KempnerForge

Making Foundation-model Training Reliable on Shared Clusters

Developed by Kempner's Research Engineering Team

Scan to learn more about our open-source framework

Naeem Khoshnevis

Lead ML Research Engineer

Kempner Institute
Harvard University

What is KempnerForge?

KempnerForge is an open-source, PyTorch-native training framework for reliable foundation-model training on shared AI clusters.

Fault-tolerant

Preemption-safe checkpointing, auto-resume, NaN detection, and NCCL/GPU health checks.

Cluster-aware

SLURM launch, multi-node training, and topology-aware parallelism for real research hardware.

Scalable

FSDP2, tensor parallelism, expert parallelism, pipeline parallelism, and FP8.

Config-driven

Typed TOML configs, validated before training, with one command from debug to scale.

Composable

Dense, MoE, and VLM models with swappable optimizers, schedulers, losses, and data backends.

Observable & Reproducible

MFU, metrics, profiler, WandB/TensorBoard, distributed checkpoints, and resumable runs.

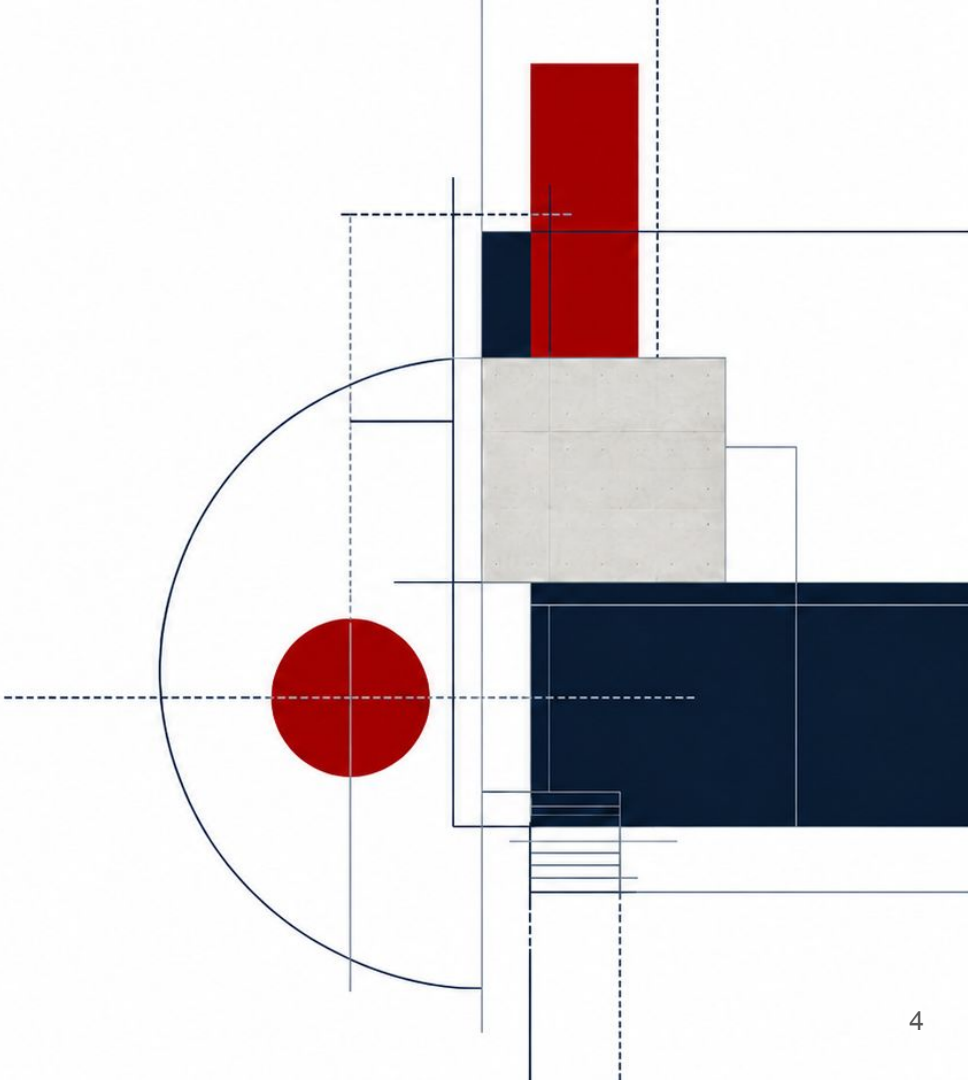
Why another distributed-training framework?

NeMo, Megatron-LM, DeepSpeed, and torchtitan are excellent choices, however, they are built for **different use cases and clusters**.

Assumptions of typical frameworks:

- 8–GPU DGX node is the natural unit of scaling
- Jobs have stable allocations until the next normal checkpoint
- GPU count and parallelism layout stay fixed across resumes
- Environment is strictly controlled by the framework owner
- Heavy custom CUDA / extension stacks are acceptable
- The target user is typically a framework engineer
- Performance tuning is left to expert recipes and deep ecosystem commitment

Design and Architecture



Design Philosophy

Resilience by default

Systems must anticipate and gracefully handle failures.

PyTorch-native

Built to leverage the full power and flexibility of the PyTorch ecosystem.

Build for the cluster

Designed specifically for multi-node, large-scale research hardware.

Reproducible

Deterministic outcomes by construction for scientific rigor.

Extensible

Modular design allows for easy integration of custom components.

Observable

Deep insights into system performance and training metrics.

Architecture

Configuration
validated before any GPU run

Dense Transformer
RoPE . GQA . SwiGLU

Mixture-of-Experts
2 routers . grouped GEMM

Vision-Langauge
4 fusion architecture

Parallelism engine - FSDP2 . Tensor . Expert . Pipeline . FP8

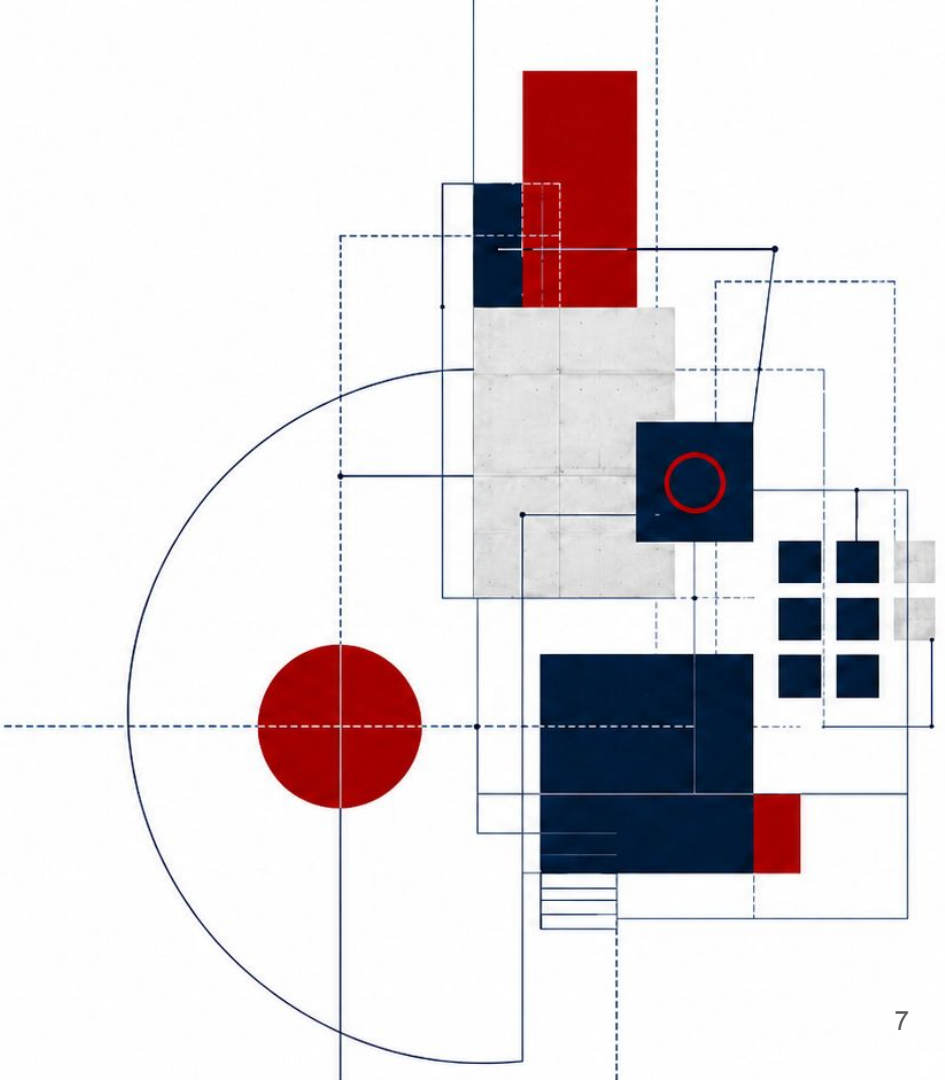
Training: optimizers . schedulers . loss function

Checkpoint & Resilience
async DCP . auto-resume . Nan/health checks

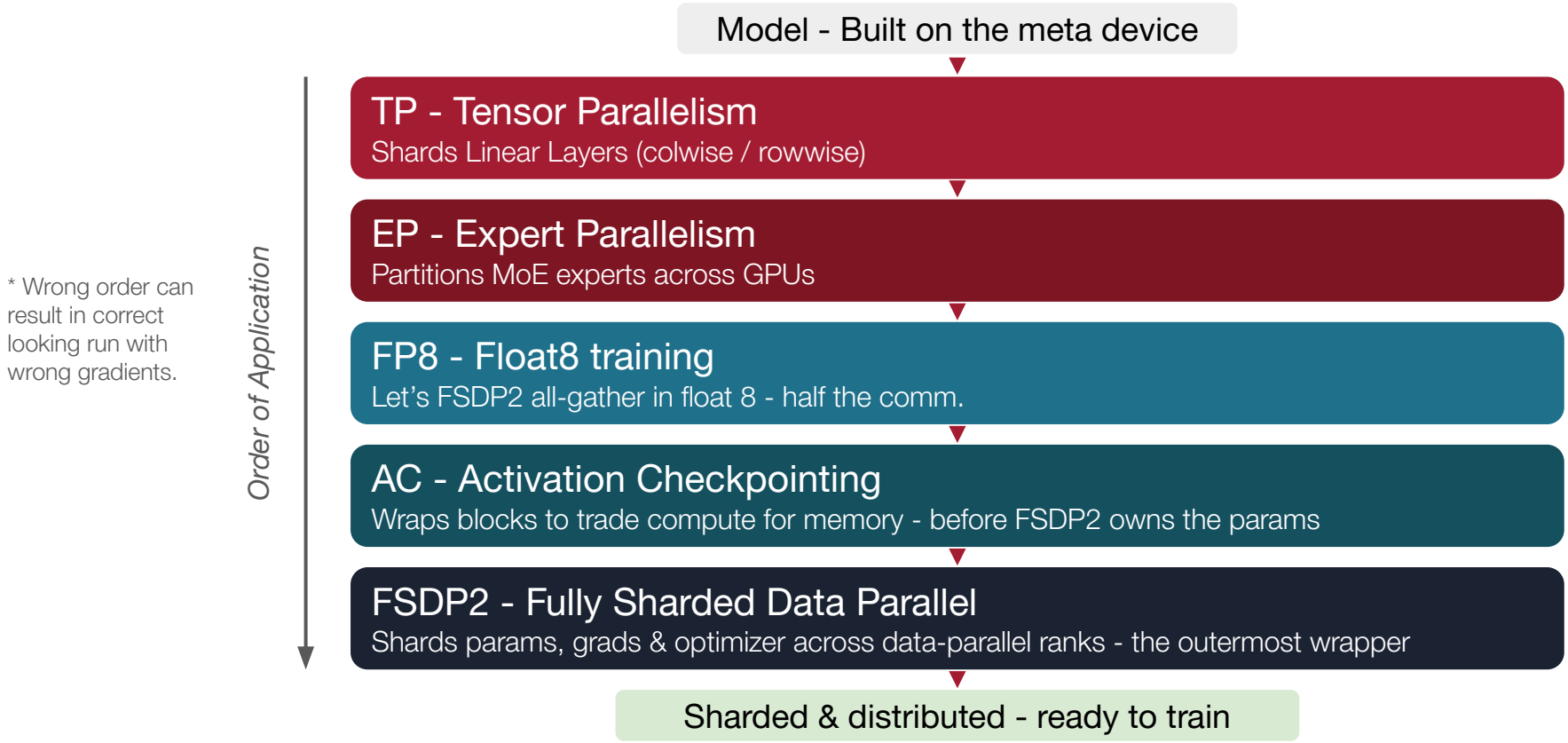
Data Pipeline
mmap . HuggingFace . Mixing . Annealing

PyTorch foundation, DTensor + Distributed Checkpoint

Scaling and Resilience



Scaling the model

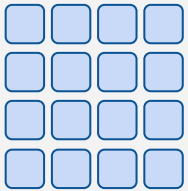


* Wrong order can result in correct looking run with wrong gradients.

Checkpointing

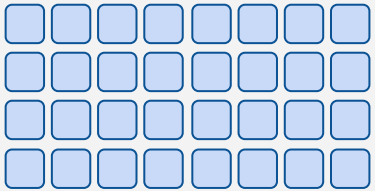
Save on N . Resume on M

16 GPUs



Distributed Checkpoint (DCP)
N-to-M resharding
async . sharded . durable

32 GPUs



Survive Preemption

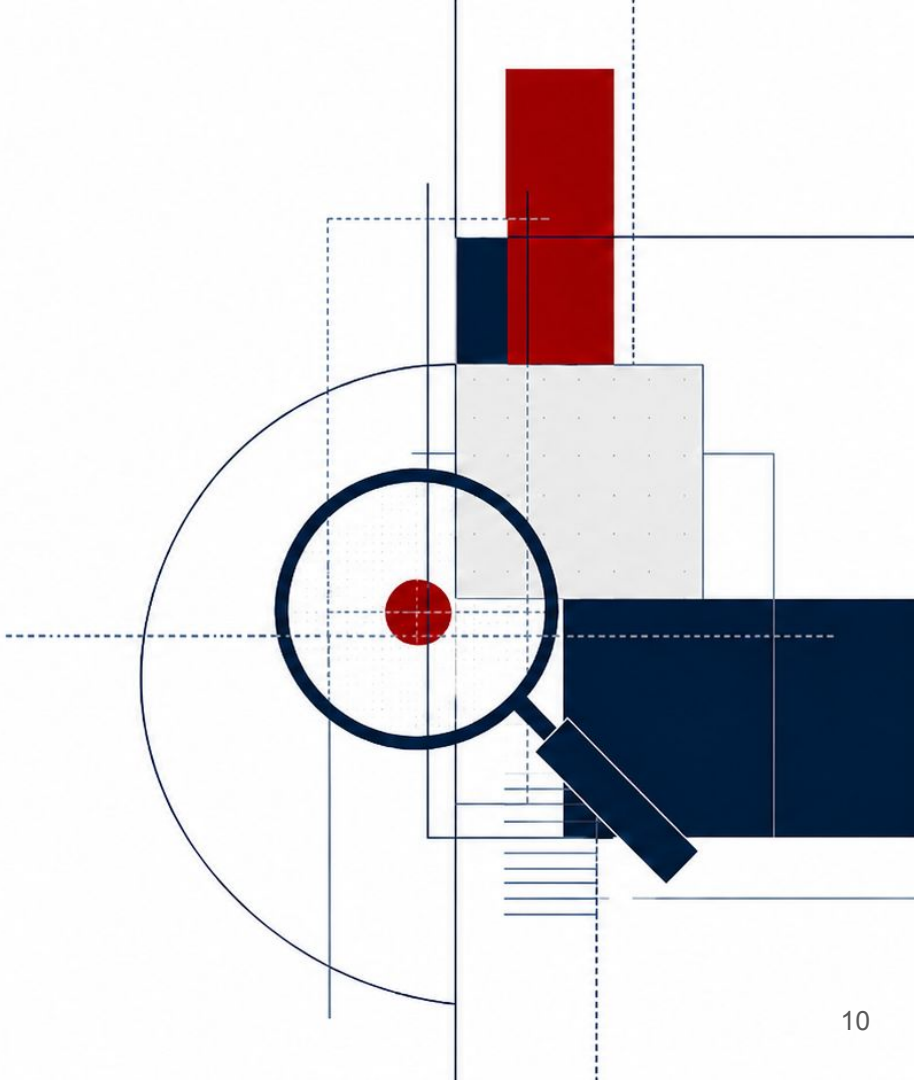
SIGTERM caught
SLURM signals preemption

Checkpoint
Written in the grace window

Requeue
SLURM relaunches the job

Auto-resume
from the latest checkpoint

Research



Models

Language Models

Dense Transformer
RoPE · GQA · QK-Norm · SwiGLU · RMSNorm

Mixture-of-Experts
Softmax & DeepSeek-V3 routers · Shared experts

Scale Range: 20M to 70B+
Consistent configuration, scaled across more GPUs

Multimodal

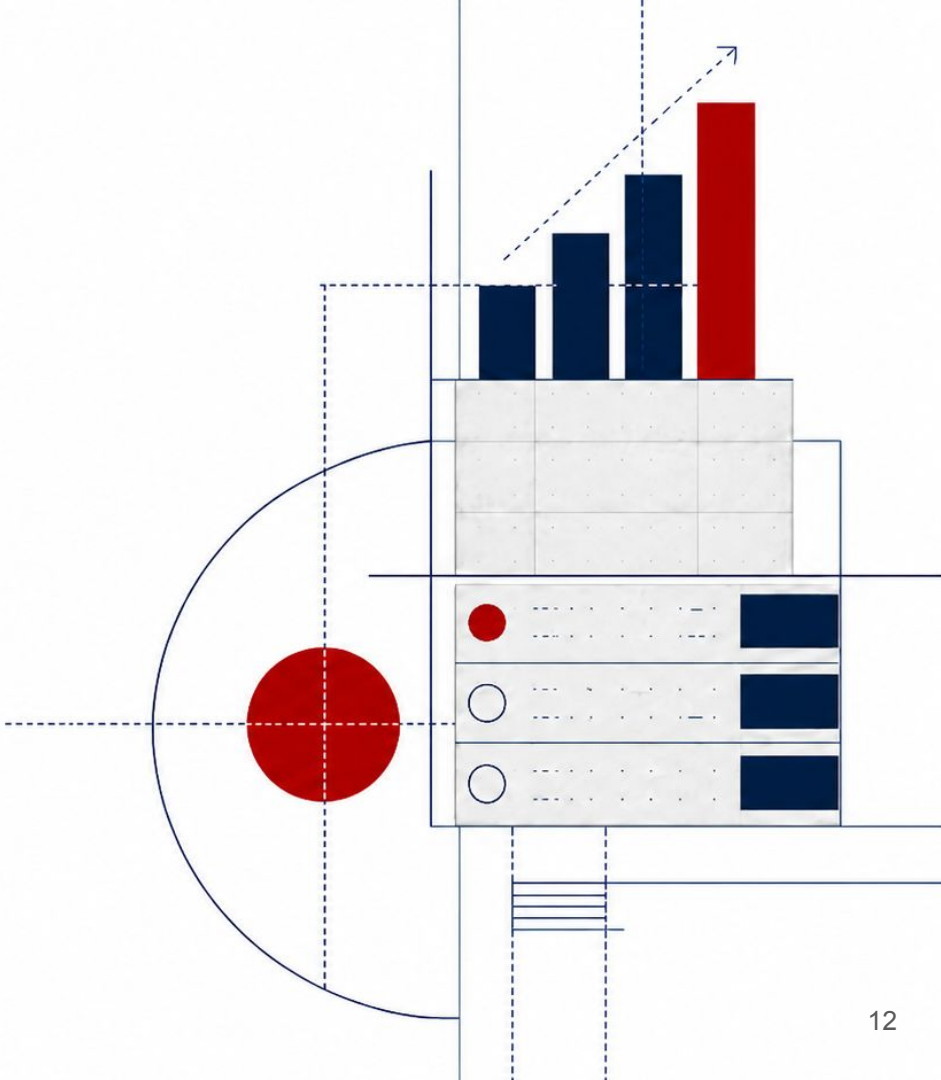
Joint-Decoder
Image tokens prepended to text streams

Cross-Attention
Llama-3-V style: image features as keys/values

Mixture-of-Transformers
Per-modality weights, warm-started from text

MoMa
Modality-aware expert Mixture-of-Experts

Benchmark Runs



Benchmark Campaigns

42
documented runs

5
benchmark campaigns

57.8 %
peak MFU (7B, 1 GPU)

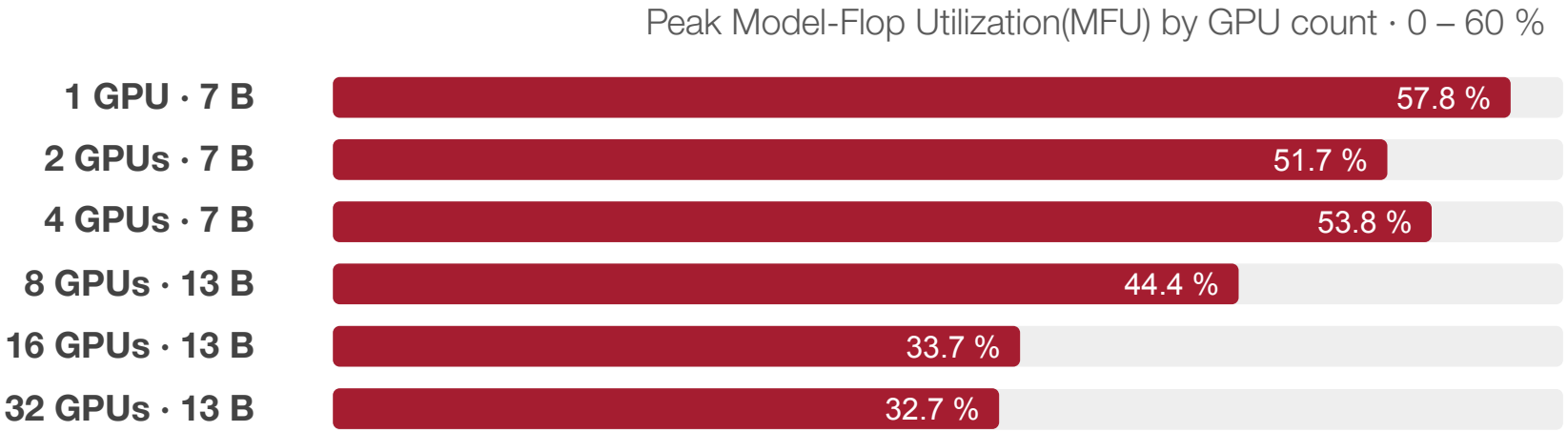
70.6B
largest model

Up to 8 nodes × 4 H200 (141 GB) · NVLink + InfiniBand NDR 400 Gbps · 20M - 70.6B params

FineWeb-Edu 499 B tokens

Campaign	Scope	Runs
MFU scaling	7B · 13B · 70B · 1-32 GPUs	**14**
Config validation	9 configs · 20 M - 70.6 B · 4-32 GPUs	**9**
MoE · expert-parallel	4B MoE · 1.8 B active · 32 GPUs	**10**
MoE · expert-packing	E = 8/16/64 · 8 GPUs	**8**
Profiling	13B · 24GPUs · torch.profiler	**1**

Performance



All runs use bf16 precision and the FineWeb-Edu dataset.

57.8%
Peak MFU — 7B on a single H200

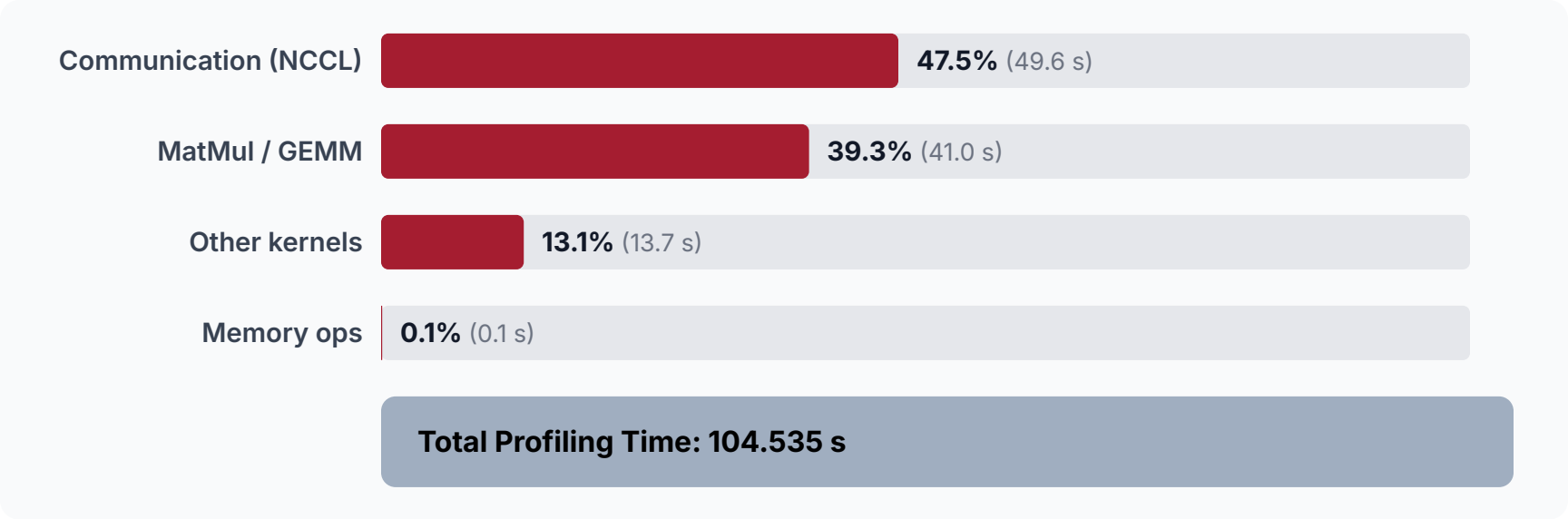
71%
Tensor Core Activity ·

~93%
Inside node throughput scaling

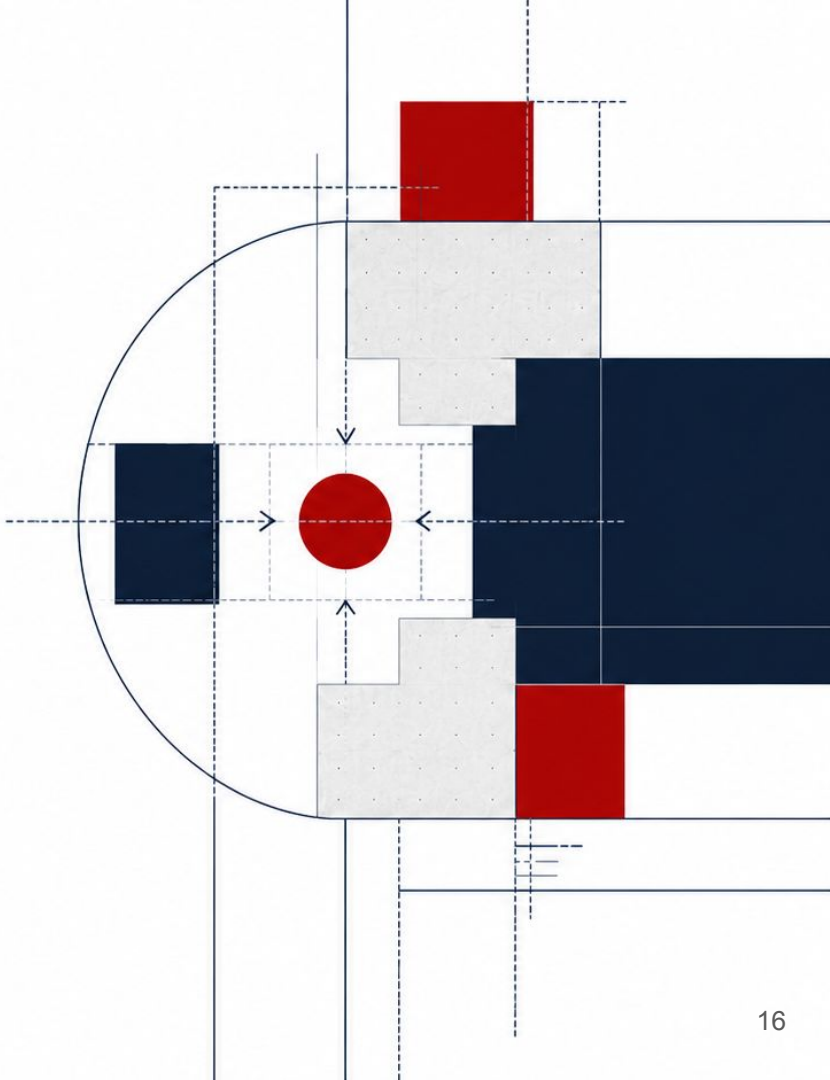
104 K token/s
13 B Throughput on 32 GPUs

Profiling

13 B · 24 GPUs · Post warm up



Contribution



Contributions & Extensibility

Contributions are welcome

1. Open an issue

Report a bug or propose a feature to start the conversation.

2. Implement an issue

Pick up an open task and send a pull request to the main branch.

Built to be easily extended

- **Pure PyTorch**
  No exotic dependencies, natively integrated.
- **Registry Integration**
  Add components seamlessly via registry system.
- **Agentic Ready Skills**
  Pre-built for complex agent behaviors.
- **1500+ Tests & Docs**
  Reliable codebase with comprehensive guidance.

Thank you

Questions?

We'd love to hear from you. Check out the repository or reach out directly.

Repository

github.com/KempnerInstitute/KempnerForge

Scan to learn more about our open-source framework