Encoder Sidecars on LLM Serving GPUs

One serving GPU did the work of 1.5635 dedicated GPUs with an embedding sidecar and 1.6133 with a reranker, with the live SLO held

Pranav Varshney
Nodus Compute

Viswa Kotra
Nodus Compute

Abstract

Retrieval-augmented applications run two kinds of model, an LLM that answers users and encoder models that embed and rerank documents. Teams usually give each its own GPU, and the serving GPU sits partly idle between bursts of traffic. In this study, a research prototype from Nodus, the infrastructure execution layer for AI workloads, ran the encoder job on the serving GPU itself, as a separate process behind a guard that read only the serving engine's own metrics. On one GPU serving Qwen3-8B with stock vLLM at an interactive SLO, across 3 seeds, the GPU delivers 1.5635 dedicated-GPU equivalents with an embedding sidecar and 1.6133 with a reranking sidecar. Live SLO attainment is 1.0000 with the embedding sidecar and at least 0.9973 with the reranker. At 28.8 req/s the guard still delivers 1.3710 with attainment of at least 0.9960, while the best static setting at normal GPU priority, tuned at 9.6 req/s, falls to an attainment of 0.0851. An ablation shows that running the encoder at below-normal GPU priority holds the SLO at both loads on its own, and the guard adds throughput on top. Co-location without either misses the SLO on every request. An independent rerun on fresh seeds reproduced the result.

1 Introduction

A GPU that serves interactive LLM traffic is sized for bursts, so between bursts much of it waits. Retrieval-augmented applications also need encoder models that embed documents and rerank search results, and that work usually runs on a second fleet of GPUs. Both fleets are paid for in full, even though the serving fleet has spare cycles most of the time.

We measure what happens when a Nodus research prototype moves the encoder job onto the serving GPU. The LLM server is stock vLLM [1] serving Qwen3-8B [2] with no change to the engine. The encoder runs as its own process beside it, either the Qwen3-Embedding-0.6B embedding model [3] or the bge-reranker-v2-m3 cross-encoder [4]. A guard decides how much encoder work may run at each moment and yields the GPU to live requests as their latency rises. Prior work shares a GPU between a latency-critical job and best-effort deep learning jobs [5] [6], resizes GPU shares for inference under latency targets [7], or fills a serving GPU with more LLM work. Here the best-effort tenant is a different model family that would otherwise hold its own GPU, and the serving engine is left untouched.

The metric is dedicated-GPU equivalents. It adds the LLM's SLO-met output tokens per second, as a fraction of a dedicated LLM GPU on the same seed, to the encoder's items per second, as a fraction of a dedicated encoder GPU. A dedicated GPU scores 1.0000. Tokens from requests that missed the SLO earn nothing.

2 Experimental setup

Workload. Live requests replay a public production LLM conversation trace [8], time-scaled to 9.6 req/s, in open loop with 120 s windows. A request meets the SLO when its TTFT is at most 500 ms and its TPOT is at most 30 ms. The embedding sidecar encodes 256-token passages and the reranking sidecar scores 512-token query and passage pairs from a backlog that never runs dry. The experiment uses 3 seeds, 1, 2 and 3, and the hypotheses, arms and gate were pre-registered before the first measured run.

Arms. The dedicated LLM GPU runs the server alone, twice per seed, so that the two runs measure run-to-run noise. The dedicated encoder GPU runs the encoder alone at its fastest setting. Naive co-location runs the encoder at that same setting beside the server with no guard. The best static setting is the co-location setting that met the SLO with the most encoder throughput in a pilot sweep at 9.6 req/s. Nodus runs the encoder beside the server under the guard. All arms run on the same GPU with the same server, and the order of arms rotates across seeds.

Gate. The pre-registered gate requires a mean of at least 1.10 dedicated-GPU equivalents, live SLO attainment within 0.01 of the dedicated LLM GPU on every seed, and TPOT p99 of at most 30 ms on every seed. A second block repeats the comparison at 28.8 req/s with the same guard and the same static setting, neither re-tuned. A third block replaces the embedding model with the reranker.

3 Results

Table 1: Dedicated-GPU equivalents on one GPU, mean of 3 seeds, with live SLO attainment in parentheses.

Arm	Embedding at 9.6 req/s	Reranker at 9.6 req/s	Embedding at 28.8 req/s
Nodus, guarded sidecar	1.5635 (1.0000)	1.6133 (0.9988)	1.3710 (0.9962)
Best static setting	1.6418 (0.9955)	not run	0.5732 (0.0851)
Naive co-location	0.8330 (0.0000)	0.8290 (0.0000)	not run
Dedicated LLM GPU	1.0000 (1.0000)	1.0000 (1.0000)	1.0000 (0.9999)

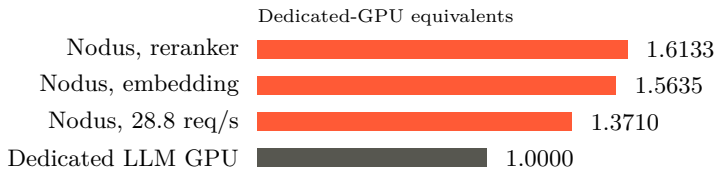


Figure 1: Dedicated-GPU equivalents delivered by one GPU, mean of 3 seeds.

One serving GPU absorbs most of an encoder GPU. With the embedding sidecar the GPU delivers 1.5575, 1.5763 and 1.5568 dedicated-GPU equivalents on the 3 seeds, with live SLO attainment of 1.0000 on every seed. The LLM's SLO-met tokens equal those of the dedicated GPU, and the sidecar adds 403.65 embeddings per second against 716.27 on a GPU of its own. With the reranker the GPU delivers 1.6070, 1.6254 and 1.6075, with attainment of 0.9992, 0.9973 and 1.0000. Both pass the pre-registered gate.

Live latency cost. The sidecar raises live TTFT p95 from between 0.0499 s and 0.0513 s to between 0.0933 s and 0.1007 s, and TPOT p99 from at most 0.00969 s to at most 0.02261 s. Both stay well inside the 500 ms and 30 ms SLO.

A static setting at normal priority does not survive a change in load. At 9.6 req/s, the load it was tuned for, the best static setting at normal GPU priority delivers 1.6418 against 1.5635 for the guard, with attainment of 0.9955 and TPOT p99 up to 0.02949 s, close to the 30 ms SLO. At 28.8 req/s the same setting falls to 0.5732 dedicated-GPU equivalents and an attainment of 0.0851, and no seed is above 0.11184. The guard throttles the sidecar

to 266.03 embeddings per second at that load and holds attainment at 0.9960 or higher, for 1.3710 dedicated-GPU equivalents.

Where the robustness comes from. The independent verifier ran an ablation the pre-registration did not include, the encoder at below-normal GPU priority with a fixed batch and no feedback. It held the SLO at both loads, with 1.3930 and 1.3933 dedicated-GPU equivalents at attainment 1.0 on seeds 11 and 12 at 9.6 req/s, and 1.2977 at attainment 0.99886 on seed 13 at 28.8 req/s. Below-normal priority is therefore what keeps live requests safe when load changes. The guard's feedback adds throughput on top, 1.5701 and 1.5676 against 1.3930 and 1.3933 on the same seeds, at the cost of live TPOT p99 rising from 0.01452 s to 0.02214 s on seed 11, still inside the SLO.

Unguarded co-location fails. Running the encoder at its fastest setting beside the server pushes TPOT p99 to 0.08417 s or more and live SLO attainment to 0.0000 on every seed, for both encoders. The capacity exists only when the encoder yields to live requests.

4 Independent verification

A separate verifier reran the comparison on the same GPU with the same code and fresh seeds it chose itself, 11 and 12 at 9.6 req/s and 13 at 28.8 req/s. The guarded embedding sidecar delivered 1.5701 and 1.5676 dedicated-GPU equivalents at 9.6 req/s, with live SLO attainment of 0.99914 and 1.0 against 1.0 on the dedicated LLM GPU. At 28.8 req/s it delivered 1.3650 with attainment 0.99542, while the static setting again fell to an attainment of 0.07237. All three reproduce the main results.

Table 2: Independent verification on fresh seeds, guarded embedding sidecar on one GPU.

Seed	Load	Dedicated-GPU equivalents	Live SLO attainment
11	9.6 req/s	1.5701	0.99914
12	9.6 req/s	1.5676	1.0
13	28.8 req/s	1.3650	0.99542
11, ablation without feedback	9.6 req/s	1.3930	1.0
13, ablation without feedback	28.8 req/s	1.2977	0.99886

5 Conclusion

On one GPU, the research prototype ran an embedding or reranking job beside a live LLM server and delivered 1.5635 and 1.6133 dedicated-GPU equivalents with the live SLO held, and 1.3710 at 28.8 req/s. The serving engine is unmodified. Below-normal GPU priority for the encoder keeps the live SLO safe across loads, the guard adds throughput on top of it, and co-location at normal priority without the guard misses the SLO on every request. The result covers one LLM and two encoders on one GPU type. For a fleet that serves an LLM and also embeds or reranks, the result suggests that much of the encoder work could move to the spare cycles of the serving GPUs instead of GPUs of its own.

References

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

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

[3] Y. Zhang et al. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. arXiv preprint, 2025.

[4] J. Chen et al. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv preprint, 2024.

[5] F. Strati, X. Ma and A. Klimovic. Orion: Interference-aware, Fine-grained GPU Sharing for ML Applications. EuroSys 2024.

[6] P. H. Coppock et al. LithOS: An Operating System for Efficient Machine Learning on GPUs. SOSP 2025.

[7] A. Dhakal, S. G. Kulkarni and K. K. Ramakrishnan. GSLICE: Controlled Spatial Sharing of GPUs for a Scalable Inference Platform. SoCC 2020.

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