BEST PAPERS: SCALABLE AND ELASTIC AI SYSTEMS

SwarmLoRA: Serverless Shared-Computation Disaggregation for Isolated, High-Throughput Multi-LoRA Serving

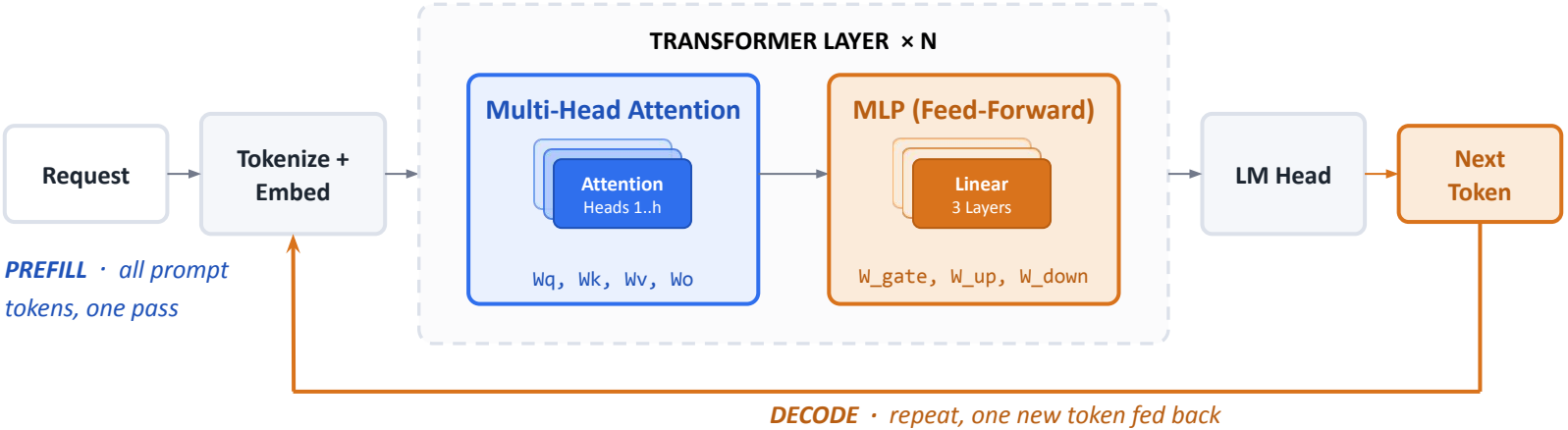
Mausam Basnet, University of North Texas; Tong Shu, University of North Texas

SC26 hpc unites | Chicago

INTRODUCTION

Large Language Models Now Power Everyday Services

Every request runs the same forward pass, tokens through embedding, N transformer layers, and a projection back to vocabulary.



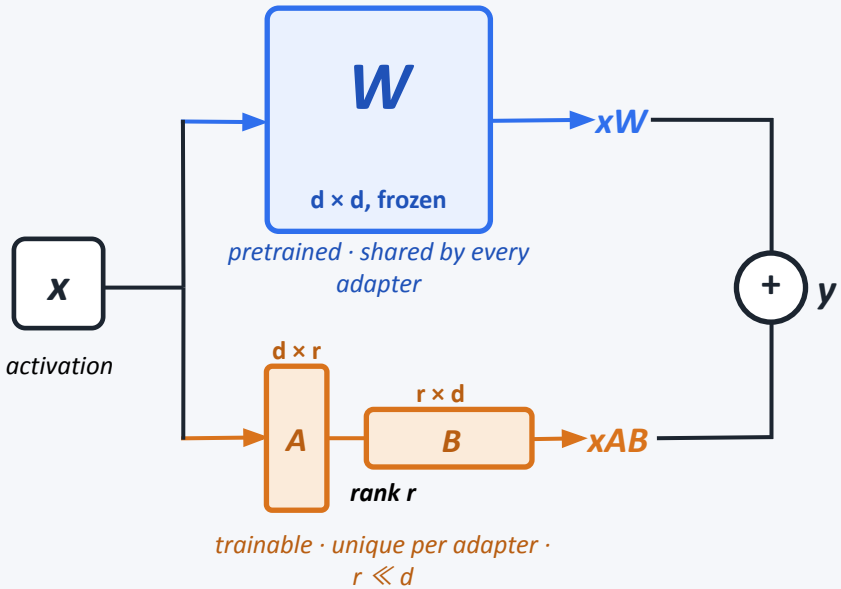
SERVED ACROSS INDUSTRIES

Healthcare | Finance | Legal | Education | E-comm | Engineering

INTRODUCTION

LoRA: Fine-Tune a Sliver, Not the Whole Model

Low-rank decomposition: freeze the pretrained weight matrix, train a rank-r bottleneck alongside it.



$$y = xW + xAB$$

10,000× fewer trainable parameters

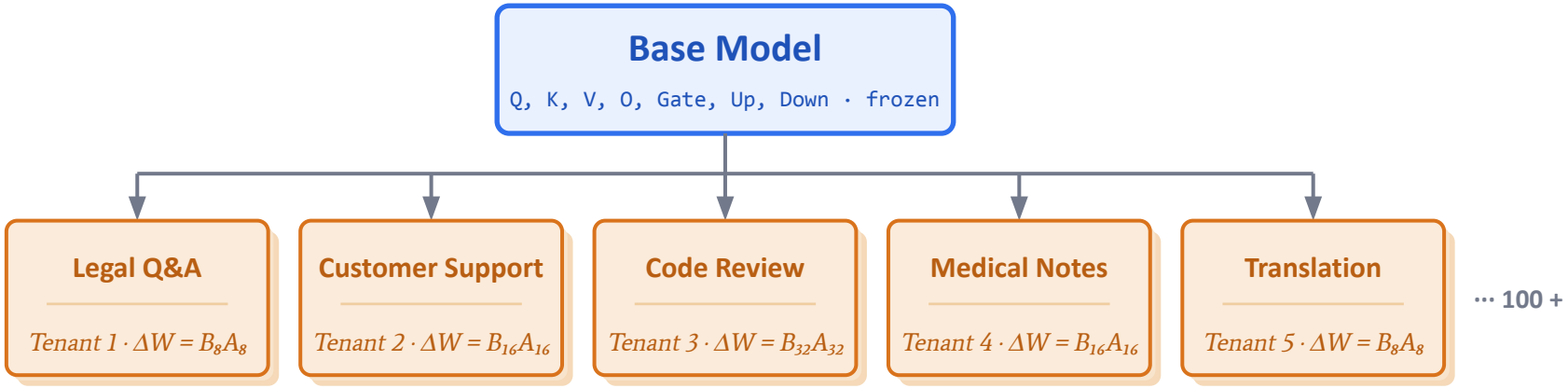
- 3× less GPU memory than full fine-tuning
- On-par quality vs. full fine-tuning
- Numbers: original LoRA paper, GPT-3 175B

Init: *A random, B zero → $\Delta W = BA = 0$. Training begins as a true no-op.*

INTRODUCTION

One Base Model, Many Tenants

Serving many LoRA adapters off one shared base model is already how the industry operates.



Already Shipping on: Hundreds of adapters served on shared GPUs cluster

Fireworks AI | Amazon SageMaker | Together AI | NVIDIA NIM

THE PROBLEM

This Is Already a Production Cost Problem

Demand is bursty, adapters multiply fast, and idle GPUs are the line item that shows up on the bill.

8×

MORE GPUS FOR BURSTY DEMAND

- 10 req/min baseline, 400 req/min spikes
- needs ~125 GPUs, not ~16 for smooth load
- ~97% of that fleet sits idle most of the time

Hebbia, "The Hidden Economics of LLM Inference," 2026

1,000s

OF ADAPTERS, ONE REAL DEPLOYMENT

- Cresta runs 1,000s of LoRA adapters, one cluster
- one per customer, contact-center AI at scale
- 100× cheaper than one model per customer

Fireworks AI / Cresta case study, 2023–24

70%+

OF MODELS SIT IDLE AT ANY MOMENT

- in a real production multi-model trace
- active model set changes 766× per hour
- static allocation cannot track that shift

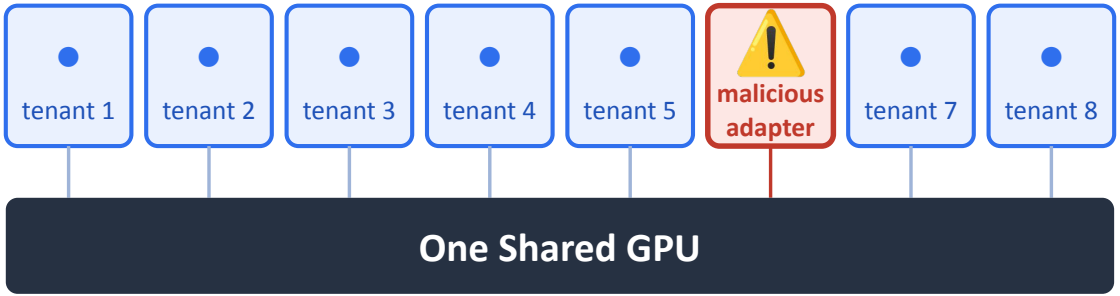
Prism, arXiv:2505.04021, 2025 (Novita AI trace)

Static, dedicated GPU allocation cannot keep up. The only affordable path is to share the GPU and stay elastic. That is exactly where existing systems break.

THE PROBLEM

Why Isolation Matters ?

Batching means many tenants share one GPU. One of them doesn't have to be honest.



DATA ISOLATION	FAULT ISOLATION
SERVERFUL	SERVERFUL
Victim data recovered	7 / 7 co-residents killed
SERVERLESS	SERVERLESS
0 bytes recovered	0 / 7 co-residents affected

SERVERFUL · one shared process



Cross-adapter batching: straightforward

Isolation: **absent**

SERVERLESS · isolated processes



Cross-adapter batching: not automatic

Isolation: **structural**

Limitations of Existing Systems

Cross-adapter batching, per-adapter isolation, on-demand scaling. Prior systems trade off between them.

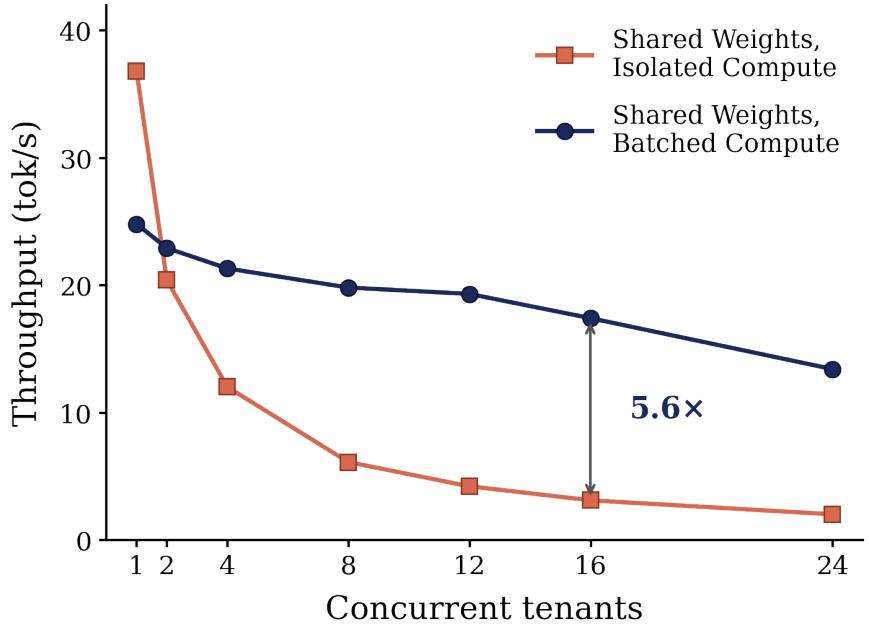
SYSTEM	CROSS-ADAPTER BATCHING	PER-ADAPTER ISOLATION	ON-DEMAND SCALING	MEMORY	BANDWIDTH
S-LoRA, Punica *serverful*	✓	✗	✗	1×	1×
ServerlessLLM *serverless*	✗	✓	✓	N×	N×
ServerlessLoRA *serverless + IPC*	✗	✓	✓	1×	N×
SwarmLoRA *this paper*	✓	✓	✓	**1×**	**1×**

No existing system checks all three boxes with minimum GPU Memory and Bandwidth usage

THE KEY OBSERVATION

The Missing Quadrant, Attempted

Both approaches share the same base-model weights across isolated tenants via CUDA IPC. Only one shares the compute too.



5.6×

throughput gap at 16 concurrent tenants, despite identical shared weights

- Isolated compute still redoes the same xW, tenant by tenant
- That redundancy is what collapses under concurrency
- Batching it keeps throughput far more stable

What if compute and state lived in different places?

THE IDEA

Disaggregate shared computation from per-adapter state.

One persistent aggregator batches xW once per round across every active worker. Each isolated worker independently holds its own adapter, KV cache, and applies xAB.

1

Batch it where no tenant can reach it

a disaggregated architecture

2

Start a worker fast enough to scale on demand

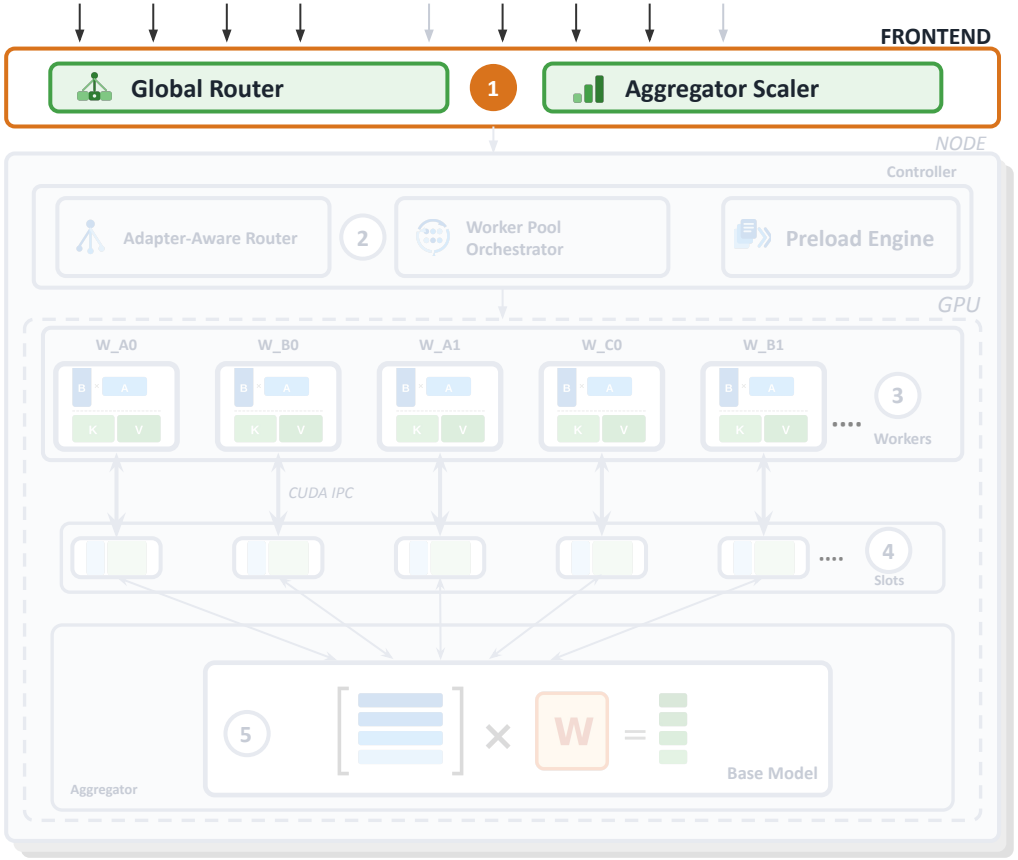
Preload Engine · cold start · hot swap

3

Keep the pool matched to which adapters are hot

adapter-aware scheduling

System Architecture

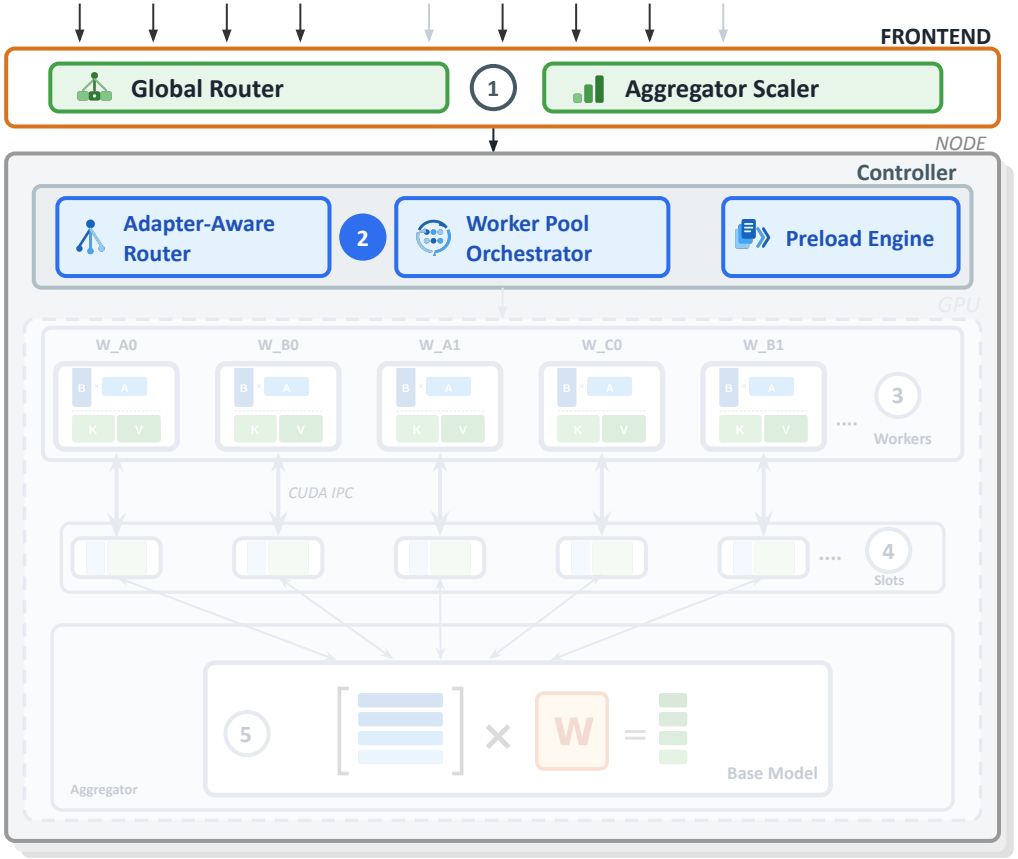


STEP 1

Frontend

- Global Router uses adapter-affinity routing
- Requests go to the node already serving that adapter
- Reuses warm state, avoids a cold start when possible

System Architecture

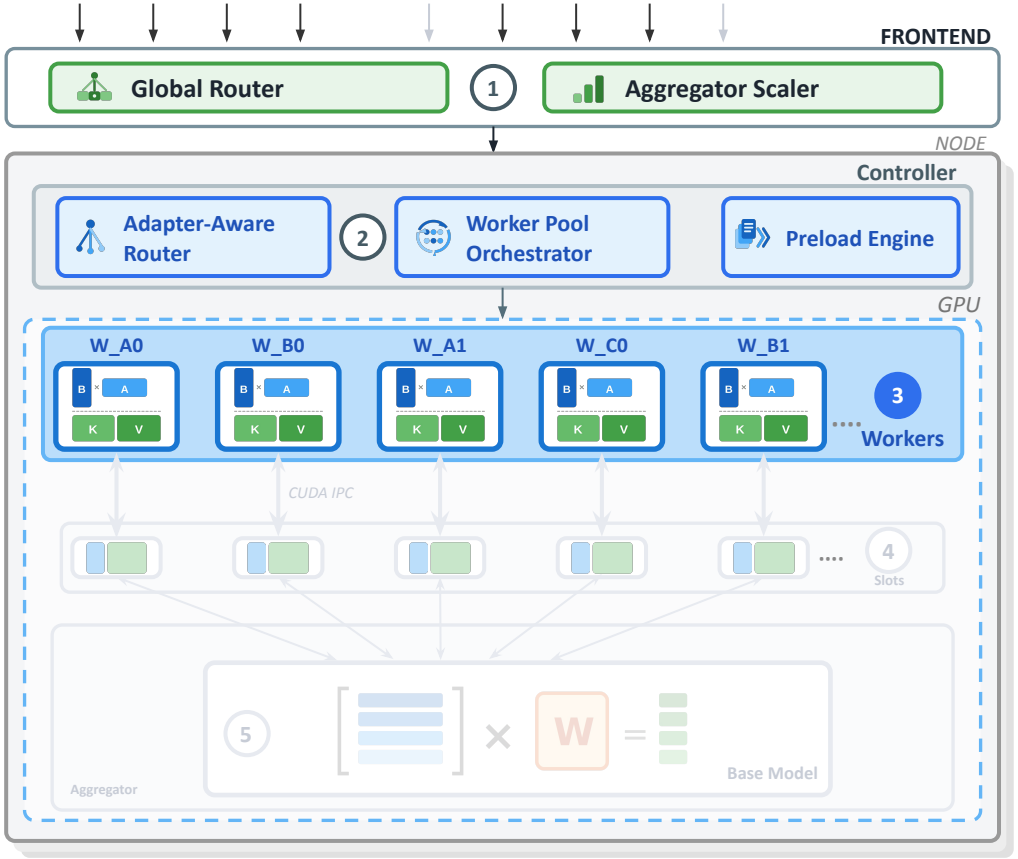


STEP 2

Controller

- Adapter-Aware Router places each request
- Worker Pool Orchestrator manages the pool
- Preload Engine accelerates cold starts

System Architecture

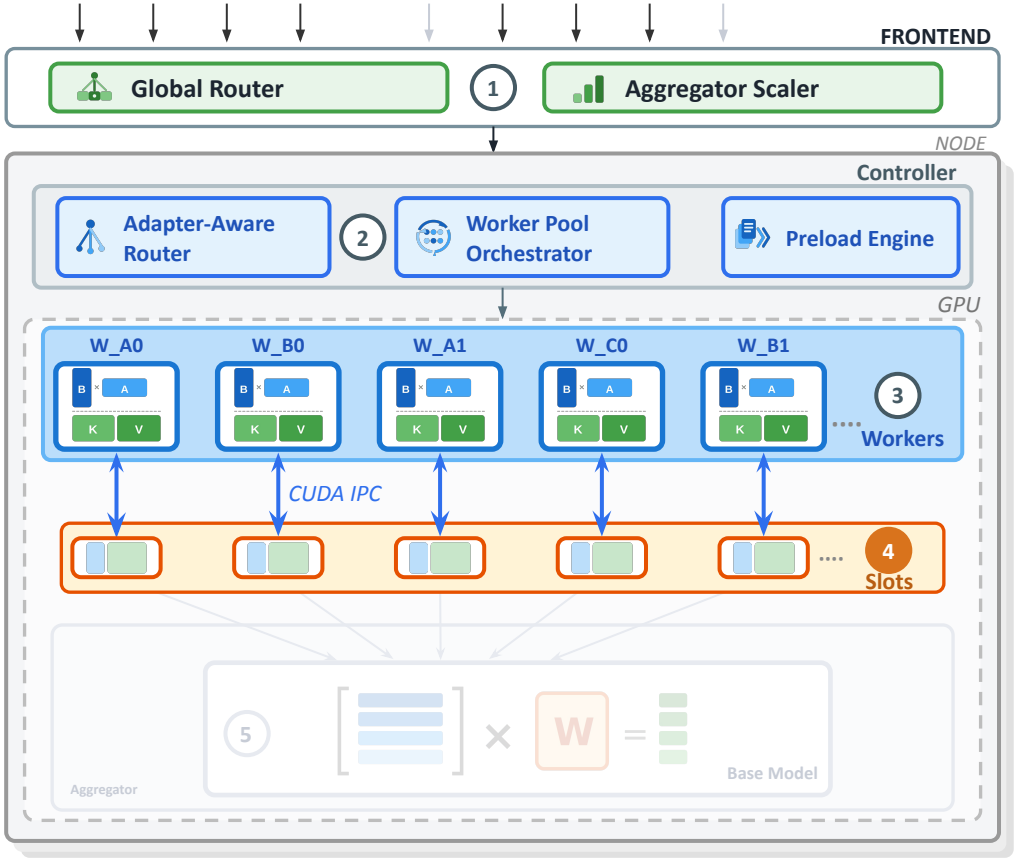


STEP 3

Workers

- One process per adapter, fully isolated
- A hollow model: only LoRA weights + KV cache
- No base weights ever materialize here

System Architecture

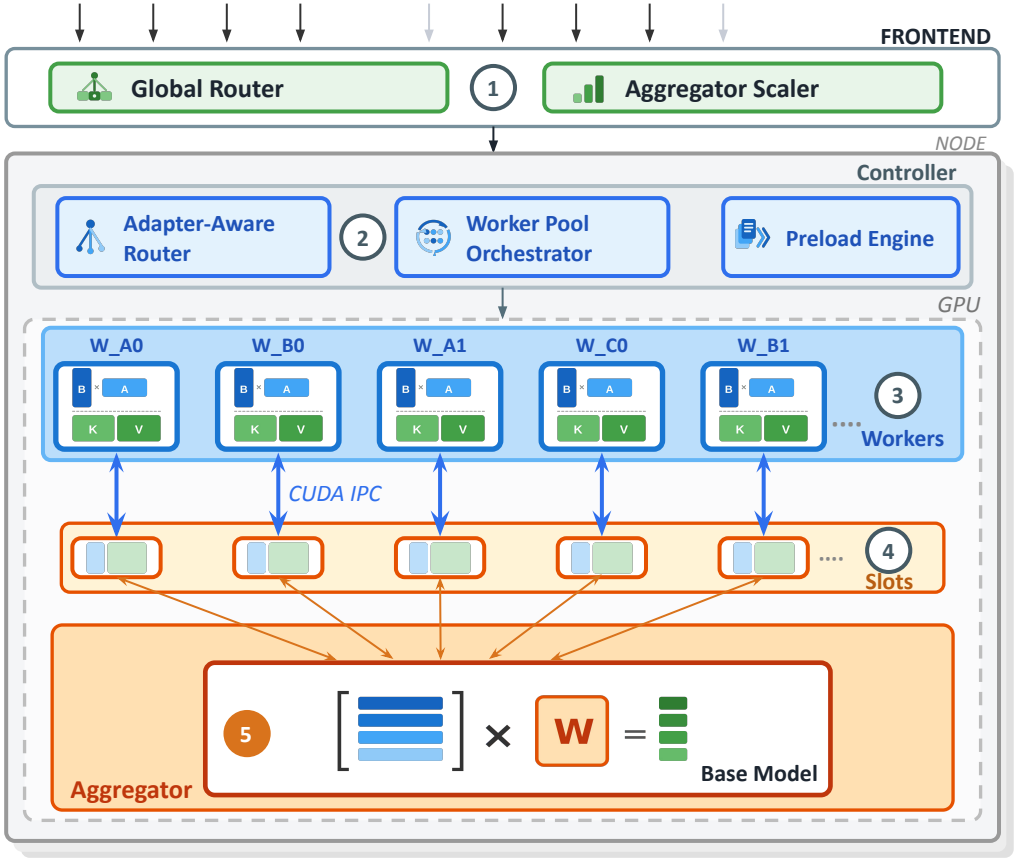


STEP 4

Exclusive Slots

- One dedicated slot pair per worker
- Wired independently via CUDA IPC
- A worker can’t read or write another’s slot

System Architecture

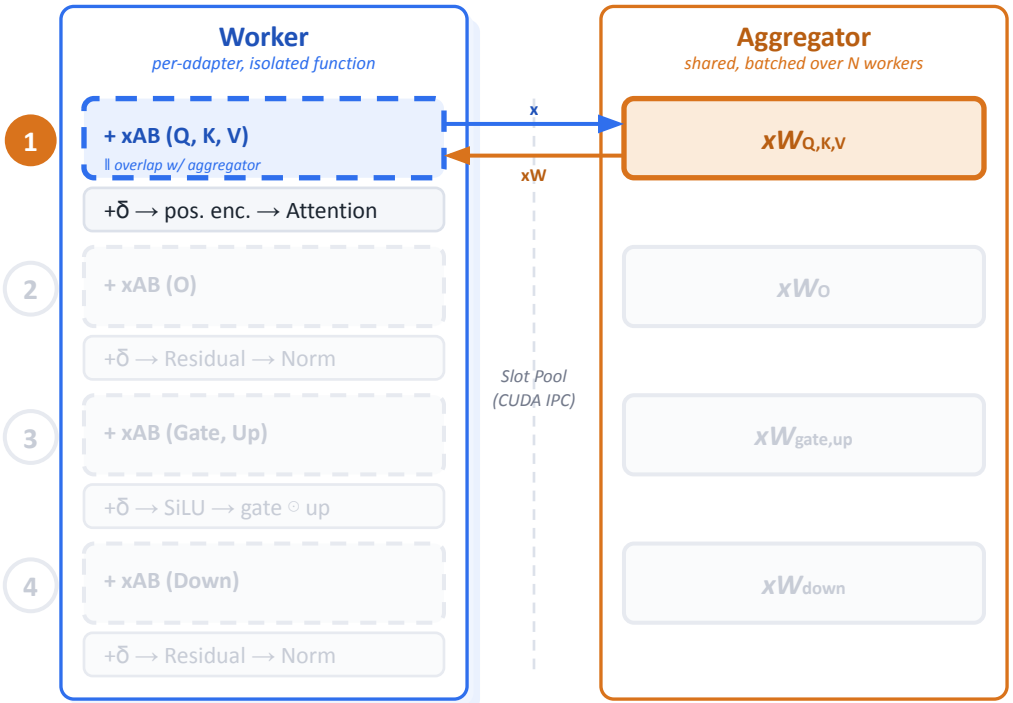


STEP 5

Aggregator

- One per GPU, holds only frozen base weights
- Batches every worker's activations into one GEMM
- 97.7–99.7% of the compute, streamed once not N times

Layer-wise Gather-Scatter

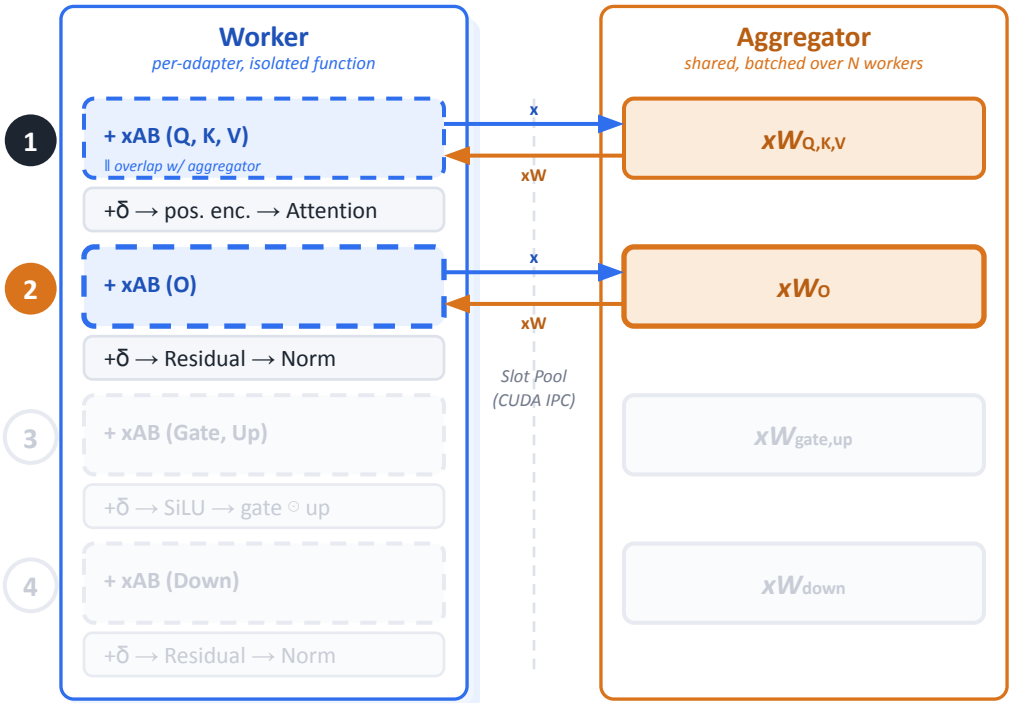


ROUND 1 OF 4

Q, K, V

- Worker submits x to its slot
- Aggregator batches xW across every worker
- Worker computes xAB locally, in parallel

Layer-wise Gather-Scatter

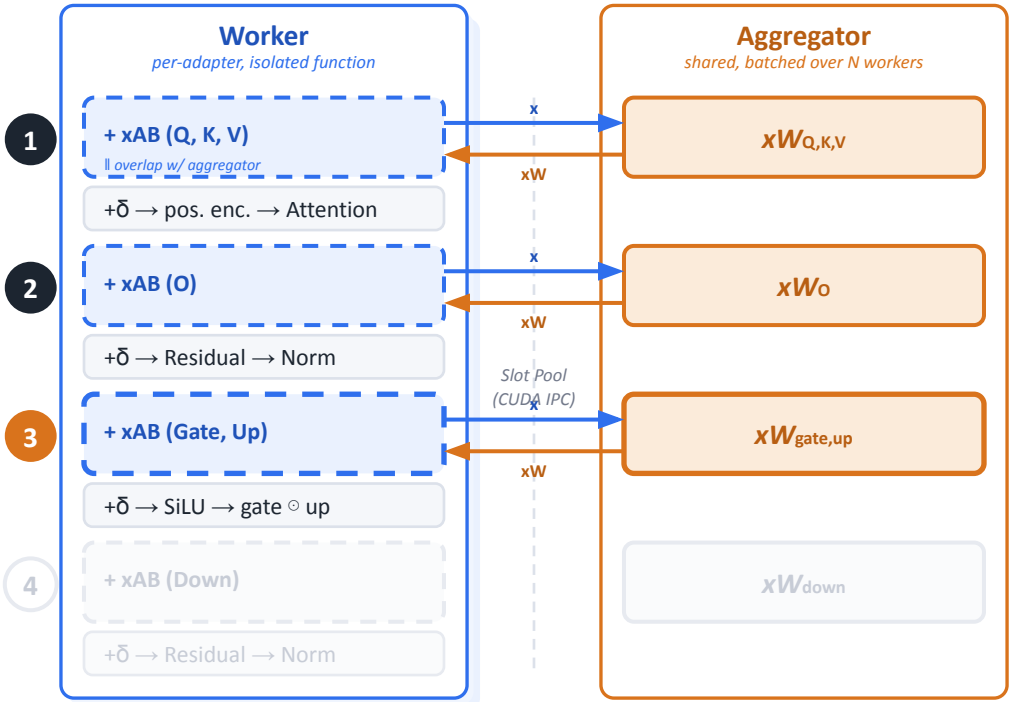


ROUND 2 OF 4

Output

- Same four-phase pattern repeats
- Output projection O this time
- Then residual, then norm

Layer-wise Gather-Scatter

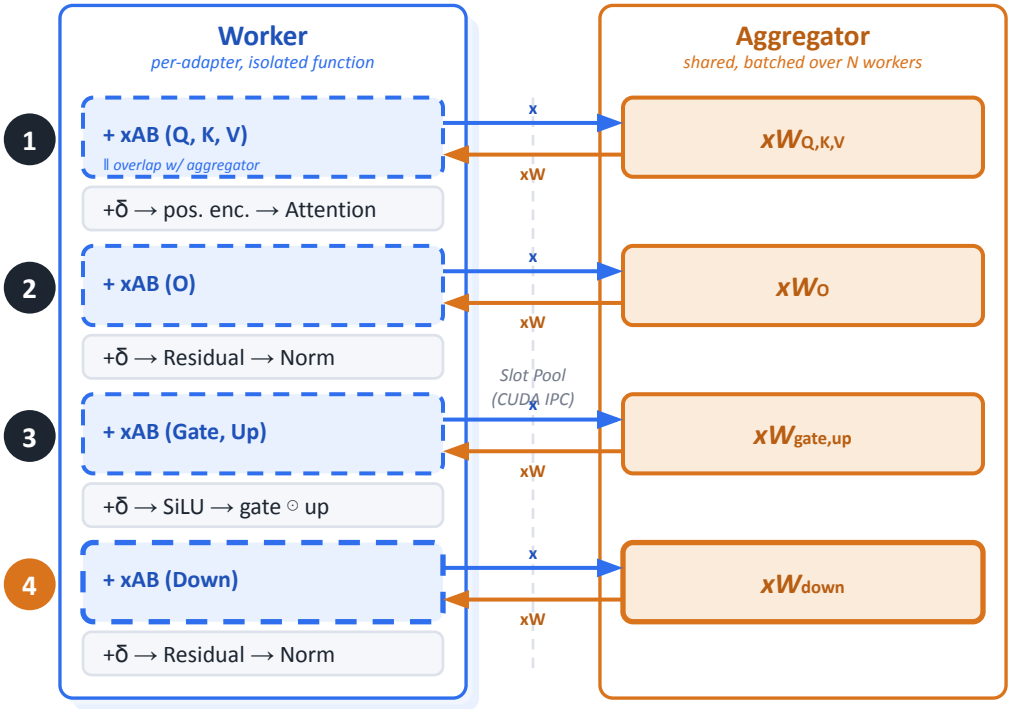


ROUND 3 OF 4

MLP: Gate, Up

- Gate and Up fused into one weight matrix
- Costs one round here, not two
- SiLU activation combines gate ⊙ up

Layer-wise Gather-Scatter



ROUND 4 OF 4

MLP: Down

- Final projection back to the residual stream
- Layer is complete after this round
- 130 rounds total per generated token

2 + 4L = 130 rounds per token. IPC traffic is over 10,000× smaller than the weights streamed per layer.

EVALUATION

Evaluation setup

Two hardware platforms, three comparison systems, and a mix of synthetic and production-trace workloads.

HARDWARE

4× L40S

48 GB GDDR6 each, PCIe 4.0; Intel Xeon Gold CPU

MODELS

Llama-3.1-8B

primary model; model generality on Mistral-7B, Qwen2.5-7B, Llama-2-13B.

WORKLOADS

Poisson + Azure

synthetic Poisson arrivals (1–12 RPS) and 5 replayed Azure Functions traffic categories.

ADAPTERS

10 – 500

rank-16 adapters; up to 500 mixed-rank adapters (r4–r64)

COMPARISON SYSTEMS

ServerlessLLM

Full model replica per tenant; optimized checkpoint loading.

ServerlessLoRA

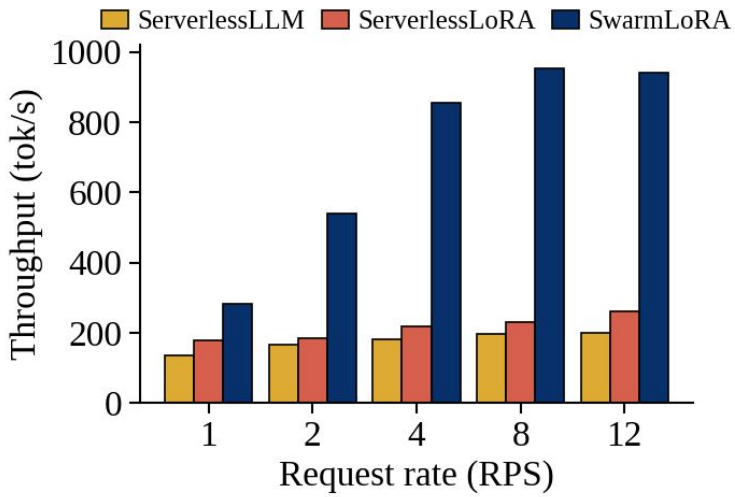
Base weights shared via CUDA IPC; each tenant still runs its own forward pass.

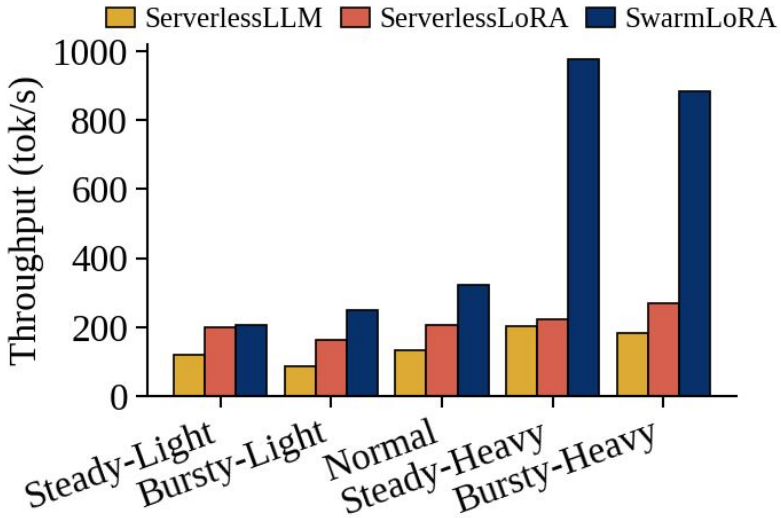
S-LoRA

Serverful, single-process co-batching. Used as the security baseline.

EVALUATION

Throughput

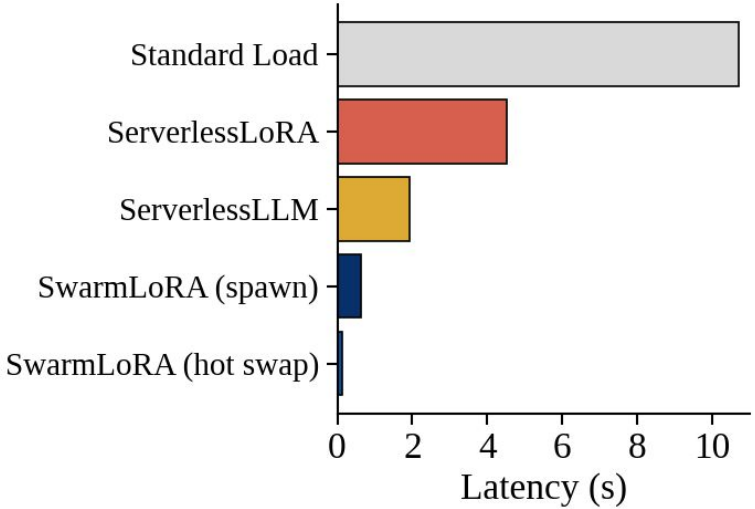


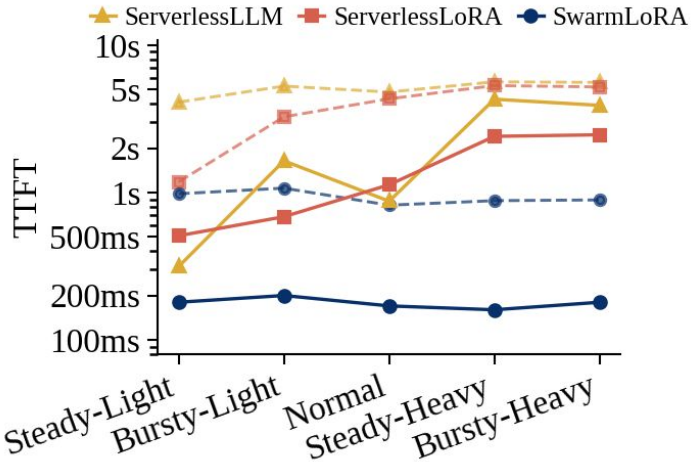


- **954 tok/s** at 8 RPS, vs. 231 (ServerlessLoRA) and 197 (ServerlessLLM)
- **977 tok/s** under steady-heavy load, 4.4–4.9× the two baselines
- **86–100%** request acceptance across all five production trace categories. Baselines fall as low as 18% and 22% under heavy load
- *Both baselines plateau. SwarmLoRA saturates only when GPU compute itself becomes the limit.*

EVALUATION

Cold Start and Adapter Swap Latency

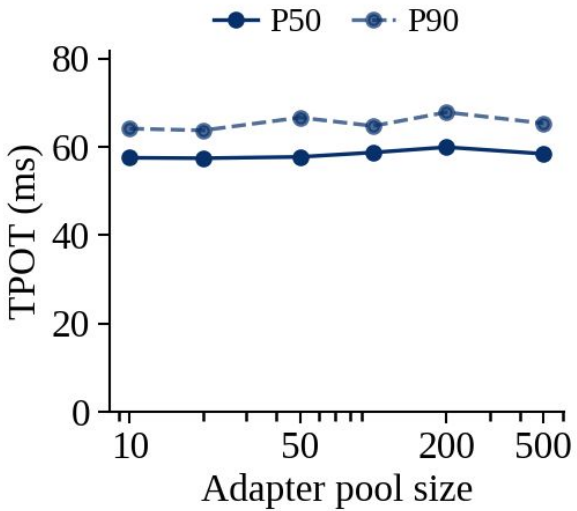


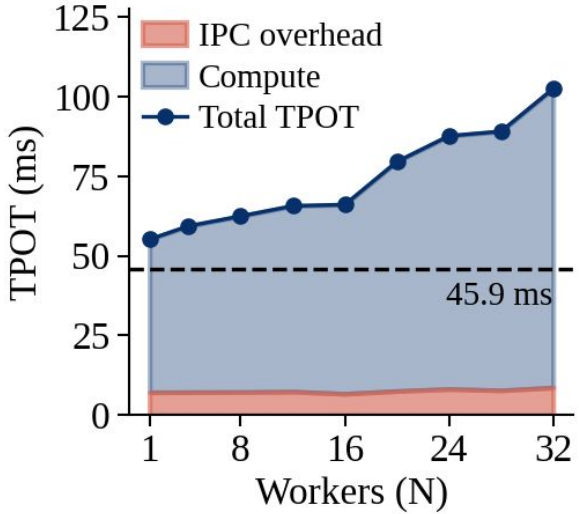


- **618 ms** cold start, vs. 1.93 s (ServerlessLLM) and 4.53 s (ServerlessLoRA)
- **121 ms** hot swap, 5.1× faster than a cold start, so the scheduler swaps before it ever spawns
- **TTFT stays ≤ 200 ms (P50)** in every trace category; both baselines exceed 2.4 s under heavy load

EVALUATION

Scalability

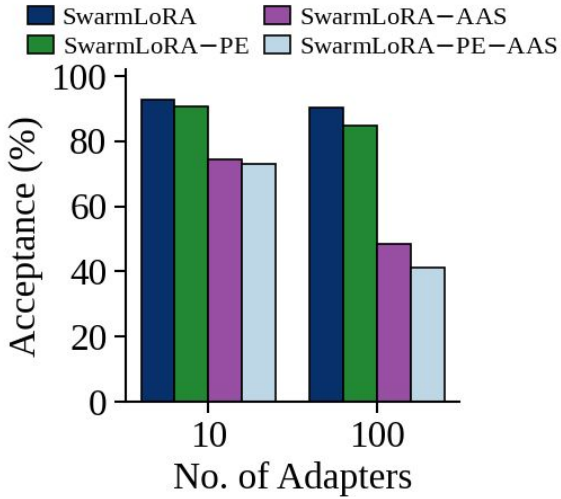


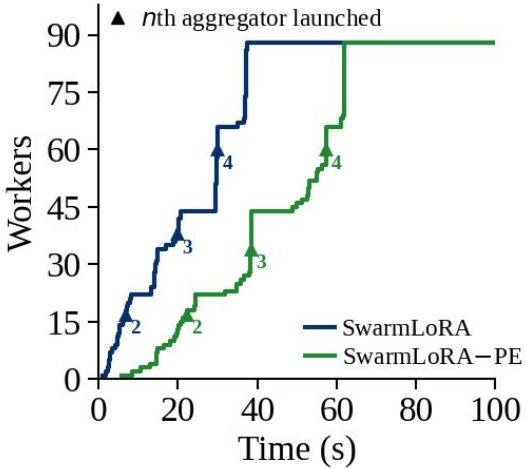


- **57–60 ms TPOT (P50), flat** across adapter pools from 10 to 500 adapters
- **8.5% IPC overhead** share of total per-token cost at N=32 workers, down from 20.2% at N=1
- *Per-token cost is set by GPU compute contention, not by how many adapters or workers share the pool.*

EVALUATION

Ablation





- **−18.3 pts** acceptance at 10 adapters, **−41.9 pts** at 100, when adapter-aware scheduling is removed
- **37 s vs. 62 s** to reach full worker capacity, with vs. without the Preload Engine
- *Scheduling matters more as the pool grows; preloading matters most during ramp-up, before the pool saturates.*

CONCLUSION

Conclusion

The first system that lets a GPU serve many tenants without trading away speed, safety, or scale.

- Removes the choice between batching efficiency and tenant isolation. It’s no longer a trade-off
- Lets one GPU fleet safely serve as many fine-tuned adapters as demand requires, not as many as a static allocation was provisioned for
- Makes multi-tenant serving elastic: capacity follows real demand instead of worst-case provisioning
- The only system where cross-adapter batching, per-adapter isolation, and on-demand scaling all hold at once
- Turns fine-tuned model variants from a cost and security liability into something a provider can safely offer at scale

Thank you. Questions welcome.

Future work: integrating SwarmLoRA into agentic AI workflow systems.

`github.com/SHUs-Lab/SwarmLoRA`

Mausam Basnet
mausam.basnet@unt.edu
Tong Shu (corresponding)
University of North Texas