vLLM 0.34 upgrade guide: extensible KV cache, new defaults and breaking changes

26 November 2026 · 4 min read · blogs.iictp.org/vllm-0-34-release-breaking-changes-upgrade-guide

Short answer: vLLM 0.34 is a release of the open-source LLM inference engine, tagged on GitHub on 18 November 2026, that turns on an extensible KV cache by default, removes the older Model Runner V1 and changes defaults that can affect memory use and latency. The project reports a larger usable cache and higher throughput at high concurrency, with slower per-token decoding on some models. Read the breaking changes before upgrading a production server.

TL;DR The short version

- vLLM 0.34, tagged on 18 November 2026, turns on an extensible KV cache by default and removes the older Model Runner V1.
- The project reports a larger usable cache and higher throughput at high concurrency, but slower per-token decoding on some models.
- Read the breaking changes and benchmark your own workload before upgrading production servers.

Read online, listen to it, or change the text size: https://blogs.iictp.org/vllm-0-34-release-breaking-changes-upgrade-guide

What is vLLM 0.34?

vLLM serves large language models behind an API. The key-value (KV) cache stores attention state for tokens already processed, so its size limits how many requests and how much context a server can hold. This release changes how that memory is sized and cleans up older code paths.

How the extensible KV cache works

- On CUDA and ROCm, cache size now comes from memory actually free after CUDA graph capture and warmup, not from an estimate.
- gpu_memory_utilization defaults to 1.0 when unset with this feature on.
- An explicit value between 0.8 and 1.0 that withholds 1 GiB or more logs a warning.
- To opt out, use --gpu-memory-utilization 0.9 or --no-enable-extensible-kv-cache.

What is new and what is removed

- Model Runner V1 is removed; setting VLLM_USE_V2_MODEL_RUNNER to anything but 1 fails at startup, and V1 logits processors must be ported.
- Default --block-size is now 64 instead of 16, though backends with a preferred size and explicit flags still apply.

- The QuTLASS fp_quant method is removed, so those checkpoints no longer load; Marlin W4A8 paths are removed in favour of another option.
- huggingface_hub 2.0.0 or later and mistral-common 1.13.0 or later are required.
- ROCm release images are ROCm 10.0 only.
- Speculative decoding gains a plugin interface for custom drafting methods.
- New models include Mistral Large 4.

Results reported by the project

- Larger cache: for MiniMax-M3 on H200, from 433,000 to 672,000 tokens.
- Trade-off: time per output token rose 6% to 18% on some H200 models at concurrency 128.
- Several hardware-specific gains, such as GLM-5.3 time to first token 1.24 to 1.40 times faster at long prompts on GB200.
- These figures are the project's own and depend on model, hardware and load.

Limitations and open questions

- A larger cache can leave too little memory for other allocations. A Hugging Face experiment reports an out-of-memory failure with a draft model until utilisation was lowered.
- The TRL project documents the same new default causing out-of-memory on weight sync and now passes 0.9.
- The release notes say Kimi K3.1 groundwork landed but that model is not servable yet. Sources were read on 26 November 2026. Products, rules and standards change, so check the linked originals before you rely on any detail.

How to upgrade safely

1. Search your launch scripts for removed flags, environment variables, quantisation methods and V1 logits processors.
2. Pin the exact version and image digest you test.
3. Set --gpu-memory-utilization explicitly, and leave headroom if you run drafters or share the GPU.
4. Compare throughput, time to first token and time per output token against your current version with the same traffic mix.
5. Set --block-size 16 if your tooling depends on the old behaviour.
6. Keep the previous image ready for rollback.

What it means for LLM serving engineers

This is not a drop-in patch. The default cache change favours throughput under load; latency-sensitive services may prefer the old setting. Decide which metric your service level objective protects, then benchmark for it. See KV cache eviction research for related background.

Frequently asked questions

- **What changes in vLLM 0.34?** An extensible KV cache by default, removal of Model Runner V1, new defaults and several removed features.
- **Why does vLLM 0.34 use more GPU memory?** The utilisation default becomes 1.0 when unset with the extensible cache.
- **Can I disable the new cache?** Yes, with the flag named in the release notes or an explicit utilisation of 0.9.
- **Will old quantised checkpoints load?** Checkpoints using the removed fp_quant method no longer load.

Related reading

- llama.cpp v1.0.0
- Inkling 2 Mini inference tuning
- LLMOps in production
- Local LLM Practitioner certification

Related credential: Local LLM Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. vLLM project. Release v0.34.0. 18 November 2026. github.com/vllm-project/vllm/releases/tag/v0.34.0
2. Hugging Face. TRL release v1.19.0. 25 November 2026. github.com/huggingface/trl/releases/tag/v1.19.0
3. Niels Rogge, Hugging Face. Hill-climbing Inkling 2 Mini inference on one RTX PRO 6000. 25 November 2026. huggingface.co/blog/inkling-2-mini-inference-hillclimb

Published by IICTP, an independent technical credential body, on 26 November 2026. The current version of this post is at https://blogs.iictp.org/vllm-0-34-release-breaking-changes-upgrade-guide. More guides: blogs.iictp.org.