What Endpoint Consolidation Saves

About 4.6× fewer GPUs for 32 endpoints at 5% busy, in-sample, and at most about 2× fewer at 33% busy

Pranav Varshney
Nodus Compute

Viswa Kotra
Nodus Compute

Abstract

An operator that hosts dedicated customer endpoints gives each one a GPU, and most of those GPUs sit idle between requests. Sharing them is the obvious saving, but how much it saves while every customer keeps its own SLO, and how fast the saving shrinks as endpoints get busier, is rarely measured on a whole fleet. Nodus, the infrastructure execution layer for AI workloads, measured it for 32 endpoints, each replaying its own production trace under its own SLO, against a dedicated GPU per endpoint measured endpoint by endpoint. With stock vLLM 0.30.0, 32 Qwen3-8B base/LoRA endpoints at about 5% busy each fit on 7 shared multi-LoRA GPUs with every endpoint inside its own SLO on the 3 in-sample seeds and on 1 of 2 fresh seeds (fresh seed 11 failed 2 endpoints reproducibly while all 32 passed alone), so about 4.6× fewer GPUs is an in-sample upper estimate at light load. At about 33% busy no measured configuration of 15 GPUs or fewer passed (at most about 2×). The gain is plain multi-LoRA consolidation plus placement, the Nodus per-GPU levers saved no GPUs, and the pre-registered 5× gate fails. The placement rules were chosen on the in-sample seeds, and the ratio of SLO-met tokens per GPU-hour equals 32/k by construction, so it restates the GPU count rather than adding evidence. A dated follow-up on a live three-GPU fleet with heavy-tailed popularity found that stock round-robin placement kept every active endpoint of 64 within its own SLO on 5 of 6 seeds and that class-wise placement gave no gain, so the placement part of the saving depends on how popularity is spread across endpoints.

1 Introduction

Fine-tuned and private model endpoints are often sold as dedicated deployments: one customer, one endpoint and at least one GPU held for that customer whether it sends requests or not. Most such endpoints are light. In the fleet measured here, each endpoint kept its own GPU busy about 5% of the time at the light load. For an operator the question is not whether sharing helps but how much it saves while every customer still meets its own latency target, and how quickly the saving shrinks as customers send more traffic.

Serving many LoRA [1] adapters of one base model on shared engines is established. S-LoRA [2] serves thousands of concurrent adapters from one GPU, Punica [3] batches requests for different adapters together and consolidates them onto fewer GPUs, dLoRA [4] moves adapters and requests across replicas, LoRAX [5] offers multi-LoRA serving as an open alternative to dedicated deployments, and Aegaeon [6] pools GPUs across many models in production and reports 82% fewer GPUs, from 1,192 to 213. Stock vLLM [7] serves many adapters per engine. This report adds no new method. It measures the fleet-level saving

on production-trace arrivals, with every endpoint judged against its own SLO and against a dedicated baseline measured endpoint by endpoint, at two loads.

The metric is GPUs for the fleet, the smallest number of GPUs on which all 32 endpoints meet their own SLO in every seed window, against 32 dedicated GPUs.

2 Experimental setup

System. Three GPUs, one per host, on the same driver at the default power limit and clocks, all running stock vLLM 0.30.0 with no change to its code. The model is Qwen3-8B [8] in BF16, and in W4A16 [9] for one ablation.

Fleet. 32 endpoints of Qwen3-8B: 9 serve the base model and 23 serve customer LoRA adapters of ranks 8 to 64 on all 7 projections. The adapters have seeded random weights rather than real fine-tunes, and prompts are random tokens with fixed answer lengths. Each endpoint replays its own segment of a production trace: 11 endpoints replay the conversation trace and 10 the code trace from the public 2023 LLM inference conversation and code traces released with Splitwise [10], 6 replay BurstGPT [11] and 5 the Mooncake conversation trace [12]. Prompts are capped at 4096 tokens and answers run 16 to 512 tokens.

SLO classes. Interactive endpoints need a p95 TTFT of at most 0.5 s and a p95 TPOT of at most 30 ms, standard endpoints 1.0 s and 50 ms, and both need at least 99% of requests completed. Relaxed endpoints need every request finished within 60 s. An endpoint passes a window only if it meets its own class, never a pooled average, and a fleet configuration passes a seed only if all 32 endpoints pass. An endpoint that missed its class even alone on a dedicated GPU in any seed was moved to the next looser class for every arm: 2 endpoints at the light load and 8 at the busy load, all replaying the code trace. The light fleet then has 6 interactive, 19 standard and 7 relaxed endpoints.

Loads. At the light load each endpoint's rate is drawn from a lognormal with a median of 2 requests per minute, 1.26 requests per second for the fleet. The busy load sends 8× each endpoint's light rate, about 10.1 requests per second for the fleet, set to approach the 35% to 46% busy readings of Nodus's own fleet telemetry. Measured on their dedicated GPUs as the union of in-flight time over the window, endpoints were 4.9% to 5.2% busy at the light load and 31% to 34% busy at the busy load. Each window has 120 s of arrivals and up to 60 s of drain, prompt content is salted per window, and the prefix-cache hit rate was 0.0 in all 188 researcher windows.

Baseline. The baseline is today's practice, each endpoint alone on its own GPU with stock vLLM in BF16. Endpoints are measured one at a time, idle gaps longer than 0.5 s are cut to 0.5 s while overlapping requests keep their real overlap, and each endpoint is charged one GPU for the whole window. After the class adjustment above, all 32 endpoints pass alone on every seed, so the baseline is exactly 32 GPUs at both loads.

Shared fleet. Endpoints share stock vLLM multi-LoRA engines, one per GPU, with up to 32 adapters resident and a KV cache of 37,232 tokens. Interactive endpoints never share a GPU with standard or relaxed traffic, short-prompt interactive endpoints are kept apart from long-prompt ones, standard and relaxed endpoints are packed by offered load, an endpoint with less than 10% headroom on its dedicated GPU is split across two GPUs, and each class is sized on its own. The Nodus per-GPU levers were measured as additions on identical partitions: W4A16 weights, SLO-class priority scheduling, a TPOT shield from an earlier Nodus report and a new class gate that holds back looser prefills while a tighter request waits.

Search and gates. Each class gets the smallest number of GPUs whose windows pass on seeds 1, 2 and 3, and the fleet's count is the sum. The pre-registered gate required at least 5× fewer GPUs than dedicated at both loads. A second gate required the per-GPU levers to cut GPUs by at least 1.5× at the busy load.

Table 1: GPUs for the fleet of 32 endpoints, and whether all 32 endpoints met their own SLO. Light is about 5% busy per endpoint and busy about 33%. Fresh seeds 11 and 12 were run by the verifier.

Load	GPUs	Configuration	Seeds 1 to 3	Fresh seeds
Light	32	Dedicated, one GPU each	pass, pass, pass	pass, pass
Light	2 to 5	Shared, round-robin	fail on seed 1	not run
Light	4	Shared, by SLO class	fail on seed 3 and a repeat	not run
Light	6	Shared, by SLO class	fail on seed 2 or a repeat	not run
Light	7	Shared, by SLO class	pass, pass, pass	fail, pass
Busy	32	Dedicated, one GPU each	pass, pass, pass	not run
Busy	11	Shared, by SLO class	fail, fail, fail	not run
Busy	13	Shared, by SLO class	seed 3 only, fail	not run
Busy	15	Shared, by SLO class	seed 3 only, fail	not run

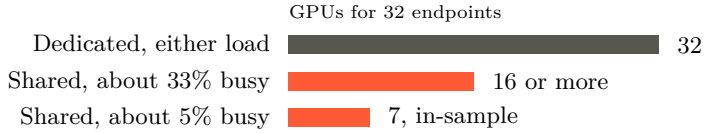


Figure 1: GPUs needed for 32 endpoints with every endpoint within its own SLO. The light shared count passed the 3 in-sample seeds and 1 of 2 fresh seeds. At the busy load no measured count of 15 or fewer passed.

3 Results

Light load: 7 GPUs in-sample. At about 5% busy, 7 GPUs held all 32 endpoints within their own SLO on seeds 1, 2 and 3: 1 GPU for the 3 short-prompt interactive endpoints, 2 for the 3 long-prompt interactive endpoints, one of which needs a GPU to itself, and 4 for the 26 standard and relaxed endpoints. That is 32/7, 4.57× fewer GPUs than dedicated. Every smaller count measured failed at least one in-sample seed or repeat, including both splits of 6 GPUs, one by 7 ms of TTFT on seed 2 and one on a same-seed repeat on another GPU. Stock multi-LoRA with round-robin placement failed at 2, 3, 4 and 5 GPUs. On a fresh seed the 7 GPUs failed 2 endpoints, as Section 4 shows, so 4.57× is an in-sample upper estimate.

The token metric restates the GPU count. SLO-met output tokens per GPU-hour rose from 13,763, 13,480 and 13,582 on dedicated GPUs to 62,919, 61,556 and 62,091 shared, ratios of 4.571, 4.566 and 4.571. When every endpoint passes in both arms, the arms serve the same deterministic traffic, so this ratio equals 32/k by construction and adds no evidence beyond the GPU count.

Busy load: at most about 2×. At about 33% busy, no measured configuration of 15 GPUs or fewer kept all 32 endpoints within SLO. The interactive class needed 3 GPUs. The standard and relaxed class failed with 8 GPUs on all 3 seeds and with 10 and 12 GPUs on seed 3, the only seed measured at those counts, so the best measured fleet of 15 GPUs left 3 endpoints out of SLO. The fleet needs at least 16 GPUs, at most $32/16 = 2.0\times$ fewer than dedicated, if adding GPUs never breaks a passing class, and counts of 9, 11 and 13 or more for that class were not measured. At 15 GPUs on seed 3 the shared fleet produced 251,488 SLO-met tokens per GPU-hour against 119,933 dedicated, 2.10×, with those 3 endpoints out of SLO.

Why the saving collapses. The shared GPUs at light load were far from full, at 1.7% to 58.0% mean utilization and 77.2 to 186.0 W per window, under half the GPU's power limit. The limit was each endpoint's tail latency under its own bursts: with 4 to 30 requests per endpoint per window, the p95 is close to the maximum, so a single burst decides an endpoint's verdict. At the busy load the binding traffic was code-trace bursts, one relaxed endpoint sending 500 to 700 requests in a 120 s window, and long-context prompts near the 4096-token cap under the standard 1 s and 50 ms targets. Two or three such endpoints

already break each other's SLO on one GPU. Eight times the traffic per endpoint cut the saving from about 4.6× to at most about 2×.

The levers saved no GPUs. Every GPU saved came from consolidation on stock multi-LoRA engines plus placement. Six of the 7 light GPUs ran the stock engine, and the seventh, the short-prompt interactive GPU with 19 to 30 requests per window, was measured only with the levers on. No configuration needed fewer GPUs with the levers than without them, so the per-GPU factor is 1.0 and the second gate failed too. SLO-class priority, the TPOT shield and the class gate failed the busy long-prompt interactive pair on 3 of 3 seeds against 1 of 3 for the stock engine, and the single light interactive GPU on 2 of 3 against 1 of 3. W4A16 weights prefilled more slowly on this GPU: one bursty endpoint's TTFT rose to 0.54 to 1.05 s against 0.40 to 0.94 s in BF16, and with 8 GPUs for the busy standard and relaxed class it left 3 endpoints failing on seed 1 against 1 in BF16. Its 3× larger KV cache, 110,400 tokens against 37,232, never mattered, with at most 14 preemptions per busy window and none at light load. Placement did matter: every round-robin consolidation at 5 GPUs or fewer failed, and every interactive failure traced to sharing a GPU with long-prompt prefill.

Noise and choices made after the data. The dedicated baseline re-measured with each half on the other host gave identical verdicts for all 32 endpoints and 14,681 SLO-met tokens both times. The standard and relaxed class on 4 GPUs repeated with identical tokens, 11,362, while on 3 GPUs it failed its repeat, which is why that class got 4 GPUs. The placement rules, the engine profile per class and the choice of 4 over 3 GPUs for standard and relaxed endpoints were all made after seeing seeds 1 to 3, in 10 timestamped addenda to the pre-registration, and the planned out-of-sample seed was dropped for time. The registered stack as first written failed at 1 GPU at light load and at 3 GPUs at busy load. The 7-GPU result is therefore in-sample. Under the originally registered baseline rule, one extra dedicated GPU for each endpoint that fails alone, the baseline would be 34 GPUs and the light ratio $34/7 = 4.86$, still below the 5× gate, which failed at both loads.

4 Independent verification

A separate verifier re-judged every endpoint of the light headline and of the busy 15-GPU windows from the per-request logs with its own code, matched every number, and confirmed identical per-endpoint traffic in both arms. It then ran the exact 7-GPU configuration and the dedicated baseline on fresh seeds 11 and 12, never used before, with the same image and the researcher's code, changed only in its workspace path and container name. Seed 12 passed, with the worst endpoint at 0.74 of its limit and a ratio of 4.57. Seed 11 failed 2 endpoints while all 32 passed alone. A standard endpoint reached a p95 TPOT of 70.8 ms against its 50 ms target when two requests with answers of 16 tokens and prompts of 4078 and 4096 tokens arrived 0.11 s apart beside 8 other endpoints, against 35.1 ms alone. An interactive endpoint reached a p95 TTFT of 0.504 s against 0.5 s. A same-seed repeat 16 minutes later failed the same endpoint at 70.7 ms with identical SLO-met tokens, 2,645, so the failure is a property of the seed's traffic, not noise. Of 5 seeds measured, 7 GPUs passed 4 and failed 1, so by the study's own rule 7 GPUs is not a passing count.

The verifier also found that the long-prompt interactive endpoint that the shared fleet places alone on a GPU had a 15% to 16% higher mean TPOT there than on its dedicated baseline GPU, at mean SM clocks of 1157 to 1176 MHz against 1408 to 1409 MHz. The cause is either the larger adapter configuration or the compressed baseline replay keeping its GPU busier and its clocks higher, and the idle-gap compression was never checked against a real-time replay. If anything, that flatters the dedicated baseline. The verifier classed the result as PARTIAL: the direction holds, the 7-GPU claim does not hold out of sample, the gates fail, and the method is established in S-LoRA [2], Punica [3], dLoRA [4], LoRAX [5] and Aegaeon [6]. What is new is only the measurement, a per-endpoint-SLO GPU multiplier on production-trace arrivals against a dedicated baseline measured endpoint by endpoint.

5 Follow-up, 6 November 2026: a three-GPU fleet with heavy-tailed popularity

Setup. A later live measurement served one fleet of endpoints on three GPUs, one per host, with all three GPUs serving at once, each running one stock vLLM 0.30.0 multi-LoRA engine. The endpoints are Qwen3-8B base and LoRA endpoints drawn as in Section 2, and the first 32 are this report's 32 endpoints at new rates. Per-endpoint popularity takes its shape from the public ServeGen [13] per-client data, 162 production API clients with 14 days of rates each, in which the top 10% of clients carry 75% of requests. The scale is set so the mean matches this report's light load, 2.37 requests per minute per endpoint. With 64 endpoints the busiest 5 carry 63% of the load and 44 of the 64 send less than 1 request per minute. Each endpoint is judged against its own SLO class from Section 2 over 120 s windows, and a window passes only if every endpoint that sent a request passes. Two placements ran on the same engines, interleaved window by window: stock round-robin, which puts endpoint i on GPU i mod 3, and this report's class-wise placement, fixed in advance. The pre-registered gate required class-wise placement to hold at least 1.5× as many endpoints as round-robin.

Result. Round-robin placement kept every active endpoint inside its own SLO with 64 endpoints on 3 GPUs on 5 of 6 distinct seeds, 1, 2, 3, 11 and 12, and same-seed repeats of seeds 1 and 11 passed as well. That is about 20 endpoints per GPU against one GPU per endpoint, almost all of it from consolidating idle endpoints, since only 26 to 32 of the 64 sent any request in a window. Fresh seed 13 failed in both of its runs on one interactive endpoint, ep53. A single request with a 3601-token prompt reached a TTFT of 0.557 s and 0.551 s against the 0.5 s target, and 0.260 s with ep53 alone on a dedicated GPU. With 80 endpoints, round-robin failed on every seed tested, 1, 11 and 12, though on seed 11 only on an endpoint that also missed its class alone. Class-wise placement gave no gain. It held 32 endpoints on fresh seeds 11, 12 and 13 but failed at 32 on seed 3, where ep28 reached a p95 TPOT of 60.8 ms against its 50 ms target, and failed at 40, 48 and 64 on seed 1, each time on one bursty standard endpoint, ep25. The gate failed. Two variants registered after that failure, an interference-aware placement and engine step control under round-robin routing, did not beat round-robin either.

Table 2: Live three-GPU fleet with heavy-tailed popularity: whether every active endpoint met its own SLO, by placement and number of endpoints. Seeds 1 to 3 were the researcher's and fresh seeds 11 to 13 the verifier's. Verdicts use the final classes, after ep28 was moved to standard.

Placement	Endpoints	Seeds 1 to 3	Fresh seeds 11 to 13
Round-robin	64	pass, pass, pass	pass, pass, fail
Round-robin	80	seed 1 only, fail	fail, fail, not run
Round-robin	96 and 128	seed 1 only, fail	not run
By SLO class	16	pass, pass, pass	pass, pass, pass
By SLO class	32	pass, pass, fail	pass, pass, pass
By SLO class	40, 48 and 64	seed 1 only, fail	not run

Independent verification. The verifier ran both placements on fresh seeds 11, 12 and 13 with the researcher's code unchanged, re-judged every window from the per-request logs with its own code and reproduced every verdict. It rated the result PARTIAL and not publishable as a report of its own. The result with 64 endpoints does not hold on every seed: it fails fresh seed 13 reproducibly, on an endpoint that passes alone. It also relies on a class change made after the data. Endpoint ep28 missed its interactive class even alone on seeds 2 and 3, with a p95 TPOT of 44.7 ms and 34.7 ms against 30 ms, so by this report's rule it was moved to standard for both placements. That rule was registered after round-robin with 64 endpoints had failed seed 2 only on ep28, and under the classes as scheduled round-robin would have held only 16. Each window is a 120 s snapshot in which the median active endpoint sends 3 requests, so its p95 is close to its maximum and one long prompt can decide its verdict. The researcher's reading that class-wise placement held only 16 endpoints, 0.25× of round-robin, also rests on one seed, and on fresh seeds the ratio is at least 0.5. The 1.5× gate fails in every reading.

Reading. The value of placement depends on the shape of popularity. On this report's fleet of 32 endpoints at similar rates, every interactive failure traced to sharing a GPU with long-prompt prefill, round-robin failed at every count measured, 2 to 5 GPUs, and class-wise placement held the fleet on 7. With heavy-tailed popularity over more endpoints, most endpoints are near idle, and pooling the few busy ones across all GPUs beats partitioning the GPUs by class. In the class-wise window with 64 endpoints on seed 1, the GPU reserved for the 14 interactive endpoints received 20 of the 242 requests, while the two GPUs left for standard and relaxed traffic received 100 and 122, and ep25 failed on one of them. Round-robin spread the same 242 requests 88, 59 and 95 and passed. Nodus should not treat class-wise placement as a general rule. The closest prior art, LoRAServe [14], places and routes LoRA adapters across a cluster using adapter rank and arrival history on production traces and reports up to 50% fewer GPUs under SLO constraints. That does not conflict with this result, since its placement differs from the class-wise rule tested here.

6 Conclusion

For an operator that hosts dedicated customer endpoints, consolidating them onto shared multi-LoRA engines is a large saving when endpoints are light and a modest one when they are busy. With stock vLLM 0.30.0, 32 Qwen3-8B endpoints at about 5% busy fit on 7 shared GPUs within every endpoint's SLO on the 3 in-sample seeds and on 1 of 2 fresh seeds, so about 4.6× fewer GPUs is an in-sample upper estimate. At about 33% busy no measured configuration of 15 GPUs or fewer passed, at most about 2×. The saving came from plain multi-LoRA consolidation, and on this fleet placement that keeps interactive endpoints away from long prompts also mattered. The follow-up in Section 5 found no gain from that class-wise placement on a three-GPU fleet with heavy-tailed popularity, where round-robin did at least as well, so placement is not a general source of the saving. The Nodus per-GPU levers saved no GPUs, and the pre-registered 5× gate failed. The study covers one GPU type, one model, one fleet of 32 endpoints with random-weight adapters and 120 s windows. A fleet plan that applies the light-load saving to endpoints at production busy levels will need more than twice the GPUs it planned.

References

[1] E. J. Hu et al. LoRA: Low-Rank Adaptation of Large Language Models. ICLR 2022.

[2] Y. Sheng et al. S-LoRA: Serving Thousands of Concurrent LoRA Adapters. MLSys 2024.

[3] L. Chen et al. Punica: Multi-Tenant LoRA Serving. MLSys 2024.

[4] B. Wu et al. dLoRA: Dynamically Orchestrating Requests and Adapters for LoRA LLM Serving. OSDI 2024.

[5] Predibase. LoRAX: Multi-LoRA Inference Server that Scales to Thousands of Fine-Tuned LLMs. Open-source software, 2023.

[6] Y. Xiang et al. Aegaeon: Effective GPU Pooling for Concurrent LLM Serving on the Market. SOSP 2025.

[7] W. Kwon et al. Efficient Memory Management for Large Language Model Serving with PagedAttention. SOSP 2023.

[8] A. Yang et al. Qwen3 Technical Report. arXiv preprint, 2025.

[9] J. Lin et al. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. MLSys 2024.

[10] P. Patel et al. Splitwise: Efficient Generative LLM Inference Using Phase Splitting. ISCA 2024.

[11] Y. Wang et al. BurstGPT: A Real-World Workload Dataset to Optimize LLM Serving Systems. KDD 2025.

[12] R. Qin et al. Mooncake: A KVCache-centric Disaggregated Architecture for LLM Serving. FAST 2025.

[13] Y. Xiang et al. ServeGen: Workload Characterization and Generation of Large Language Model Serving in Production. NSDI 2026.

[14] S. Jaiswal et al. Serving Heterogeneous LoRA Adapters in Distributed LLM Inference Systems. arXiv preprint arXiv:2511.22880, 2025.