

Alibaba releases Qwen3, eight open-weights models under Apache 2.0

Source copy kept by wheresthe.ai for the story "Alibaba releases Qwen3, eight open-weights models under Apache 2.0" (<https://wheresthe.ai/d/cn-alibaba-qwen3-open-weights-2025-04/>).

Source: <https://github.com/QwenLM/Qwen3>

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

SHA-256 of the saved text: 3375bca4d8141f3fba6e291d507ede6d60f2733659767643aec44bfa6205c05

♥ **Qwen Chat** | [Hugging Face](#) | [ModelScope](#) | [Paper](#) | [Blog](#) | [Documentation](#) 📱 [Demo](#) | [WeChat \(微信\)](#) | [Discord](#)

Visit our Hugging Face or ModelScope organization (click links above), search checkpoints with names starting with `Qwen3-` or visit the Qwen3 collection, and you will find all you need! Enjoy!

To learn more about Qwen3, feel free to read our documentation [EN|ZH]. Our documentation consists of the following sections:

* Quickstart: the basic usages and demonstrations; * Inference: the guidance for the inference with Transformers, including batch inference, streaming, etc.; * Run Locally: the instructions for running LLM locally on CPU and GPU, with frameworks like llama.cpp, Ollama, and LM Studio; * Deployment: the demonstration of how to deploy Qwen for large-scale inference with frameworks like SGLang, vLLM, TGI, etc.; * Quantization: the practice of quantizing LLMs with GPTQ, AWQ, as well as the guidance for how to make high-quality quantized GGUF files; * Training: the instructions for post-training, including SFT and RLHF (TODO) with frameworks like Axolotl, LLaMA-Factory, etc. * Framework: the usage of Qwen with frameworks for application, e.g., RAG, Agent, etc.

Introduction

Qwen3-2507

Over the past three months, we continued to explore the potential of the Qwen3 families and we are excited to introduce the updated **Qwen3-2507** in two variants, Qwen3-Instruct-2507 and Qwen3-Thinking-2507, and three sizes, 235B-A22B, 30B-A3B, and 4B.

Qwen3-Instruct-2507 is the updated version of the previous Qwen3 non-thinking mode, featuring the following key enhancements:

* **Significant improvements** in general capabilities, including **instruction following**, logical reasoning, text comprehension, mathematics, science, coding and tool usage. * **Substantial gains** in long-tail knowledge coverage across **multiple languages**. * **Markedly better alignment** with user preferences in **subjective and open-ended tasks**, enabling more helpful responses and higher-quality text generation. * **Enhanced capabilities** in **256K-token long-context understanding**, extendable up to **1 million tokens**.

Qwen3-Thinking-2507 is the continuation of Qwen3 thinking model, with improved quality and depth of reasoning, featuring the following key enhancements:

* **Significantly improved performance** on reasoning tasks, including logical reasoning, mathematics, science, coding, and academic benchmarks that typically require human expertise — achieving **state-of-the-art results** among open-weight thinking models. * **Markedly better general capabilities**, such as instruction following, tool usage, text generation, and alignment with human preferences. * **Enhanced 256K long-context understanding** capabilities, extendable up to **1 million tokens**.

Previous Qwen3 Release ### Qwen3 (aka Qwen3-2504)

We are excited to announce the release of Qwen3, the latest addition to the Qwen family of large language models. These models represent our most advanced and intelligent systems to date, improving from our experience in building QwQ and Qwen2.5. We are making the weights of Qwen3 available to the public, including both dense and Mixture-of-Expert (MoE) models.

The highlights from Qwen3 include:

Dense and Mixture-of-Experts (MoE) models of various sizes, available in 0.6B, 1.7B, 4B, 8B, 14B, 32B and 30B-A3B, 235B-A22B. **Seamless switching between thinking mode** (for complex logical reasoning, math, and coding) and **non-thinking mode** (for efficient, general-purpose chat), ensuring optimal performance across various scenarios. **Significantly enhancement in reasoning capabilities**, surpassing previous QwQ (in thinking mode) and Qwen2.5 instruct models (in non-thinking mode) on mathematics, code generation, and commonsense logical reasoning. **Superior human preference alignment**, excelling in creative writing, role-playing, multi-turn dialogues, and instruction following, to deliver a more natural, engaging, and immersive conversational experience. **Expertise in agent capabilities**, enabling precise integration with external tools in both thinking and unthinking modes and achieving leading performance among open-source models in complex agent-based tasks. **Support of 100+ languages and dialects** with strong capabilities for **multilingual instruction following** and **translation**.

News

2025.08.08: You can now use Qwen3-2507 to handle ultra-long inputs of **1 million tokens**! See the update modelcards (235B-A22B-Instruct-2507, 235B-A22B-Thinking-2507, A30B-A3B-Instruct-2507, A30B-A3B-Thinking-2507) for how to enable this feature. **2025.08.06:** The final open release of Qwen3-2507, Qwen3-4B-Instruct-2507 and Qwen3-4B-Thinking-2507, is out! **2025.07.31:** Qwen3-30B-A3B-Thinking-2507 is released. Check out the modelcard for more details! **2025.07.30:** Qwen3-30B-A3B-Instruct-2507 is released. Check out the modelcard for more details! **2025.07.25:** We released the updated version of Qwen3-235B-A22B thinking mode, named Qwen3-235B-A22B-Thinking-2507. Check out the modelcard for more details! **2025.07.21:** We released the updated version of Qwen3-235B-A22B non-thinking mode, named Qwen3-235B-A22B-Instruct-2507, featuring significant enhancements over the previous version and supporting 256K-token long-context understanding. Check our modelcard for more details! **2025.04.29:** We released the Qwen3 series. Check our blog for more details! **2024.09.19:** We released the Qwen2.5 series. This time there are 3 extra model sizes: 3B, 14B, and 32B for more possibilities. Check our blog for more! **2024.06.06:** We released the Qwen2 series. Check our blog! **2024.03.28:** We released the first MoE model of Qwen: Qwen1.5-MoE-A2.7B! Temporarily, only HF transformers and vLLM support the model. We will soon add the support of llama.cpp, mlx-lm, etc. Check our blog for more information! **2024.02.05:** We released the Qwen1.5 series.

Performance

Detailed evaluation results are reported in this [blog](#) (Qwen3-2504) and this [blog](#) (Qwen3-2507) [coming soon]](<https://github.com/QwenLM/Qwen3/blob/main>).

For requirements on GPU memory and the respective throughput, see results [here](#).

Run Qwen3

Transformers

Transformers is a library of pretrained natural language processing for inference and training. The latest version of `transformers` is recommended and `transformers>=4.51.0` is required.

Qwen3-Instruct-2507

The following contains a code snippet illustrating how to use Qwen3-30B-A3B-Instruct-2507 to generate content based on given inputs.

```
from transformers import AutoModelForCausalLM, AutoTokenizer

model_name = "Qwen/Qwen3-30B-A3B-Instruct-2507"

# load the tokenizer and the model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name, torch_dtype="auto", device_map="auto")

# prepare the model input prompt
prompt = "Give me a short introduction to large language model."
messages = [{"role": "user", "content": prompt}]
text = tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
model_inputs = tokenizer([text], return_tensors="pt").to(model.device)

# conduct text completion
generated_ids = model.generate(model_inputs, max_new_tokens=16384)
output_ids = generated_ids[0][len(model_inputs.input_ids[0]):].tolist()

content = tokenizer.decode(output_ids, skip_special_tokens=True)
```

```
print("content:", content)
```

Note

Qwen3-Instruct-2507 supports only non-thinking mode and does not generate ``<think></think>`` blocks in its output. Meanwhile, specifying ``enable_thinking=False`` is no longer required.

Qwen3-Thinking-2507

The following contains a code snippet illustrating how to use Qwen3-30B-A3B-Thinking-2507 to generate content based on given inputs.

```
from transformers import AutoModelForCausalLM, AutoTokenizer

model_name = "Qwen/Qwen3-30B-A3B-Thinking-2507"

# load the tokenizer and the model
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name, torch_dtype="auto", device_map="auto")

# prepare the model input
prompt = "Give me a short introduction to large language model."
messages = [{"role": "user", "content": prompt}]
text = tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
model_inputs = tokenizer([text], return_tensors="pt").to(model.device)

# conduct text completion
generated_ids = model.generate(model_inputs, max_new_tokens=32768)
output_ids = generated_ids[0][len(model_inputs.input_ids):].tolist()

# parsing thinking content
try:
    # find index of </think>
    index = len(output_ids) - output_ids[::-1].index(151668)
except ValueError:
    index = 0

thinking_content = tokenizer.decode(output_ids[:index], skip_special_tokens=True).strip("\n")
content = tokenizer.decode(output_ids[index:], skip_special_tokens=True).strip("\n")

print("thinking content:", thinking_content)
# no opening <think> tag
print("content:", content)
```

Note

Qwen3-Thinking-2507 supports only thinking mode. Additionally, to enforce model thinking, the default chat template automatically includes ``<think>``. Therefore, it is normal for the model's output to contain only ``</think>`` without an explicit opening ``<think>`` tag.

Qwen3-Thinking-2507 also features an increased thinking length. We strongly recommend its use in highly complex reasoning tasks with adequate maximum generation length.

****Switching Thinking/Non-thinking Modes for Previous Qwen3 Models**** By default, Qwen3 models will think before response. This could be controlled by

* ``enable_thinking=False``: Passing ``enable_thinking=False`` to ``tokenizer.apply_chat_template`` will strictly prevent the model from generating thinking content. * ``/think`` and ``/no_think`` instructions: Use those words in the system or user message to signify whether Qwen3 should think. In multi-turn conversations, the latest instruction is followed.

ModelScope

We strongly advise users especially those in mainland China to use ModelScope. ModelScope adopts a Python API similar to Transformers. The CLI tool ``modelscope download`` can help you solve issues concerning downloading checkpoints. For vLLM and SGLang, the environment variable ``VLLM_USE_MODELSCOPE=true`` and ``SGLANG_USE_MODELSCOPE=true`` can be used respectively.

llama.cpp

`llama.cpp` enables LLM inference with minimal setup and state-of-the-art performance on a wide range of hardware. `llama.cpp>=b5401` is recommended for the full support of Qwen3.

To use the CLI, run the following in a terminal:

```
./llama-cli -hf Qwen/Qwen3-8B-GGUF:Q8_0 --jinja --color -ngl 99 -fa -sm row --temp 0.6 --top-k 20 --top-p 0.95 --min-p 0 -c 40960 -n 32768 --no-context-shift # CTRL+C to exit
```

To use the API server, run the following in a terminal:

```
./llama-server -hf Qwen/Qwen3-8B-GGUF:Q8_0 --jinja --reasoning-format deepseek -ngl 99 -fa -sm row --temp 0.6 --top-k 20 --top-p 0.95 --min-p 0 -c 40960 -n 32768 --no-context-shift --port 8080
```

A simple web front end will be at `http://localhost:8080` and an OpenAI-compatible API will be at `http://localhost:8080/v1`.

For additional guides, please refer to our documentation.

Note

llama.cpp adopts "rotating context management" and infinite generation is made possible by evicting earlier tokens. It could be configured by parameters and the commands above effectively disable it. For more details, please refer to our documentation.

Ollama

After installing Ollama, you can initiate the Ollama service with the following command (Ollama v0.9.0 or higher is recommended):

`ollama serve` # You need to keep this service running whenever you are using ollama

To pull a model checkpoint and run the model, use the `ollama run` command. You can specify a model size by adding a suffix to `qwen3`, such as `:8b` or `:30b-a3b`:

```
ollama run qwen3:8b # Setting parameters, type "/set parameter num_ctx 40960" and "/set parameter num_predict 32768" # To exit, type "/bye" and press ENTER # For Qwen3-2504 models, # - To enable thinking, which is the default, type "/set think" # - To disable thinking, type "/set nothink"
```

You can also access the Ollama service via its OpenAI-compatible API. Please note that you need to (1) keep `ollama serve` running while using the API, and (2) execute `ollama run qwen3:8b` before utilizing this API to ensure that the model checkpoint is prepared. The API is at `http://localhost:11434/v1` by default.

For additional details, please visit ollama.ai.

Note

Ollama's naming may not be consistent with the Qwen's original naming. For example, `qwen3:30b-a3b` in Ollama points to `qwen3:30b-a3b-thinking-2507-q4_K_M` as of August 2025. Please check <https://ollama.com/library/qwen3/tags> before use.

Note

Ollama adopts the same "rotating context management" with llama.cpp. However, its default settings (`num_ctx` 2048 and `num_predict` -1), suggesting infinite generation with a 2048-token context, could lead to trouble for Qwen3 models. We recommend setting `num_ctx` and `num_predict` properly.

LMStudio

Qwen3 has already been supported by lmstudio.ai. You can directly use LMStudio with our GGUF files.

ExecuTorch

To export and run on ExecuTorch (iOS, Android, Mac, Linux, and more), please follow this example.

MNN

To export and run on MNN, which supports Qwen3 on mobile devices, please visit [Alibaba MNN](https://mnn.lol/).

MLX LM

If you are running on Apple Silicon, `mlx-lm` also supports Qwen3 (`mlx-lm>=0.24.0`). Look for models ending with MLX on Hugging Face Hub.

OpenVINO

If you are running on Intel CPU or GPU, OpenVINO toolkit supports Qwen3. You can follow this chatbot example.

Deploy Qwen3

Qwen3 is supported by multiple inference frameworks. Here we demonstrate the usage of `SGLang`, `vLLM` and `TensorRT-LLM`. You can also find Qwen3 models from various inference providers, e.g., Alibaba Cloud Model Studio.

SGLang

SGLang is a fast serving framework for large language models and vision language models. SGLang could be used to launch a server with OpenAI-compatible API service. `sglang>=0.4.6.post1` is required.

For Qwen3-Instruct-2507,

```
python -m sglang.launch_server --model-path Qwen/Qwen3-30B-A3B-Instruct-2507 --port 30000 --context-length 262144
```

For Qwen3-Thinking-2507,

```
python -m sglang.launch_server --model-path Qwen/Qwen3-30B-A3B-Thinking-2507 --port 30000 --context-length 262144 --reasoning-parser deepseek-r1
```

For Qwen3, it is

```
python -m sglang.launch_server --model-path Qwen/Qwen3-8B --port 30000 --context-length 131072 --reasoning-parser qwen3
```

An OpenAI-compatible API will be available at `http://localhost:30000/v1`.

Note

Due to the preprocessing of API requests in SGLang, which drops all `reasoning\_content` fields, the quality of **multi-step tool use with Qwen3 thinking models** may be suboptimal, which requires the existence of the related thinking content. While the fixes are being worked on, as a workaround, we recommend passing the content as it is, without extracting thinking content, and the chat template will correctly handle the processing.

vLLM

vLLM is a high-throughput and memory-efficient inference and serving engine for LLMs. `vllm>=0.9.0` is recommended.

For Qwen3-Instruct-2507,

```
vllm serve Qwen/Qwen3-30B-A3B-Instruct-2507 --port 8000 --max-model-len 262144
```

For Qwen3-Thinking-2507,

```
vllm serve Qwen/Qwen3-30B-A3B-Thinking-2507 --port 8000 --max-model-len 262144 --enable-reasoning --reasoning-parser deepseek_r1
```

For Qwen3, it is

```
vllm serve Qwen/Qwen3-8B --port 8000 --max-model-len 131072 --enable-reasoning --reasoning-parser qwen3
```

An OpenAI-compatible API will be available at `http://localhost:8000/v1`.

Note

Due to the preprocessing of API requests in vLLM, which drops all `reasoning\_content` fields, the quality of **multi-step tool use with Qwen3 thinking models** may be suboptimal, which requires the existence of the related thinking content. While the fixes are being worked on, as a workaround, we recommend passing the content as it is, without extracting thinking content, and the chat template will correctly handle the processing.

TensorRT-LLM

TensorRT-LLM is an open-source LLM inference engine from NVIDIA, which provides optimizations including custom attention kernels, quantization and more on NVIDIA GPUs. Qwen3 is supported in its re-architected PyTorch backend. `tensorrt\_llm>=0.20.0rc3` is recommended. Please refer to the README page for more details.

trtllm-serve Qwen/Qwen3-8B --host localhost --port 8000 --backend pytorch

An OpenAI-compatible API will be available at `http://localhost:8000/v1`.

MindIE

For deployment on Ascend NPUs, please visit Modelers and search for Qwen3.

Build with Qwen3

Tool Use

For tool use capabilities, we recommend taking a look at Qwen-Agent, which provides a wrapper around these APIs to support tool use or function calling with MCP support. Tool use with Qwen3 can also be conducted with SGLang, vLLM, Transformers, llama.cpp, Ollama, etc. Follow guides in our documentation to see how to enable the support.

Finetuning

We advise you to use training frameworks, including Axolotl, UnSloth, Swift, Llama-Factory, etc., to finetune your models with SFT, DPO, GRPO, etc.

License Agreement

All our open-weight models are licensed under Apache 2.0. You can find the license files in the respective Hugging Face repositories.

Citation

If you find our work helpful, feel free to give us a cite.

@article{qwen3, title={Qwen3 Technical Report}, author={An Yang and Anfeng Li and Baosong Yang and Beichen Zhang and Binyuan Hui and Bo Zheng and Bowen Yu and Chang Gao and Chengen Huang and Chenxu Lv and Chujie Zheng and Dayiheng Liu and Fan Zhou and Fei Huang and Feng Hu and Hao Ge and Haoran Wei and Huan Lin and Jialong Tang and Jian Yang and Jianhong Tu and Jianwei Zhang and Jianxin Yang and Jiaxi Yang and Jing Zhou and Jingren Zhou and Junyang Lin and Kai Dang and Keqin Bao and Kexin Yang and Le Yu and Lianghao Deng and Mei Li and Mingfeng Xue and Mingze Li and Pei Zhang and Peng Wang and Qin Zhu and Rui Men and Ruize Gao and Shixuan Liu and Shuang Luo and Tianhao Li and Tianyi Tang and Wenbiao Yin and Xingzhang Ren and Xinyu Wang and Xinyu Zhang and Xuancheng Ren and Yang Fan and Yang Su and Yichang Zhang and Yinger Zhang and Yu Wan and Yuqiong Liu and Zekun Wang and Zeyu Cui and Zhenru Zhang and Zhipeng Zhou and Zihan Qiu}, journal = {arXiv preprint arXiv:2505.09388}, year={2025} }

@article{qwen2.5, title = {Qwen2.5 Technical Report}, author = {An Yang and Baosong Yang and Beichen Zhang and Binyuan Hui and Bo Zheng and Bowen Yu and Chengyuan Li and Dayiheng Liu and Fei Huang and Haoran Wei and Huan Lin and Jian Yang and Jianhong Tu and Jianwei Zhang and Jianxin Yang and Jiaxi Yang and Jingren Zhou and Junyang Lin and Kai Dang and Keming Lu and Keqin Bao and Kexin Yang and Le Yu and Mei Li and Mingfeng Xue and Pei Zhang and Qin Zhu and Rui Men and Runji Lin and Tianhao Li and Tingyu Xia and Xingzhang Ren and Xuancheng Ren and Yang Fan and Yang Su and Yichang Zhang and Yu Wan and Yuqiong Liu and Zeyu Cui and Zhenru Zhang and Zihan Qiu}, journal = {arXiv preprint arXiv:2412.15115}, year = {2024} }

@article{qwen2, title = {Qwen2 Technical Report}, author = {An Yang and Baosong Yang and Binyuan Hui and Bo Zheng and Bowen Yu and Chang Zhou and Chengpeng Li and Chengyuan Li and Dayiheng Liu and Fei Huang and Guanting Dong and Haoran Wei and Huan Lin and Jialong Tang and Jialin Wang and Jian Yang and Jianhong Tu and Jianwei Zhang and Jianxin Ma and Jin Xu and Jingren Zhou and Jinze Bai and Jinzheng He and Junyang Lin and Kai Dang and Keming Lu and Keqin Chen and Kexin Yang and Mei Li and Mingfeng Xue and Na Ni and Pei Zhang and Peng Wang and Ru Peng and Rui Men and Ruize Gao and Runji Lin and Shijie Wang and Shuai Bai and Sinan Tan and Tianhang Zhu and Tianhao Li and Tianyu Liu and Wenbin Ge and Xiaodong Deng and Xiaohuan Zhou and Xingzhang Ren and Xinyu Zhang and Xipin Wei and Xuancheng Ren and Yang Fan and Yang Yao and Yichang Zhang and Yu Wan and Yunfei Chu and Yuqiong Liu and Zeyu Cui and Zhenru Zhang and Zhihao Fan}, journal = {arXiv preprint arXiv:2407.10671}, year = {2024} }

Contact Us

If you are interested to leave a message to either our research team or product team, join our Discord or WeChat

groups!