Many Training Jobs on One GPU

16 jobs shared one GPU at 4.0305× for a shared-trunk sweep and 6.93× for unrelated LoRA tenants, with bit-identical results

Pranav Varshney
Nodus Compute

Viswa Kotra
Nodus Compute

Abstract

Small training jobs rarely fill a large GPU, yet each is usually given a GPU of its own. In this study, a research prototype from Nodus, the infrastructure execution layer for AI workloads, ran many of them on one GPU and kept every result exact. We report two independently audited single-GPU results. First, shared-trunk training, which is a same-data learning-rate sweep over one base model and one dataset and not unrelated tenants, runs 16 jobs at 4.0305× the throughput of dedicated runs and 3.7011× that of stock MPS, with losses bit-identical to dedicated runs and zero errors. Second, MPS100 unrelated-tenant LoRA packing runs 16 tenants at 5.86× cold, including setup, and 6.93× steady, and every per-step loss and every adapter byte matches its solo run, with bit-identical held-out loss.

1 Introduction

A paid GPU-hour earns only the training work done in it. Hyperparameter sweeps and per-customer fine-tunes consist of many small jobs, and giving each one a dedicated GPU leaves most of that GPU idle. Sharing the GPU is only useful if the shared jobs produce exactly the results they would have produced alone.

This report covers two cases with different scopes, and it absorbs report 016, Live Lab: Measuring GPU Virtualization, the lab record of these same runs, whose published comparisons and measurement rules are a section below. Shared-trunk training is a same-data learning-rate sweep over one base model and one dataset, not unrelated tenants. MPS100 unrelated-tenant LoRA packing runs LoRA jobs from unrelated tenants together on one GPU. Both were measured on real GPUs and independently audited, and the only comparison baselines are dedicated runs, stock MPS and solo runs as stated below.

2 Methodology

Shared-trunk training. N jobs of the same sweep, with N of 16 or 8, run together on one GPU and are compared with the same jobs on dedicated runs and under stock MPS. Correctness requires losses bit-identical to the dedicated runs and zero errors. Configurations and variants are labelled as in the audited results, and each result is reported for the GPU it was measured on.

LoRA packing. 16 unrelated-tenant LoRA jobs run together on one GPU as MPS clients at MPS100, the lab's name for a 100% active-thread setting per client, so no job's share of compute is capped. Cold throughput includes setup, and steady throughput excludes it. Correctness requires every per-step loss and every adapter byte to match the job's solo run, and held-out loss to be bit-identical.

Host names are replaced with neutral labels. Results from different GPU types are reported separately and never pooled.

3 Results

Table 1: Shared-trunk training, a same-data learning-rate sweep over one base model and one dataset, not unrelated tenants. Losses bit-identical to dedicated runs, zero errors. Audited.

GPU	Variant	vs dedicated	vs stock MPS
Host B, GPU 3	paired streams	4.092×	n/a
Host A, GPU 3	N16 top-1	4.0305×	3.7011×
Host A, GPU 2	overlap	4.024×	n/a
Host A, GPU 1	N16 top-1	4.0010×	3.6685×
Host B, GPU 2	overlap	4.004×	n/a
Host A, GPU 0	N16 top-2	3.3794×	2.9779×
Host B, GPU 3	N8 top-1	3.3830×	2.9685×

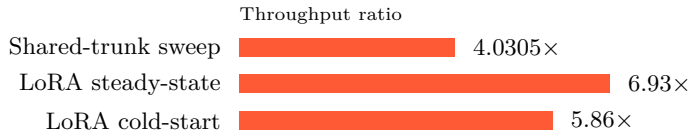


Figure 1: Measured throughput ratios against sequential or dedicated execution on the same GPU.

Across both top-1 runs at N16 the gain over dedicated runs is 4.0305× and 4.0010×, and the gain over stock MPS is 3.7011× and 3.6685×. Losses are bit-identical to the dedicated runs in every case and there are zero errors.

32 jobs on Host A (independently audited). A further shared-trunk run with N32 and J32 on Host A passed every gate with zero errors and no audit mismatches. Scope: matched top-1-layer LoRA hyperparameter cohorts, NOT all-layer LoRA or a foreground 5% slowdown claim.

Table 2: Shared-trunk N32 J32 on Host A, against 32 dedicated runs. Matched top-1-layer LoRA hyperparameter cohorts, NOT all-layer LoRA or a foreground 5% slowdown claim. Audited, zero errors.

Comparison	Ratio
vs dedicated	4.453223805913928×
vs stock MPS	3.9610916533879577×
vs packed without reuse	4.635884941361824×

Streams, graphs and overlap (negative result). On the same-data learning-rate sweep, adding streams, graphs or overlap adds 0–2% against dedicated runs beyond the base configurations above.

The audited Host C result is 5.86× cold, including setup, and 6.93× steady. Unaudited repeat runs on other hosts are 6.18× and 7.43×, and 6.84× and 8.57×. The Host E run, 9.85× cold and 12.39× steady, is measured, not qualified, and is reported on its own and not pooled with the other hosts.

4 Execution models and accounting

The two passing training results exploit different sharing opportunities. Shared-trunk training reuses common forward work across a same-data hyperparameter cohort and must not

Table 3: Mean SM active in the same N32 J32 run.

Arm	Mean SM active
Shared trunk	0.8938308080808082
Stock MPS	0.9816929275920134
Packed without reuse	0.8101001683501684

Table 4: MPS100 unrelated-tenant LoRA packing, N16. Every per-step loss and every adapter byte matches its solo run, held-out loss bit-identical.

Host	Status	Cold, incl. setup	Steady
Host C	Audited	5.86×	6.93×
Host B	Unaudited repeat run	6.18×	7.43×
Host D	Unaudited repeat run	6.84×	8.57×
Host E (different GPU type)	Measured, not qualified	9.85×	12.39×

be described as unrelated tenants. MPS100 packing instead co-resides 16 independent LoRA tenants and compares each packed job with its own solo execution. Keeping these scopes separate prevents a reuse result from being presented as generic tenant multiplexing.

Shared-trunk throughput counts the matched cohort relative to dedicated runs and stock MPS. The unrelated-tenant campaign reports cold throughput, which includes setup, and steady throughput, which excludes setup. Host topology is part of the result: audited Host C, unaudited repeat hosts, and the measured Host E are never pooled.

Table 5: Two distinct sharing mechanisms in the consolidated training paper.

Campaign	Sharing opportunity	Baseline	Correctness gate
Shared trunk	Shared base work	Dedicated / MPS	Exact losses; 0 errors
MPS100 packing	LoRA tenants	Solo jobs	Exact loss + adapter

5 Correctness and audit protocol

Throughput is accepted only with exact result checks. In the shared-trunk campaign every reported loss matches the dedicated run bit for bit and every arm has zero errors. In unrelated-tenant packing, every per-step loss and every adapter byte matches the corresponding solo job; held-out loss is also bit-identical. These checks are stronger than comparing final scalar accuracy and make silent numerical drift visible.

The N32 J32 run passes every gate with zero audit mismatches. Its scope is matched top-1-layer LoRA hyperparameter cohorts. It is not an all-layer LoRA result and does not support a foreground 5% slowdown claim. Mean SM activity is reported as an observation, not as the objective: stock MPS reaches 0.98169 mean SM active yet delivers less useful cohort throughput than the shared trunk at 0.89383.

6 Replication, ablations, and boundaries

The N16 top-1 result repeats on two Host A GPUs at 4.0305× and 4.0010× versus dedicated, and 3.7011× and 3.6685× versus MPS. Overlap variants on Host A and Host B reach 4.024× and 4.004×. N8 top-1 reaches 3.3830× versus dedicated and 2.9685× versus MPS. The N32 result reaches 4.453223805913928× versus dedicated and 3.9610916533879577× versus MPS.

Streams, graphs, and overlap add only 0–2% beyond the base shared-trunk configurations. That negative result locates the main gain in work reuse and packing rather than generic CUDA concurrency features. For unrelated tenants, only the Host C result is audited. Host B and D are repeat measurements without the same audit status; Host E is measured but not qualified.

The paper does not claim that every training job can share one GPU without interference. It covers the stated cohort and LoRA shapes, the measured hosts, job counts, and exactness checks. Memory-heavy optimizers, arbitrary models, longer training horizons, and adversarial tenant behavior require separate measurements.

7 Published numbers for existing approaches

The figures below are published by their authors and were not measured by Nodus.

Table 6: Published results for existing GPU sharing approaches.

Source	Published result
Orion [1]	Up to 7.3× per-GPU request throughput collocating best-effort inference
Orion [1]	Up to 1.49× lower training cost than dedicated GPU allocation
HAMi [2]	Priority 0 (high) or 1 (low) per task, runtime preemption on one GPU

The current training evidence does not establish arbitrary multi-tenant isolation, foreground slowdown bounds, all-layer LoRA performance, or transfer to other accelerators. Published Orion and HAMi results provide context but are not matched arms in the Nodus experiments.

The lab treats useful work per paid GPU-hour as the top-level outcome and separates it from device activity. A run is publishable only when its workload unit, occupied GPU count, elapsed interval, correctness gate, host topology, comparator, and audit status are all explicit. SM and tensor-core activity explain execution but never substitute for completed work.

Every campaign is classified before aggregation. Same-data shared-trunk sweeps, unrelated-tenant LoRA packing, serving, and mixed-workload runs have different correctness conditions and denominators. Published third-party numbers remain labeled as published, and Nodus measurements remain labeled by host and qualification state.

8 Reproducibility

A reproduction must preserve cohort membership, model and dataset revisions, LoRA layer selection, learning rates, initialization, job count, setup inclusion, GPU topology, and baseline semantics. Dedicated, stock MPS, packed-without-reuse, and shared-trunk arms must run the same logical jobs. Cold and steady timing must remain separate.

Exactness requires retaining complete per-step loss sequences and serialized adapter bytes for comparison with solo references, not just aggregate throughput. Audit status and host identity should travel with every reported row so later summaries cannot pool results from different GPU types or qualified and unqualified runs.

9 Conclusion

On one GPU, Nodus research ran 16 training jobs together and kept every result exact. A same-data learning-rate sweep over one base model and one dataset reaches 4.0305× the throughput of dedicated runs and 3.7011× that of stock MPS, with bit-identical losses. Sixteen unrelated-tenant LoRA jobs reach 6.93× steady and 5.86× cold on the audited host, with every per-step loss and adapter byte matching solo runs. For teams running sweeps or per-customer fine-tunes, the result suggests many jobs could share one GPU without changing a single result. These results were measured without an MPS compute cap and do not cover thread-capped sharing.

References

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

[2] Project HAMi. HAMi documentation, FAQ. project-hami.io, 2026.