2 Branches 0 Tags Go to file Go to file Code

MiniMax-AI-Dev Merge pull request #42 from rogeryoungh/patch-2

2e575ef · 3 months ago

.github/ISSUE_TEMPLATE	update issue template	3 months ago
docs	update wechat	4 months ago
figures	update bench figure	4 months ago
LICENSE	Reformat license	3 months ago
README.md	update README	3 months ago

README License

Join Our 💬 WeChat | 🧩 Discord community.
MiniMax Agent | ⚡ API (Now Free for a limited time!) | MCP | MiniMax Website
🤗 Hugging Face | 🐙 GitHub | 🤖 ModelScope | 📄 License: MIT

Meet MiniMax-M2

Today, we release and open source MiniMax-M2, a **Mini** model built for **Max** coding & agentic workflows.

MiniMax-M2 redefines efficiency for agents. It's a compact, fast, and cost-effective MoE model (230 billion total parameters with 10 billion active parameters) built for elite performance in coding and agentic tasks, all while maintaining powerful general intelligence. With just 10 billion activated parameters, MiniMax-M2 provides the sophisticated, end-to-end tool use performance expected from today's leading models, but in a streamlined form factor that makes deployment and scaling easier than ever.

About

MiniMax-M2, a model built for Max coding & agentic workflows.

www.minimax.io/

#large-language-models #llm

Readme
View license
Activity
Custom properties
2.4k stars
12 watching

Releases

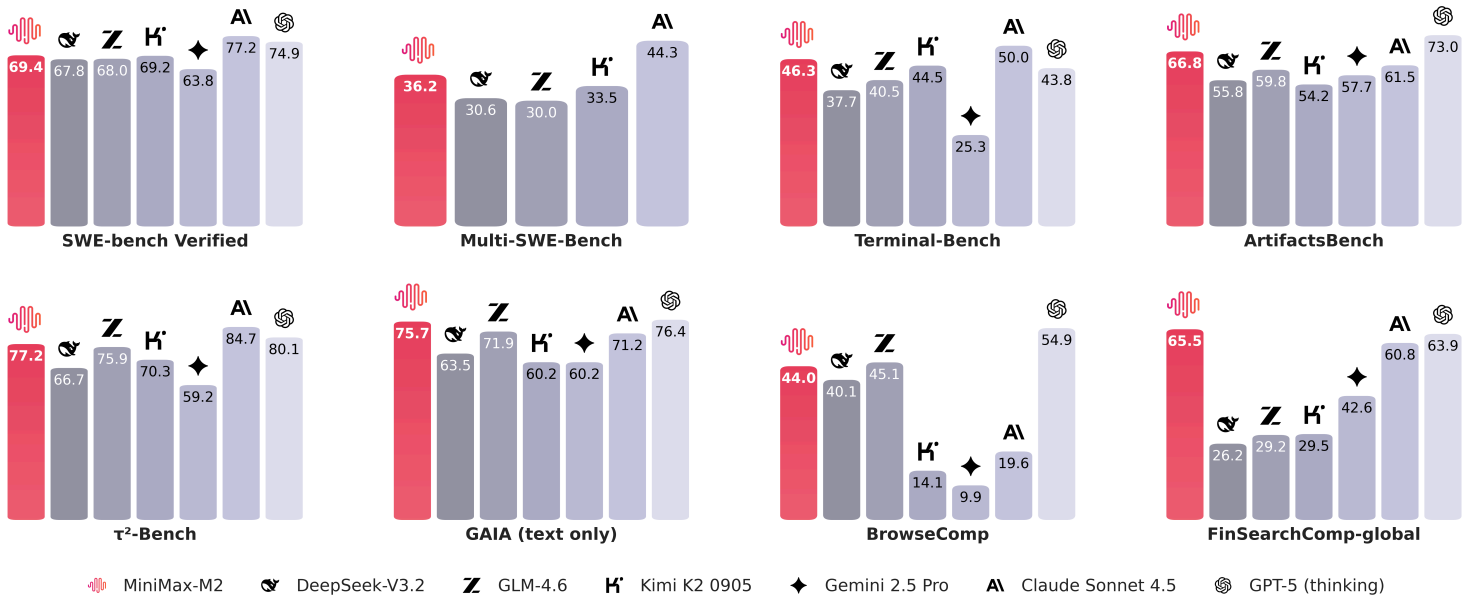
No releases published

Packages

No packages published

Contributors 4

sriting Jiaren Cai
MiniMax-AI-Dev MiniMax
rogeryoungh Roger Young
ximiximi423 ximi



Highlights

Superior Intelligence. According to benchmarks from Artificial Analysis, MiniMax-M2 demonstrates highly competitive general intelligence across mathematics, science, instruction following, coding, and agentic tool use. **Its composite score ranks #1 among open-source models globally**.

Advanced Coding. Engineered for end-to-end developer workflows, MiniMax-M2 excels at multi-file edits, coding-run-fix loops, and test-validated repairs. Strong performance on Terminal-Bench and (Multi-)SWE-Bench–style tasks demonstrates practical effectiveness in terminals, IDEs, and CI across languages.

Agent Performance. MiniMax-M2 plans and executes complex, long-horizon toolchains across shell, browser, retrieval, and code runners. In BrowseComp-style evaluations, it consistently locates hard-to-surface sources, maintains evidence traceable, and gracefully recovers from flaky steps.

Efficient Design. With 10 billion activated parameters (230 billion in total), MiniMax-M2 delivers lower latency, lower cost, and higher throughput for interactive agents and batched sampling—perfectly aligned with the shift toward highly deployable models that still shine on coding and agentic tasks.

Coding & Agentic Benchmarks

These comprehensive evaluations test real-world end-to-end coding and agentic tool use: editing real repos, executing commands, browsing the web, and delivering functional solutions. Performance on this suite correlates with day-to-day developer experience in terminals, IDEs, and CI.

Benchmark	MiniMax-M2	Claude Sonnet 4	Claude Sonnet 4.5	Gemini 2.5 Pro	GPT-5 (thinking)	GLM-4.6	Kimi K2 0905	DeepSeek-V3.2
SWE-bench Verified	69.4	72.7 *	77.2 *	63.8 *	74.9 *	68 *	69.2 *	67.8 *
Multi-SWE-Bench	36.2	35.7 *	44.3	/	/	30	33.5	30.6
SWE-bench Multilingual	56.5	56.9 *	68	/	/	53.8	55.9 *	57.9 *
Terminal-Bench	46.3	36.4 *	50 *	25.3 *	43.8 *	40.5 *	44.5 *	37.7 *
ArtifactsBench	66.8	57.3*	61.5	57.7*	73*	59.8	54.2	55.8
BrowseComp	44	12.2	19.6	9.9	54.9*	45.1*	14.1	40.1*
BrowseComp-zh	48.5	29.1	40.8	32.2	65	49.5	28.8	47.9*
GAIA (text only)	75.7	68.3	71.2	60.2	76.4	71.9	60.2	63.5
xbench-DeepSearch	72	64.6	66	56	77.8	70	61	71

Benchmark	MiniMax-M2	Claude Sonnet 4	Claude Sonnet 4.5	Gemini 2.5 Pro	GPT-5 (thinking)	GLM-4.6	Kimi K2 0905	DeepSeek-V3.2
HLE (w/ tools)	31.8	20.3	24.5	28.4 *	35.2 *	30.4 *	26.9 *	27.2 *
τ²-Bench	77.2	65.5*	84.7*	59.2	80.1*	75.9*	70.3	66.7
FinSearchComp-global	65.5	42	60.8	42.6*	63.9*	29.2	29.5*	26.2
AgentCompany	36	37	41	39.3*	/	35	30	34

Notes: Data points marked with an asterisk (*) are taken directly from the model's official tech report or blog. All other metrics were obtained using the evaluation methods described below.

- SWE-bench Verified: We use the same scaffold as R2E-Gym (Jain et al. 2025) on top of OpenHands to test with agents on SWE tasks. All scores are validated on our internal infrastructure with 128k context length, 100 max steps, and no test-time scaling. All git-related content is removed to ensure agent sees only the code at the issue point.
- Multi-SWE-Bench & SWE-bench Multilingual: All scores are averaged across 8 runs using the claude-code CLI (300 max steps) as the evaluation scaffold.
- Terminal-Bench: All scores are evaluated with the official claude-code from the original Terminal-Bench repository(commit `94bf692`), averaged over 8 runs to report the mean pass rate.
- ArtifactsBench: All Scores are computed by averaging three runs with the official implementation of ArtifactsBench, using the stable Gemini-2.5-Pro as the judge model.
- BrowseComp & BrowseComp-zh & GAIA (text only) & xbench-DeepSearch: All scores reported use the same agent framework as WebExplorer (Liu et al. 2025), with minor tools description adjustment. We use the 103-sample text-only GAIA validation subset following WebExplorer (Liu et al. 2025).
- HLE (w/ tools): All reported scores are obtained using search tools and a Python tool. The search tools employ the same agent framework as WebExplorer (Liu et al. 2025), and the Python tool runs in a Jupyter environment. We use the text-only HLE subset.
- τ²-Bench: All scores reported use "extended thinking with tool use", and employ GPT-4.1 as the user simulator.
- FinSearchComp-global: Official results are reported for GPT-5-Thinking, Gemini 2.5 Pro, and Kimi-K2. Other models are evaluated using the open-source FinSearchComp (Hu et al. 2025) framework using both search and Python tools, launched simultaneously for consistency.
- AgentCompany: All scores reported use OpenHands 0.42 agent framework.

Intelligence Benchmarks

We align with **Artificial Analysis**, which aggregates challenging benchmarks using a consistent methodology to reflect a model's broader **intelligence profile** across math, science, instruction following, coding, and agentic tool use.

Metric (AA)	MiniMax-M2	Claude Sonnet 4	Claude Sonnet 4.5	Gemini 2.5 Pro	GPT-5 (thinking)	GLM-4.6	Kimi K2 0905	DeepSeek-V3.2
AIME25	78	74	88	88	94	86	57	88
MMLU-Pro	82	84	88	86	87	83	82	85
GPQA-Diamond	78	78	83	84	85	78	77	80
HLE (w/o tools)	12.5	9.6	17.3	21.1	26.5	13.3	6.3	13.8
LiveCodeBench (LCB)	83	66	71	80	85	70	61	79
SciCode	36	40	45	43	43	38	31	38
IFBench	72	55	57	49	73	43	42	54
AA-LCR	61	65	66	66	76	54	52	69
τ²-Bench-Telecom	87	65	78	54	85	71	73	34
Terminal-Bench-Hard	24	30	33	25	31	23	23	29
AA Intelligence	61	57	63	60	69	56	50	57

> AA: All scores of MiniMax-M2 aligned with Artificial Analysis Intelligence Benchmarking Methodology (https://artificialanalysis.ai/methodology/intelligence-benchmarking). All scores of other models reported from https://artificialanalysis.ai/.

Why activation size matters

By maintaining activations around **10B** , the plan → act → verify loop in the agentic workflow is streamlined, improving responsiveness and reducing compute overhead:

- **Faster feedback cycles** in compile-run-test and browse-retrieve-cite chains.
- **More concurrent runs** on the same budget for regression suites and multi-seed explorations.
- **Simpler capacity planning** with smaller per-request memory and steadier tail latency.

In short: **10B activations = responsive agent loops + better unit economics**.

At a glance

If you need frontier-style coding and agents without frontier-scale costs, **MiniMax-M2** hits the sweet spot: fast inference speeds, robust tool-use capabilities, and a deployment-friendly footprint.

We look forward to your feedback and to collaborating with developers and researchers to bring the future of intelligent collaboration one step closer.

How to Use

- Our product **MiniMax Agent**, built on MiniMax-M2, is now **publicly available and free** for a limited time: https://agent.minimax.io/
- The MiniMax-M2 API is now live on the **MiniMax Open Platform** and is **free** for a limited time: https://platform.minimax.io/docs/guides/text-generation
- The MiniMax-M2 model weights are now **open-source**, allowing for local deployment and use: https://huggingface.co/MiniMaxAI/MiniMax-M2.

Local Deployment Guide

Download the model from HuggingFace repository: https://huggingface.co/MiniMaxAI/MiniMax-M2. We recommend using the following inference frameworks (listed alphabetically) to serve the model:

SGLang

We recommend using SGLang to serve MiniMax-M2. SGLang provides solid day-0 support for MiniMax-M2 model. Please refer to our SGLang Deployment Guide for more details, and thanks so much for our collaboration with the SGLang team.

vLLM

We recommend using vLLM to serve MiniMax-M2. vLLM provides efficient day-0 support of MiniMax-M2 model, check https://docs.vllm.ai/projects/recipes/en/latest/MiniMax/MiniMax-M2.html for latest deployment guide. We also provide our vLLM Deployment Guide.

MLX

We recommend using MLX-LM to serve MiniMax-M2. Please refer to our MLX Deployment Guide for more details.

Inference Parameters

We recommend using the following parameters for best performance: `temperature=1.0` , `top_p = 0.95` , `top_k = 40` .

IMPORTANT: MiniMax-M2 is an interleaved thinking model. Therefore, when using it, it is important to retain the thinking content from the