

Moonshot AI releases Kimi K2, a 1-trillion-parameter open-weights model

Source copy kept by wheresthe.ai for the story “Moonshot AI releases Kimi K2, a 1-trillion-parameter open-weights model” (<https://wheresthe.ai/d/cn-moonshot-kimi-k2-open-weights-2025-07/>).

Source: <https://github.com/moonshotai/Kimi-K2>

Text saved 18 Oct 2026 (the page text through a reader, because the site refused a direct fetch).

SHA-256 of the saved text: 129ea5c8610d866d5a37e383a8534ce5ebb41a78049cf50eb6fc3b235e670de2

* * *

** 📖 Tech Blog ** | ** 📖 Full Report **

1. Model Introduction

Kimi K2 is a state-of-the-art mixture-of-experts (MoE) language model with 32 billion activated parameters and 1 trillion total parameters. Trained with the Muon optimizer, Kimi K2 achieves exceptional performance across frontier knowledge, reasoning, and coding tasks while being meticulously optimized for agentic capabilities.

Key Features

* **Large-Scale Training:** Pre-trained a 1T parameter MoE model on 15.5T tokens with zero training instability. * **MuonClip Optimizer:** We apply the Muon optimizer to an unprecedented scale, and develop novel optimization techniques to resolve instabilities while scaling up. * **Agentic Intelligence:** Specifically designed for tool use, reasoning, and autonomous problem-solving.

Model Variants

* **Kimi-K2-Base**: The foundation model, a strong start for researchers and builders who want full control for fine-tuning and custom solutions. * **Kimi-K2-Instruct**: The post-trained model, best for drop-in, general-purpose chat and agentic experiences. It is a reflex-grade model without long thinking.

2. Model Summary

| | | | --- | --- | | **Architecture** | Mixture-of-Experts (MoE) | | **Total Parameters** | 1T | | **Activated Parameters** | 32B | | **Number of Layers** (Dense layer included) | 61 | | **Number of Dense Layers** | 1 | | **Attention Hidden Dimension** | 7168 | | **MoE Hidden Dimension** (per Expert) | 2048 | | **Number of Attention Heads** | 64 | | **Number of Experts** | 384 | | **Selected Experts per Token** | 8 | | **Number of Shared Experts** | 1 | | **Vocabulary Size** | 160K | | **Context Length** | 128K | | **Attention Mechanism** | MLA | | **Activation Function** | SwiGLU |

3. Evaluation Results

Instruction model evaluation results

| Benchmark | Metric | Kimi K2 Instruct | DeepSeek-V3-0324 | Qwen3-235B-A22B (non-thinking) | Claude Sonnet 4 (w/o extended thinking) | Claude Opus 4 (w/o extended thinking) | GPT-4.1 | Gemini 2.5 Flash Preview (05-20) | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Coding Tasks** | | LiveCodeBench v6 (Aug 24 - May 25) | Pass@1 | **53.7** | 46.9 | 37.0 | 48.5 | 47.4 | 44.7 | 44.7 | | OJBench | Pass@1 | **27.1** | 24.0 | 11.3 | 15.3 | 19.6 | 19.5 | 19.5 | | MultiPL-E | Pass@1 | **85.7** | 83.1 | 78.2 | 88.6 | **89.6** | 86.7 | 85.6 | | SWE-bench Verified (Agentless Coding) | Single Patch w/o Test (Acc) | **51.8** | 36.6 | 39.4 | 50.2 | **53.0** | 40.8 | 32.6 | | SWE-bench Verified (Agentic Coding) | Single Attempt (Acc) | **65.8** | 38.8 | 34.4 | **72.7** | 72.5 | 54.6 | — | | Multiple Attempts (Acc) | **71.6** | — | — | **80.2** | 79.4 | — | — | | SWE-bench Multilingual (Agentic Coding) | Single Attempt (Acc) | **47.3** | 25.8 | 20.9 | **51.0** | — | 31.5 | — | | TerminalBench | Inhouse Framework (Acc) | **30.0** | — | — | 35.5 | **43.2** | 8.3 | — | | Terminus (Acc) | **25.0** | 16.3 | 6.6 | — | — | **30.3** | 16.8 | | Aider-Polyglot | Acc | 60.0 | 55.1 | **61.8** | 56.4 | **70.7** | 52.4 | 44.0 | | **Tool Use Tasks** | | Tau2 retail | Avg@4 | **70.6** | 69.1 | 57.0 | 75.0 | **81.8** | 74.8 | 64.3 | | Tau2 airline | Avg@4 | **56.5** | 39.0 | 26.5 | 55.5 | **60.0** | 54.5 | 42.5 | | Tau2 telecom | Avg@4 | **65.8** | 32.5 | 22.1 | 45.2 | 57.0 | 38.6 | 16.9 | | AceBench | Acc | **76.5** | 72.7 | 70.5 | 76.2 | 75.6 | **80.1** | 74.5 | | **Math & STEM Tasks** | | AIME 2024 | Avg@64 | **69.6** | 59.4 | 40.1 | 43.4 | 48.2 | 46.5 | 61.3 | | AIME 2025 | Avg@64 | **49.5** | 46.7 | 24.7 | 33.1 | 33.9 | 37.0 | 46.6 | | MATH-500 | Acc |

****97.4**** | 94.0* | 91.2* | 94.0 | 94.4 | 92.4 | 95.4 | | HMMT 2025 | Avg@32 | ****38.8**** | 27.5 | 11.9 | 15.9 | 15.9 | 19.4 | 34.7 | | CNMO 2024 | Avg@16 | 74.3 | ****74.7**** | 48.6 | 60.4 | 57.6 | 56.6 | ****75.0**** | | PolyMath-en | Avg@4 | ****65.1**** | 59.5 | 51.9 | 52.8 | 49.8 | 54.0 | 49.9 | | ZebraLogic | Acc | ****89.0**** | 84.0 | 37.7* | 73.7 | 59.3 | 58.5 | 57.9 | | AutoLogi | Acc | ****89.5**** | 88.9 | 83.3 | ****89.8**** | 86.1 | 88.2 | 84.1 | | GPQA-Diamond | Avg@8 | ****75.1**** | 68.4* | 62.9* | 70.0* | 74.9* | 66.3 | 68.2 | | SuperGPQA | Acc | ****57.2**** | 53.7 | 50.2 | 55.7 | 56.5 | 50.8 | 49.6 | | Humanity's Last Exam (Text Only) | - | 4.7 | 5.2 | ****5.7**** | 5.8 | ****7.1**** | 3.7 | 5.6 | | ****General Tasks**** | | MMLU | EM | ****89.5**** | 89.4 | 87.0 | 91.5 | ****92.9**** | 90.4 | 90.1 | | MMLU-Redux | EM | ****92.7**** | 90.5 | 89.2 | 93.6 | ****94.2**** | 92.4 | 90.6 | | MMLU-Pro | EM | 81.1 | ****81.2**** | 77.3 | 83.7 | ****86.6**** | 81.8 | 79.4 | | IFEval | Prompt Strict | ****89.8**** | 81.1 | 83.2* | 87.6 | 87.4 | 88.0 | 84.3 | | Multi-Challenge | Acc | ****54.1**** | 31.4 | 34.0 | 46.8 | 49.0 | 36.4 | 39.5 | | SimpleQA | Correct | ****31.0**** | 27.7 | 13.2 | 15.9 | 22.8 | ****42.3**** | 23.3 | | Livebench | Pass@1 | ****76.4**** | 72.4 | 67.6 | 74.8 | 74.6 | 69.8 | 67.8 |

- Bold denotes global SOTA, and underlined denotes open-source SOTA.
- Data points marked with * are directly from the model's tech report or blog.
- All metrics, except for SWE-bench Verified (Agentless), are evaluated with an 8k output token length. SWE-bench Verified (Agentless) is limited to a 16k output token length.
- Kimi K2 achieves 65.8% pass@1 on the SWE-bench Verified tests with bash/editor tools (single-attempt patches, no test-time compute). It also achieves a 47.3% pass@1 on the SWE-bench Multilingual tests under the same conditions. Additionally, we report results on SWE-bench Verified tests (71.6%) that leverage parallel test-time compute by sampling multiple sequences and selecting the single best via an internal scoring model.
- To ensure the stability of the evaluation, we employed avg@k on the AIME, HMMT, CNMO, PolyMath-en, GPQA-Diamond, EvalPlus, Tau2.
- Some data points have been omitted due to prohibitively expensive evaluation costs.

* * *

Base model evaluation results

Benchmark	Metric	Shot	Kimi K2 Base	Deepseek-V3-Base	Qwen2.5-72B	Llama 4 Maverick	---	---	---
General Tasks	MMLU	EM	5-shot	**87.8**	87.1	86.1	84.9		
	MMLU-pro	EM	5-shot	**69.2**	60.6	62.8	63.5		
	MMLU-redux-2.0	EM	5-shot	**90.2**	89.5	87.8	88.2		
	SimpleQA	Correct	5-shot	**35.3**	26.5	10.3	23.7		
	TriviaQA	EM	5-shot	**85.1**	84.1	76.0	79.3		
	GPQA-Diamond	Avg@8	5-shot	48.1	**50.5**	40.8	49.4		
	SuperGPQA	EM	5-shot	**44.7**	39.2	34.2	38.8		
	Coding Tasks	LiveCodeBench v6	Pass@1	1-shot	**26.3**	22.9	21.1	25.1	
	EvalPlus	Pass@1	-	**80.3**	65.6	66.0	65.5		
	Mathematics Tasks	MATH	EM	4-shot	**70.2**	60.1	61.0	63.0	
GSM8k	EM	8-shot	**92.1**	91.7	90.4	86.3			
	Chinese Tasks	C-Eval	EM	5-shot	**92.5**	90.0	90.9	80.9	
CSimpleQA	Correct	5-shot	**77.6**	72.1	50.5	53.5			

- We only evaluate open-source pretrained models in this work. We report results for Qwen2.5-72B because the base checkpoint for Qwen3-235B-A22B was not open-sourced at the time of our study.
- All models are evaluated using the same evaluation protocol.

4. Deployment

Note

You can access Kimi K2's API on <https://platform.moonshot.ai>, we provide an OpenAI/Anthropic-compatible API for you.

The Anthropic-compatible API maps temperature by `real_temperature = request_temperature * 0.6` for better compatibility with existing applications.

Our model checkpoints are stored in block-fp8 format, you can find it on Huggingface.

Currently, it is recommended to run Kimi-K2 on the following inference engines:

* vLLM * SGLang * KTransformers * TensorRT-LLM

Deployment examples for vLLM and SGLang can be found in the Model Deployment Guide.

* * *

5. Model Usage

Chat Completion

Once the local inference service is set up, you can interact with it through the chat endpoint:

```
def simple_chat(client: OpenAI, model_name: str): messages = [ {"role": "system", "content": "You are Kimi, an AI assistant created by Moonshot AI."}, {"role": "user", "content": [{"type": "text", "text": "Please give a brief self-introduction."}]} ] response = client.chat.completions.create( model=model_name, messages=messages, stream=False, temperature=0.6, max_tokens=256 ) print(response.choices[0].message.content)
```

Note

The recommended temperature for Kimi-K2-Instruct is `temperature = 0.6`. If no special instructions are required, the system prompt is a good default.

Tool Calling

Kimi-K2-Instruct has strong tool-calling capabilities. To enable them, you need to pass the list of available tools in each request, then the model will autonomously decide when and how to invoke them.

The following example demonstrates calling a weather tool end-to-end:

```
# Your tool implementation
def get_weather(city: str) -> dict: return {"weather": "Sunny"}

# Tool schema definition
tools = [ {"type": "function", "function": { "name": "get_weather", "description": "Retrieve current weather information. Call this when the user asks about the weather.", "parameters": { "type": "object", "required": ["city"], "properties": { "city": { "type": "string", "description": "Name of the city" } } } } ]

# Map tool names to their implementations
tool_map = { "get_weather": get_weather }

def tool_call_with_client(client: OpenAI, model_name: str):
    messages = [ {"role": "system", "content": "You are Kimi, an AI assistant created by Moonshot AI."}, {"role": "user", "content": "What's the weather like in Beijing today? Use the tool to check."} ]
    finish_reason = None
    while finish_reason is None or finish_reason == "tool_calls":
        completion = client.chat.completions.create( model=model_name, messages=messages, temperature=0.6, tools=tools, # tool list defined above
        tool_choice="auto" )
        choice = completion.choices[0]
        finish_reason = choice.finish_reason
        if finish_reason == "tool_calls":
            messages.append(choice.message)
            for tool_call in choice.message.tool_calls:
                tool_call_name = tool_call.function.name
                tool_call_arguments = json.loads(tool_call.function.arguments)
                tool_function = tool_map[tool_call_name]
                tool_result = tool_function(**tool_call_arguments)
                print("tool_result:", tool_result)

            messages.append({ "role": "tool", "tool_call_id": tool_call.id, "name": tool_call_name, "content": json.dumps(tool_result) })
            print("-" * 100)
            print(choice.message.content)
```

The `tool\_call\_with\_client` function implements the pipeline from user query to tool execution. This pipeline requires the inference engine to support Kimi-K2's native tool-parsing logic. For streaming output and manual tool-parsing, see the Tool Calling Guide.

6. License

Both the code and the model weights are released under the Modified MIT License.

7. Citation

```
``` @misc{kimiteam2025kimik2openagentic, title={Kimi K2: Open Agentic Intelligence}, author={Kimi Team and Yifan Bai and Yiping Bao and Guanduo Chen and Jiahao Chen and Ningxin Chen and Ruijie Chen and Yanru Chen and Yuankun Chen and Yutian Chen and Zhuofu Chen and Jialei Cui and Hao Ding and Mengnan Dong and Angang Du and Chenzhuang Du and Dikang Du and Yulun Du and Yu Fan and Yichen Feng and Kelin Fu and Bofei Gao and Hongcheng Gao and Peizhong Gao and Tong Gao and Xinran Gu and Longyu Guan and Haiqing Guo and Jianhang Guo and Hao Hu and Xiaoru Hao and Tianhong He and Weiran He and Wenyang He and Chao Hong and Yangyang Hu and Zhenxing Hu and Weixiao Huang and Zhiqi Huang and Zihao Huang and Tao Jiang and Zhejun Jiang and Xinyi Jin and Yongsheng Kang and Guokun Lai and Cheng Li and Fang Li and Haoyang Li and Ming Li and Wentao Li and Yanhao Li and Yiwei Li and Zhaowei Li and Zheming Li and Hongzhan Lin and Xiaohan Lin and Zongyu Lin and Chengyin Liu and Chenyu Liu and Hongzhang Liu and
```

Jingyuan Liu and Junqi Liu and Liang Liu and Shaowei Liu and T. Y. Liu and Tianwei Liu and Weizhou Liu and Yangyang Liu and Yibo Liu and Yiping Liu and Yue Liu and Zhengying Liu and Enzhe Lu and Lijun Lu and Shengling Ma and Xinyu Ma and Yingwei Ma and Shaoguang Mao and Jie Mei and Xin Men and Yibo Miao and Siyuan Pan and Yebo Peng and Ruoyu Qin and Bowen Qu and Zeyu Shang and Lidong Shi and Shengyuan Shi and Feifan Song and Jianlin Su and Zhengyuan Su and Xinjie Sun and Flood Sung and Heyi Tang and Jiawen Tao and Qifeng Teng and Chensi Wang and Dinglu Wang and Feng Wang and Haiming Wang and Jianzhou Wang and Jiaxing Wang and Jinhong Wang and Shengjie Wang and Shuyi Wang and Yao Wang and Yejie Wang and Yiqin Wang and Yuxin Wang and Yuzhi Wang and Zhaoji Wang and Zhengtao Wang and Zhexu Wang and Chu Wei and Qianqian Wei and Wenhao Wu and Xingzhe Wu and Yuxin Wu and Chenjun Xiao and Xiaotong Xie and Weimin Xiong and Boyu Xu and Jing Xu and Jinjing Xu and L. H. Xu and Lin Xu and Suting Xu and Weixin Xu and Xinran Xu and Yangchuan Xu and Ziyao Xu and Junjie Yan and Yuzi Yan and Xiaofei Yang and Ying Yang and Zhen Yang and Zhilin Yang and Zonghan Yang and Haotian Yao and Xingcheng Yao and Wenjie Ye and Zhuorui Ye and Bohong Yin and Longhui Yu and Enming Yuan and Hongbang Yuan and Mengjie Yuan and Haobing Zhan and Dehao Zhang and Hao Zhang and Wanlu Zhang and Xiaobin Zhang and Yangkun Zhang and Yizhi Zhang and Yongting Zhang and Yu Zhang and Yutao Zhang and Yutong Zhang and Zheng Zhang and Haotian Zhao and Yikai Zhao and Huabin Zheng and Shaojie Zheng and Jianren Zhou and Xinyu Zhou and Zaida Zhou and Zhen Zhu and Weiyu Zhuang and Xinxing Zu}, year={2025}, eprint={2507.20534}, archivePrefix={arXiv}, primaryClass={cs.LG}, url={https://arxiv.org/abs/2507.20534}, }

\*\*\*

## ## 8. Contact Us

If you have any questions or concerns, please reach out to us at [support@moonshot.cn](mailto:support@moonshot.cn).