Qwen/**Qwen3.8-2.4T-A95B** like 144 Follow Qwen 96k

Text Generation · Transformers · Safetensors · qwen3_5_moe_text · conversational · Eval Results · License: qwen3.8-max

Deploy · Copy to bucket NEW · Use this model

Model card · Files xet · Community 6

Qwen3.8-2.4T-A95B

Qwen Studio

> This repository contains model weights and configuration files for the post-trained model in the Hugging Face Transformers format.
>
> These artifacts are compatible with vLLM, SGLang, TokenSpeed, etc.

> For users seeking managed, scalable inference without infrastructure maintenance, the official Qwen API service is provided by Qwen Cloud.
>
> In particular, **Qwen3.8-Max** is the official version based on Qwen3.8-2.4T-A95B with more features, such as vision input & non-thinking support, 1M context length by default, official built-in tools, etc. For more information, please refer to the Qwen3.8-Max Overview.

Following the widespread community adoption of the Qwen3.5 and Qwen3.6 series, we are pleased to introduce Qwen3.8, the most capable generation in the Qwen open-model family to date.

For the first time, Qwen3.8 brings a Qwen-Max-class model to open release. Built on the architectural foundation of Qwen3.5, Qwen3.8 delivers substantial gains across coding, professional work, research, and long-horizon agentic tasks. Beyond answering harder questions, Qwen3.8 is designed to carry complex, multi-step tasks through to completion with greater reliability.

Qwen3.8 Highlights

Qwen3.8 features the following enhancements:

- **Core Capabilities**: Comprehensive improvements across coding, professional work, research, and long-horizon agentic tasks.
- **Agent Execution**: Stronger autonomous planning and better handling of environment feedback, leading to more reliable end-to-end task completion.
- **Downstream Compatibility**: Broader support for popular harnesses and development tools, making it easier to integrate into your existing stack.
- **Flexible Thinking Control**: Reasoning depth can be tuned with `reasoning_effort`, and reasoning context from historical messages is retained via `preserve_thinking`.

For more details, please refer to our blog post Qwen3.8-Max.

Model Overview

- Type: Causal Language Model
- Training Stage: Pre-training & Post-training
- Language Model
  - Number of Parameters: 2.4T in total and 95B activated
  - Hidden Dimension: 8192
  - Token Embedding: 248,320 (Padded)
  - Number of Layers: 92
  - Hidden Layout: 23 × (3 × (Gated DeltaNet → MoE) → 1 × (Gated Attention → MoE))
  - Gated DeltaNet:
    - Number of Linear Attention Heads: 128 for V and 16 for QK
    - Head Dimension: 128
  - Gated Attention:
    - Number of Attention Heads: 64 for Q and 4 for KV
    - Head Dimension: 256
    - Rotary Position Embedding Dimension: 64
  - Mixture of Experts:
    - Number of Experts: 512
    - Number of Activated Experts: 10 Routed + 1 Shared
    - Expert Intermediate Dimension: 2048
  - LM Output: 248,320 (Padded)
  - MTP (Multi-Token Prediction): trained with multiple steps
- Context Length: 262,144 natively and extensible up to 1,010,000 tokens.

Benchmark Results

	Opus 4.8	Fable 5	GPT 5.6 Sol (max)	Qwen3.7-Max
Coding Agent				
Terminal Bench 2.1	84.6	84.6	88.8	74.5
SWE-bench Pro	69.2	80.0	64.6	60.6
DeepSWE 1.1	59.0	70.0	73.0	21.6
NL2Repo-Bench	69.4	--	--	47.2
FrontierSWE	70.0	88.8	--	40.7
MLS-Bench-Lite	42.8	49.9	46.2	31.7
PaperBench	80.3	88.8	90.5	64.8
AndroidBench	69.8	84.5	74.0	56.5
QwenSWEBench	84.0	86.3	73.5	63.4
QwenQoderBench	62.7	63.1	53.8	36.8
QwenReactBench	1694	1770	1564	1538
QwenSVGBench	1648	1690	1758	1499
General Agent				
CoWorkBench	72.3	75.9	71.5	64.6
WorkSpaceBench	66.8	68.7	65.6	61.4
JobBench	48.4	57.4	45.4	31.3
SkillsBench	65.1	70.9	73.5	61.2
Agents' Last Exam (Pass / Score)	27.0 / 45.1	-- / --	30.6 / 53.6	11.8 / 31.1
Automation-Bench (Pass@1)	27.2	29.1	29.7	14.2
Toolathlon Verified (Pass@1)	76.2	77.9	74.9	49.7
WideSearch	72.9	81.2	--	75.2
HLE w/ tools	57.9	64.5	58.0	53.5
General Capabilities				
GPQA Diamond	92.0	92.6	94.1	92.4
HLE	45.7	53.3	47.2	41.4
IFBench	62.2	63.5	72.7	79.1
$OneMillion-Bench (expert score)	41.8	55.9	53.8	44.4
HealthBench	52.4	--	55.3	54.5
PLawBench	69.6	70.2	72.3	58.9
PRBench-Legal	52.7	57.6	57.6	48.5
PRBench-Finance	51.9	55.8	55.5	46.8
MRCR v2 256K (8-needle)	83.2	--	93.8	86.7
LongBench v2	69.1	--	67.1	65.3

1. Fable5 results may involve fallbacks.
2. Terminal Bench 2.1: Evaluated with Claude Code (avg@10), using a 5-hour timeout and max_tokens=131,072. For all other models, we report the best published score across harnesses: Claude Opus 4.8 and Claude Fable 5 with Terminus 2 from Artificial Analysis (https://artificialanalysis.ai/evaluations/terminalbench-v2-1); GPT-5.6 Sol with Codex (https://openai.com/index/previewing-gpt-5-6-sol/).
3. SWE-bench Pro: Evaluated with the Claude Code harness, temp=1.0, top_p=0.95, and a 256K context window. Problematic tasks corrected and all baselines evaluated on the refined benchmark.
4. DeepSWE 1.1: Evaluated with the Claude Code and mini-SWE-agent harnesses, temp=1.0, top_p=0.95, and a 256K context window. We report the highest score among both harnesses; notably, Qwen3.8-Max performs best on Claude Code.
5. NL2Repo-Bench: Evaluated with the Claude Code harness. To prevent reward hacking, we disable Bash commands that attempt to access the specific repository, such as pip download, pip install, and git clone.
6. FrontierSWE: Evaluated with the Claude Code harness. All other available MEAN@5 results are taken from the official FrontierSWE leaderboard (https://www.frontierswe.com) as of August 3, 2026. Dominance scores are recomputed from the raw scores using the official evaluation script. "--" indicates that no official MEAN@5 result was available as of that date.
7. MLS-Bench-Lite: Evaluated with Claude Code using a 5-hour timeout and max_tokens=131,072. All other model scores are taken from the official leaderboard.
8. PaperBench: Evaluated in the BasicAgent setting under Code-Dev mode, judged by Claude Opus 4.6, and averaged over 3 runs (max 12 hours per run).
9. AndroidBench: Evaluated on the 95-task public subset, reporting avg@3 scores.
10. QwenSWEBench: Inhouse coding benchmark to evaluate models' software engineering capabilities. Evaluated with the Claude Code harness. Reporting avg@3 with an 8-hour timeout, max_tokens=32,768, temperature=1.0, and a 256K-token context window.
11. QwenQoderBench: Inhouse coding benchmark to evaluate user experience on Qoder. Evaluated with the Claude Code harness. Reporting avg@5 with a 6-hour timeout, max_tokens=32,768, temperature=1.0, and a 256K-token context window.
12. QwenReactBench: Inhouse React project building benchmark using Claude Code as the harness, bilingual (EN/CN), 7 categories; auto-render + multimodal judge; BT/Elo rating.
13. QwenSVGBench: Inhouse SVG code generation benchmark; bilingual (EN/CN), auto-render + multimodal judge; BT/Elo rating.
14. CoWorkBench: Inhouse cowork benchmark for evaluating long-horizon tasks across computer science, finance, law, medical, and other productivity domains.
15. SkillsBench: Evaluated on the public SkillsBench v1.1 benchmark across 87 tasks, reporting the average score over three runs per task. Opus 4.8 and Fable 5 are evaluated on Claude Code; GPT-5.6 Sol is evaluated on Codex; the Qwen-series are evaluated on OpenCode. All results are from our own testing.
16. Automation-Bench: Evaluated on the 600-task public subset.
17. WideSearch: Evaluated with the Claude Code harness for external models and the Qwen-Agent harness for ours, reporting the average item-F1 over four runs.
18. $OneMillion-Bench: Evaluated using gemini-3.1-pro-preview.
19. PLawBench: Evaluated using gemini-3.1-pro-preview.
20. Empty cells (--): Scores are not yet available or are not applicable.

Quickstart

For streamlined integration, we recommend using Qwen3.8 via APIs.

Serving Qwen3.8

> Inference efficiency and throughput vary significantly across frameworks. We recommend using the latest framework versions to ensure optimal performance and compatibility. For production workloads or high-throughput scenarios, dedicated serving engines such as SGLang, vLLM, or TokenSpeed are recommended.

Qwen3.8 can be deployed with popular inference frameworks, e.g.:

- SGLang: Qwen3.8 Cookbook
- vLLM: Qwen3.8 Recipe
- TokenSpeed: Qwen3.8 Recipe

API Usage

> Qwen3.8-2.4T-A95B is a text-only model that requires thinking mode for all interactions. Multimodal inputs are not supported, and thinking cannot be disabled. Every response will automatically begin with reasoning enclosed in <think>\n...</think>\n\n before the final output.

> We recommend using the following set of sampling parameters for generation:
>
> - `temperature=1.0, top_p=0.95, top_k=20, min_p=0.0, presence_penalty=0.0, repetition_penalty=1.0`
>
> Please note that the support for sampling parameters varies according to inference frameworks.

Qwen3.8 comes with official support for `reasoning_effort`, which can be used to adjust reasoning depth and control cost:

- `xhigh` (default): for complex tasks demanding thorough analysis
- `medium`: balancing accuracy and speed
- `low`: efficient reasoning optimizing for speed and cost

In addition, `preserve_thinking` is enabled by default for all workloads for the best out-of-the-box experience.

Chat Completions API

The Chat Completions API can be used with most inference frameworks, as well as Qwen Cloud. Before starting, make sure the OpenAI Python SDK is installed and the API key and the API base URL are configured, e.g.:

```
pip install -U openai

# Set the following accordingly
export OPENAI_BASE_URL='your-base-url'
export OPENAI_API_KEY='your-api-key'
```

Text-Only Input

```
from openai import OpenAI
# Configured by environment variables
client = OpenAI()

messages = [{"role": "user", "content": "W

completion = client.chat.completions.creat
    model="Qwen/Qwen3.8-2.4T-A95B",
    messages=messages,
    extra_body={
        "chat_template_kwargs": {
            "enable_thinking": True,  # on
            "preserve_thinking": True, # o
        },
    },
    reasoning_effort="xhigh",  # xhigh by
    stream=True,
    stream_options={"include_usage": True}
)

reasoning_content = ""
answer_content = ""
is_answering = False
print("\n" + "=" * 20 + "Reasoning" + "="
```

```
for chunk in completion:
    if not chunk.choices:
        print("\nUsage:")
        print(chunk.usage)
        continue

    delta = chunk.choices[0].delta

    if hasattr(delta, 'reasoning_content'):
        if not is_answering:
            print(delta.reasoning_content,
        reasoning_content += delta.reasoni

    if hasattr(delta, "content") and delta
        if not is_answering:
            print("\n" + "=" * 20 + "Answe
            is_answering = True
        print(delta.content, end="", flush
        answer_content += delta.content
```

> If you are using APIs from Qwen Cloud, in addition to changing `model`, please pass `extra_body={"enable_thinking": True, "preserve_thinking": True}` instead of `extra_body={"chat_template_kwargs": {"enable_thinking": True, "preserve_thinking": True}}`.

Best Practices

To achieve optimal performance, we recommend the following settings:

1. **Sampling Parameters**:
   - We suggest using the following set of sampling parameters:
     - `temperature=1.0, top_p=0.95, top_k=20, min_p=0.0, presence_penalty=0.0, repetition_penalty=1.0`
   - For supported frameworks, you can adjust the `presence_penalty` parameter between 0 and 2 to reduce endless repetition. However, using a higher value may occasionally result in language mixing and a slight decrease in model performance.
2. **Adequate Output Length**: To optimize performance on agentic tasks, we recommend allocating sufficient output length to allow the model to generate detailed and comprehensive responses. For frameworks that support separate token limits for internal reasoning and final outputs, we suggest the following configuration within the 1M context length:
   - **Reasoning Content:** Set the maximum output length to 262,144 tokens.
   - **Final Response:** Set the maximum output length to 131,072 tokens.

   These settings provide the necessary capacity for complex reasoning while ensuring ample space for high-quality final deliverables.

Citation

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

```
@misc{qwen3.5,
    title = {Qwen3.5-Max}: A New Bar for
    url = {https://qwen.ai/blog?id=qwen3.5
    author = {Qwen Team},
    month = {August},
    year = {2026}
}
```