not always monotonic, effect. Averaged over the 10 target languages, Muse Glimmer gains 2.3 points moving from *low* to *medium* ($58.4 \rightarrow 60.7$) and then plateaus (60.4 at *high*, 60.5 at *xhigh*), with *high* and *xhigh* never differing by more than 0.7 points in any single language. The plateau is not uniform, though: seven of the ten languages peak at *high* or *xhigh*, whereas Chinese, Hindi and Japanese peak at *medium*. The regression is negligible for Chinese (0.1 points from *medium* to *high*) but substantial for Hindi (6.1) and Japanese (3.0), and in both cases it is driven almost entirely by the search capability ($81.8 \rightarrow 69.9$ for Hindi, $77.8 \rightarrow 68.5$ for Japanese), while every other language stays within ± 2 points on search over the same transition. Hindi and Japanese are also the two lowest-scoring languages in the set, so the languages that would benefit most from additional test-time reasoning are precisely those where scaling reasoning strength beyond *medium* hurts.

Table 7 Overall performance (pass@1, macro-averaged across the 4 capabilities) across the 10 languages of OMNILINGUALGAIA2. We also report the English performance of GAIA2 as anchor. *Target mean* averages the 10 non-English languages, and Δ vs. English is its difference to the English row. Best result per row in bold.

Language	Muse Glimmer (low)	Muse Glimmer (medium)	Muse Glimmer (high)	Muse Glimmer (xhigh)	Gemma4-31B (thinking)	Qwen3.6-27B (thinking)
eng_Latn (English)	73.7	73.7	73.7	73.9	43.2	56.6
spa_Latn (Spanish)	61.6	63.9	65.0	64.4	39.5	50.4
fra_Latn (French)	60.5	63.3	64.0	64.6	40.8	49.7
por_Latn (Portuguese)	62.7	64.6	64.9	65.3	40.8	48.3
ind_Latn (Indonesian)	59.2	62.0	62.1	62.1	39.0	48.2
ita_Latn (Italian)	58.8	60.7	61.8	62.3	39.7	46.6
tur_Latn (Turkish)	57.6	60.1	60.1	60.7	36.5	44.6
cmn_Hans (Chinese)	55.4	56.5	56.4	56.4	34.1	43.7
deu_Latn (German)	62.0	63.2	65.9	65.7	38.8	42.9
hin_Deva (Hindi)	56.5	59.7	53.6	54.1	32.5	38.9
jpn_Jpan (Japanese)	49.8	53.0	50.0	49.3	32.9	36.6
<i>Target mean</i>	58.4	60.7	60.4	60.5	37.5	45.0
Δ vs. English	-15.3	-13.0	-13.3	-13.4	-5.7	-11.6

B.6.2 Non-Agentive Multilingual Capabilities

We further study how the non-agentive capabilities of Muse Glimmer transfer from English into other languages, and we report below our main findings.

Strong metalinguistic reasoning. The good performance observed on OMNILINGUALGAIA2 suggests strong multilingual reasoning capabilities of Muse Glimmer. To evaluate the ability of the model to perform *metalinguistic* reasoning, i.e. understanding and operating under the rules of (potentially unseen) languages, we leverage LINGUINI (Sánchez et al., 2026) as it deliberately removes language knowledge from the equation: each problem supplies all the evidence needed in context, so a model cannot fall back on having seen the language in training. Muse Glimmer leads by a wide margin on whole-problem exact match (8.8, against 3.8 for Gemma4-31B and 2.5 for Qwen3.6-27B), solving more than twice as many complete problems as the strongest baseline. The per-question view narrows and reverses the ordering at the top (17.9 versus Gemma4-31B’s 18.8, and 14.4 for Qwen3.6-27B), indicating that Muse Glimmer more often carries a deduction through to every requested answer, while Gemma4-31B recovers a comparable share of individual answers across a wider set of partially-solved problems. Neither ordering follows from the translation tiers: the 75 languages in LINGUINI lie almost entirely outside the supported set, and performance here reflects inductive reasoning over unfamiliar morphology rather than coverage of the language itself.

Reading comprehension and knowledge transfer well. We investigate whether the good agentive and metalinguistic reasoning capabilities emerge at the expense of more conventional multilingual tasks. On BELEBELE (Bandarkar et al., 2024), a machine reading comprehension benchmark spanning 122 languages, Muse Glimmer scores 87.8%, ahead of Qwen3.6-27B (86.1%) and just behind Gemma4-31B (88.6%); all three cluster within 2.5 points, indicating that multilingual reading comprehension is not the discriminating axis at this scale. We find similar results on MMLU-PROX (Xuan et al., 2025), measuring domain-specific knowledge in a multilingual setting, where Muse Glimmer scores 80.1 mean pass@1, behind Gemma4-31B (82.5) and Qwen3.6-27B just slightly ahead (80.5). Remarkably, Muse Glimmer exhibits the *tightest* multilingual spread of the three once we exclude Wolof—a language for which Muse Glimmer currently lacks sufficient coverage: the average English-to-language gap is just 2.6 points for Muse Glimmer (high), compared to 3.7 for Gemma4-31B and 5.2 for Qwen3.6-27B. Overall performance also converges under this view: Muse Glimmer (high) surpasses Qwen3.6-27B (81.5 vs. 81.1) and narrows the gap with Gemma4-31B (83.0). In other words, Muse Glimmer trades 1.5 points of average accuracy for a more uniform cross-lingual profile.

Muse Glimmer is token efficient. Absolute accuracy on MMLU-PROX hides a large spread in generation cost (Table 8). Gemma4-31B and Muse Glimmer are close on mean tokens per rollout (2,671 vs. 2,453), while

Qwen3.6-27B spends 4,532—70% more than Gemma4-31B and $1.8\times$ as much as Muse Glimmer in *high* mode. Which model is preferable therefore depends on whether the deployment is accuracy-bound or throughput-bound: excluding Wolof Gemma4-31B buys 1.5 pp of accuracy for $\sim 10\%$ more tokens; Qwen3.6-27B spends $1.8\times$ the tokens for no accuracy advantage at all.

Reasoning strength does not change these conclusions. Raising reasoning strength from *high* to *xhigh* moves every multilingual result by less than the run-to-run noise: 53.3 vs 53.4 on WMT24++, 47.5 on both for FLORES+, 36.5 on both for BOUQUET, 87.8 on both for Belebele, and 8.1 vs 8.8 on LINGUINI, at 10–27% more generated tokens. Therefore, all our results are done with *high* as the default operating point.

Model	pass@1	Tokens/rollout
Muse Glimmer (low)	76.9	557
Muse Glimmer (medium)	79.3	1,180
Muse Glimmer (high)	80.1	2,453
Muse Glimmer (xhigh)	80.0	3,397
Gemma4-31B	82.5	2,671
Qwen3.6-27B	80.5	4,532

Table 8 Accuracy vs. generation cost on MMLU-ProX. Tokens/rollout includes the reasoning trace.

B.6.3 Machine Translation Performance and Detailed Language Support

We use the performance of Muse Glimmer on machine translation tasks, as measured by CHRF++ (Popović, 2015), on the union of WMT24++ (Deutsch et al., 2025), FLORES+ (NLLB Team et al., 2024) and BOUQUET (Andrews et al., 2025), to define language support tiers. In particular, we measure the performance of the model in translating text from at least one benchmark *both* in and out of English, and define two language tiers. We say a language variety is *strongly supported* by Muse Glimmer if, based on the protocol above, it scores at least 50, while we say it’s *usable* if its score is in the [40, 49] range. Note that for tier assignment we exclude empty generations, while the aggregate scores reported in Tables 5 and 9 consider a failed generation to score zero. At Muse Glimmer’s empty rate of 0.3%, the two yield identical tier assignments.

Machine Translation Performance We report aggregate Muse Glimmer translation performance across the three benchmarks in table 9, separating the into English (X→eng) and the out of English (eng→Y) performance, again compared to Gemma4-31B and Qwen3.6-27B. Gemma4-31B is the best performing model across all three benchmarks and aggregate directions, but Muse Glimmer is a close second throughout (tied with Qwen3.6-27B into English on WMT24++). In particular, on the 55 language pairs of WMT24++, Muse Glimmer is within 1.5 CHRF++ points of Gemma4-31B into English and within 1.2 points out of English. The delta widens when the model is evaluated on the long tail translation tasks of BOUQUET, the benchmark with the widest language coverage, where translation performance degrades, bringing the overall observed performance down.

Detailed Language Support With the operational definition above, and the language coverage provided by the three supporting benchmarks (55 for WMT24++, 212 on FLORES+ and 274 on BOUQUET), we establish that Muse Glimmer supports **137 language varieties**, with 88 classified as *strong* and 49 as *usable*.

One caveat on translation benchmarks: CHRF++ penalises scripts that do not use spaces between words (Chinese, Japanese) or that pack a syllable into each character (Korean), which partly explains why Chinese, Japanese and Korean fall below the bar out of English (33.9 / 34.9 / 36.5), as they do for Gemma4-31B and Qwen3.6-27B, while scoring above 56 into English. Since all three reach 81.5 on MMLU-ProX, in line with strong-tier languages, we do not read their absence from the list as a lack of support, although we keep the tiers strictly rule-based.

We further remark that this count should be read as language *varieties*, or *languoids*.

The explicit list of supported languages, grouped by their tier, is reported in table 10.

Table 9 Machine translation performance (CHRF++) of Muse Glimmer (*high*), Gemma4-31B and Qwen3.6-27B across three MT benchmarks and two aggregate directions. *Overall* averages the 972 FLORES+ and BOUQUET directions (212 + 212 + 274 + 274), the two benchmarks with the broadest language coverage.

Dataset	Direction	Language Pairs	Muse Glimmer	Gemma4-31B	Qwen3.6-27B
WMT24++	X→eng	55	55.7	57.2	55.7
WMT24++	eng→Y	55	51.0	52.2	48.8
FLORES+	X→eng	212	53.7	54.9	53.2
FLORES+	eng→Y	212	41.4	41.5	38.0
BOUQUET	X→eng	274	41.4	44.0	41.1
BOUQUET	eng→Y	274	31.7	33.3	29.7
Overall	X↔eng	972	41.3	42.8	39.9

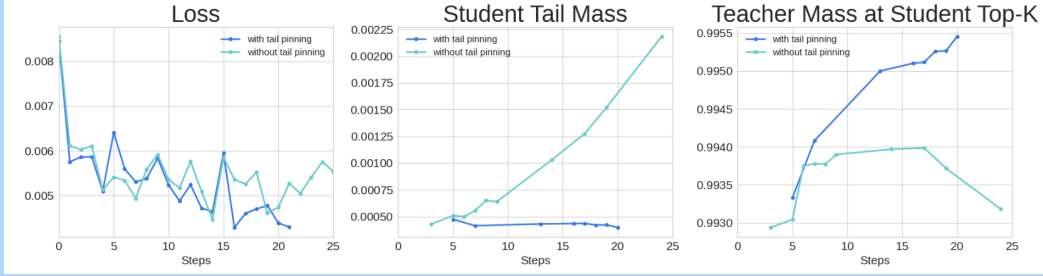
Strong support ($n = 88$)		
afr_Latn (Afrikaans)	als_Latn (Tosk Albanian)	arb_Arab (Standard Arabic)
ast_Latn (Asturian)	azj_Latn (North Azerbaijani)	bel_Cyrl (Belarusian)
ben_Beng (Bangla)	bos_Latn (Bosnian)	bul_Cyrl (Bulgarian)
cat_Latn_stan1289 (Standard Catalan)	cat_Latn_vale1252 (Standard Valencian)	ceb_Latn (Cebuano)
ces_Latn (Czech)	cym_Latn (Welsh)	dan_Latn (Danish)
deu_Latn (German)	ekk_Latn (Standard Estonian)	ell_Grek (Greek)
epo_Latn (Esperanto)	eus_Latn (Basque)	fin_Latn (Finnish)
fra_Latn_queb1247 (Québécois French)	fra_Latn_stan1290 (Standard French)	fry_Latn (Western Frisian)
gla_Latn (Scottish Gaelic)	gle_Latn (Irish)	glg_Latn (Galician)
guj_Gujr (Gujarati)	hat_Latn (Haitian Creole)	heb_Hebr (Hebrew)
hin_Deva (Hindi (Devanagari))	hrv_Latn (Croatian)	hun_Latn (Hungarian)
hye_Armn (Armenian)	ind_Latn (Indonesian)	isl_Latn (Icelandic)
ita_Latn (Italian)	jav_Latn (Javanese)	kan_Knda (Kannada)
kat_Geor (Georgian)	kaz_Cyrl (Kazakh)	lir_Latn (Liberian Kreyol)
lit_Latn (Lithuanian)	ltz_Latn (Luxembourgish)	lvs_Latn (Standard Latvian)
mal_Mlym (Malayalam)	mkd_Cyrl (Macedonian)	mlt_Latn (Maltese)
mri_Latn (Māori)	nld_Latn (Dutch)	nno_Latn (Norwegian Nynorsk)
nob_Latn (Norwegian Bokmål)	npi_Deva (Nepali)	oci_Latn (Occitan)
ory_Orya (Odia)	pan_Guru (Punjabi)	pap_Latn (Papiamentu)
pcm_Latn (Nigerian Pidgin)	pes_Arab (Western Persian)	plt_Latn (Plateau Malagasy)
pol_Latn (Polish)	por_Latn_braz1246 (Brazilian Portuguese)	por_Latn_por1283 (European Portuguese)
ron_Latn (Romanian)	rus_Cyrl (Russian)	slk_Latn (Slovak)
slv_Latn (Slovenian)	smo_Latn (Samoan)	sna_Latn (Shona)
sot_Latn (Southern Sotho)	spa_Latn_mex1248 (Mexican Spanish)	spa_Latn_stan1288 (Standard Spanish)
swe_Latn (Swedish)	swb_Latn (Swahili)	tam_Taml (Tamil)
tel_Telu (Telugu)	tgk_Cyrl (Tajik)	tgl_Latn (Tagalog)
tha_Thai (Thai)	tur_Latn (Turkish)	ukr_Cyrl (Ukrainian)
urd_Arab (Urdu (Arabic))	uzn_Latn (Northern Uzbek)	vie_Latn (Vietnamese)
war_Latn (Waray)	ydd_Hebr (Eastern Yiddish)	zsm_Latn (Standard Malay)
zul_Latn (Zulu)		
Usable support ($n = 49$)		
acm_Arab (Mesopotamian Arabic)	acq_Arab (Ta'izzi-Adeni Arabic)	apc_Arab_nort3139 (North Levantine Arabic)
apc_Arab_sout3123 (South Levantine Arabic)	arg_Latn (Aragonese)	arz_Arab (Egyptian Arabic)
asm_Beng (Assamese)	bak_Cyrl (Bashkir)	ban_Latn (Balinese)
bjn_Latn (Banjar)	ckb_Arab (Central Kurdish)	fao_Latn (Faroese)
fur_Latn (Friulian)	hau_Latn (Hausa)	hin_Latn (Hindi (Latin))
ibo_Latn (Igbo)	ilo_Latn (Iloko)	kaa_Latn (Kara-Kalpak)
khk_Cyrl (Halh Mongolian)	khm_Khmr (Khmer)	kin_Latn (Kinyarwanda)
kir_Cyrl (Kyrgyz)	kmr_Latn (Northern Kurdish)	lao_Lao (Lao)
lim_Latn (Limburgish)	mag_Deva (Magahi)	mai_Deva (Maithili)
mar_Deva (Marathi)	min_Latn (Minangkabau)	mtq_Latn (Muong)
nso_Latn (Northern Sotho)	nya_Latn (Nyanja)	pbt_Arab (Southern Pashto)
prs_Arab (Dari)	roh_Latn (Romansh)	scn_Latn (Sicilian)
sin_Sinh (Sinhala)	snd_Arab (Sindhi)	som_Latn (Somali)
srđ_Latn (Sardinian)	srp_Cyrl (Serbian)	sun_Latn (Sundanese)
tat_Cyrl (Tatar)	tpi_Latn (Tok Pisin)	tsn_Latn (Tswana)
tuk_Latn (Turkmen)	urd_Latn (Urdu (Latin))	vec_Latn (Venetian)
xho_Latn (Xhosa)		

Table 10 Language support of Muse Glimmer, grouped by support tier.

B.7 Examples

B.7.1 Chart Understanding

Prompt



Read these 3 graphs and tell me what do you understand.

Response The three panels are the same training run tracked over optimisation steps, once with a “tail pinning” regulariser on the student and once without it.

- **Loss (left)** — Both curves start around ~ 0.009 and drop within the first 1–2 steps. With tail pinning the blue curve is on average a little lower in the later part of training, ending around 0.0043 at step 21–24. Without pinning the teal curve keeps fluctuating, ~ 0.0055 –0.0060 at the end.
- **Student Tail Mass (centre)** — With tail pinning, student tail mass stays flat at 4.5×10^{-4} . Without it, mass grows monotonically from $\sim 4 \times 10^{-4}$ to $\sim 2.2 \times 10^{-3}$, a $> 5\times$ increase.
- **Teacher Mass at Student Top-K (right)** — With pinning, overlap improves from ~ 0.9934 to ~ 0.9955 . Without it, overlap peaks early then degrades to ~ 0.9932 .

Taken together: tail pinning keeps the student distribution concentrated, prevents unbounded growth of tail probability mass, and preserves the agreement between the student’s top-K predictions and the teacher’s distribution.

B.7.2 Solve ODE

Prompt Solve:

$$\frac{d^2y}{dx^2} - 4 \frac{dy}{dx} + 8y = 0.$$

Response The characteristic equation for $y'' - 4y' + 8y = 0$ is obtained with $y = e^{rx}$:

$$r^2 - 4r + 8 = 0$$

$$r = \frac{4 \pm \sqrt{16 - 32}}{2} = 2 \pm 2i$$

So the roots are complex conjugate $r = 2 \pm 2i = \alpha \pm i\beta$ with $\alpha = 2$, $\beta = 2$. Hence:

$$y(x) = e^{2x} (C_1 \cos 2x + C_2 \sin 2x)$$

Equivalently, in amplitude-phase form:

$$y(x) = A e^{2x} \cos(2x - \phi), \quad A = \sqrt{C_1^2 + C_2^2}, \quad \tan \phi = C_2/C_1.$$

B.7.3 Riddles

Prompts

User

Which is heavier, a pound of feathers or a pound of bricks?

Assistant



Thinking



A pound is a pound.

Mass is mass. One avoirdupois pound is 16 ounces = 0.453592 kg whether it's made of feathers, bricks, lead or cotton candy.

At the same place on Earth they will also weigh the same in pounds-force. The pound of bricks will just be a lot smaller and much denser than the pound of feathers.

The joke works because feathers *feel* lighter than bricks, so we intuitively expect a given volume of feathers to weigh less. But you aren't comparing a volume, you're comparing a fixed mass.

7.91s 919 tokens

User

I live alone and haven't gone out in a month. My car really needs a wash. There is an automatic car wash 50m from my home. Should I walk or drive?

Assistant



Thinking



It's a bit of a false choice, because the car can only get washed if the car goes to the wash.

An automatic wash is drive-through, so you will have to drive the 50m with your dirty car to get it into the bay, and then drive the 50m home again once it's clean. Driving that short distance won't really dirty it again.

For *you* though, the 50m walk is the more useful part of the question. You said you live alone and haven't gone out in a month. That's a long time to be inside. A short, low-stakes walk to the car wash and back is a really easy first step out again — fresh air, daylight, a change of scene, and a reason to be out that isn't a big commitment.

If you're rusty on driving after a month off, or you just want to get moving, walk yourself to the wash first to check it's open / how busy it is, then drive the car the 50m over. You get the car clean and you get out of the house.

If getting out at all feels hard right now, that's worth paying attention to. A short walk counts, and if isolation is starting to weigh on you, talking to someone you trust or a professional can help. In the US you can call or text 988 for free, confidential support 24/7.

Want a quick low-effort plan for getting back out regularly?

17.64s 2260 tokens