FreeScale: Distributed Training for Sequence Recommendation Models with Minimal Scaling Cost

Chenhao Feng*, Haoli Zhang*, Shakhzod Ali-Zade, Yanli Zhao, Liang Luo, Jennifer Cao, Lisen Deng, Siqiao Chen, Chenyu Zhao, Tristan Rice, Daniel Johnson, Min Si, Tiantu Xu, Yi Zhang, Siqi Yan, Chuanhao Zhuge, Min Ni, Bi Xue, Qunshu Zhang, Shen Li

Meta Inc.

MLSys 2026, Bellevue, WA

** Equal contribution*

Deep Learning Recommendation Models Are Critical — and Costly to Scale

Context: Industrial DLRMs at Scale

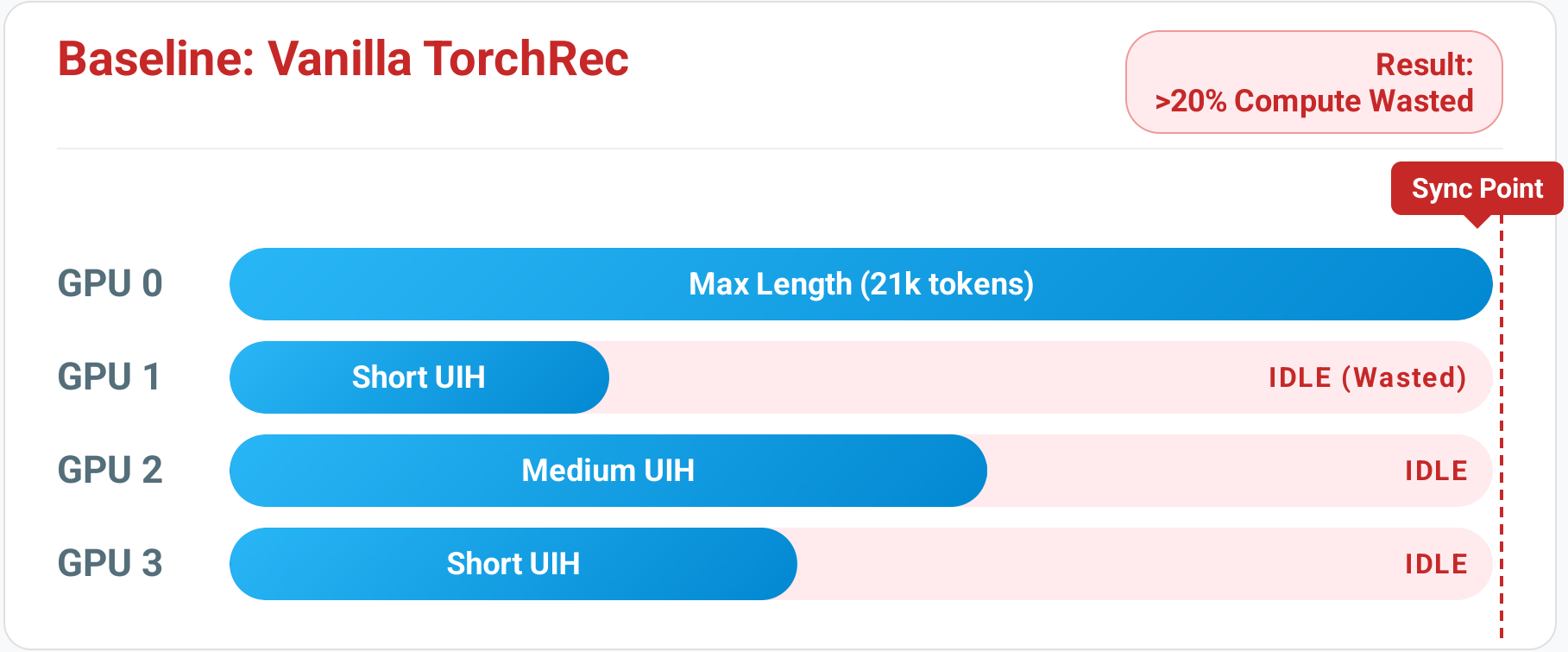
- DLRMs power eCommerce, social media, and digital advertising at Meta scale
- Modern DLRMs use **sequential interaction histories** (User Interaction History, UIH) to extract user preferences
- Architecture: massive **sparse embedding tables** (sharded) + **dense DNN layers** (replicated/FSDP)
- Training involves AllToAll communications for embedding lookups across hundreds of GPUs

The Scaling Challenge

Component	Characteristic
Embedding tables	Partitioned across GPUs via AllToAll
Dense DNN layers	Replicated/FSDP; gradients synced via AllReduce
UIH sequences	Highly variable length (up to thousands of tokens)
Benchmark Cluster size	256 of H100 GPUs

Problem 1: Stragglers — Variable UIH Lengths Waste GPU Compute

User Interaction History (UIH) lengths vary wildly (from a few tokens to 21,000+). Standard pipelines pad sequences to the maximum length, causing GPUs with shorter sequences to sit idle while waiting for the "straggler" GPU to finish.

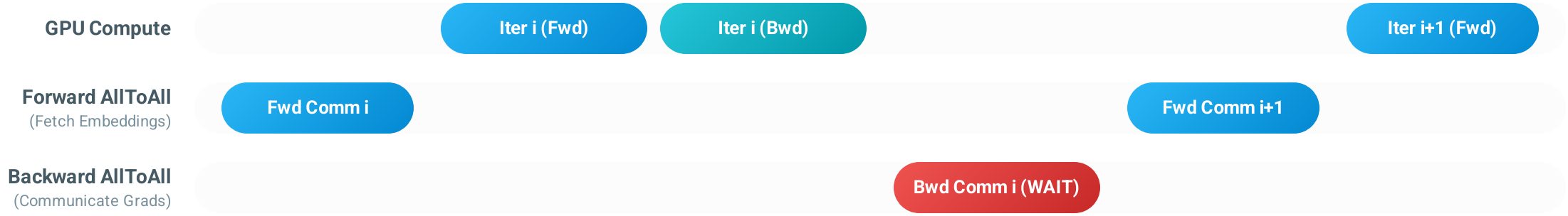


Problem 2: Blocking Communications — Two-Stage AllToAll

Each iteration requires two communications: **Forward AllToAll** (fetching embeddings) and **Backward AllToAll** (communicating gradients). The baseline blocks the next iteration until both finish. Naïve prefetching fetches *stale* embeddings.

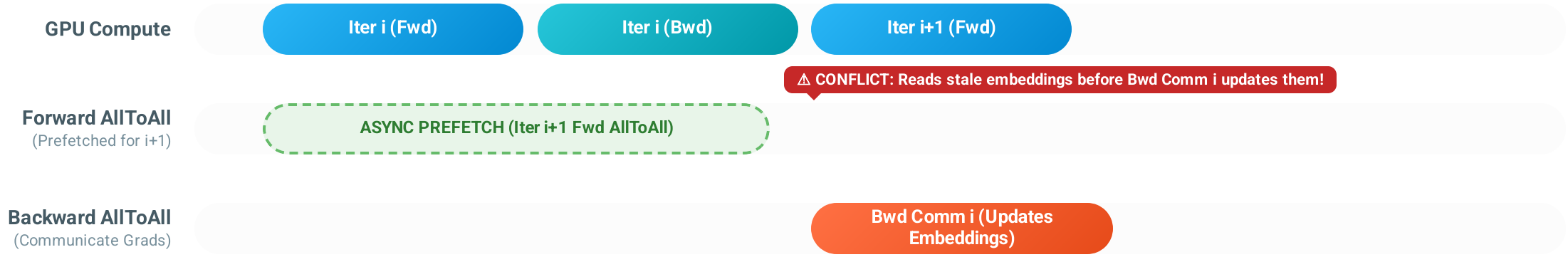
Baseline: Strict Blocking

Result: Severe resource under-utilization



The Pitfall: Naïve Prefetching

Result: Stale embeddings degrade model quality



FreeScale: Three Coordinated Techniques Eliminate Computational Bubbles

System Overview

FreeScale is a holistic solution built on **PyTorch + TorchRec + Triton** that addresses both root causes:

Technique	Problem Addressed	Key Idea
Sequence Load Balancing	Stragglers	Redistribute samples by UIH length before each iteration
Prioritized Embedding Updates	Blocking communications	Prefetch non-collision rows; wait only on collision rows
SM-Free Communication	GPU resource contention	Route overlapped comms through CPU-RDMA, not NCCL

> **Implementation:** ~8,600 lines of code (2.5% of TorchRec codebase)

Load Balancing: Redistribute Samples to Equalize GPU Workloads

Three-Stage Protocol (Overlapped with Computation)

1 Gather Lengths
Collect UIH lengths and candidate counts from all ranks

→

2 Gather Composite Candidate Lengths
AllGather composite candidate lengths across the cluster

→

3 Partition & Shuffle
Run partitioning algorithm and shuffle samples to balance GPUs

Two Partitioning Strategies

Fixed Batch Size (FBS)

Zig-zag assignment of globally sorted samples

Sorted Pool: [Long, Long, Med, Med, Short, Short]

GPU 0	Long, Short	(2 Samples)
GPU 1	Long, Short	(2 Samples)
GPU 2	Med, Med	(2 Samples)

Every GPU gets exactly 2 samples.

Trade-off: Versatile, but may cause intra-kernel imbalance

Variable Batch Size (VBS)

Segment by weight L^{α}; auto-tune batch size

Goal: Equalize Total Compute Weight

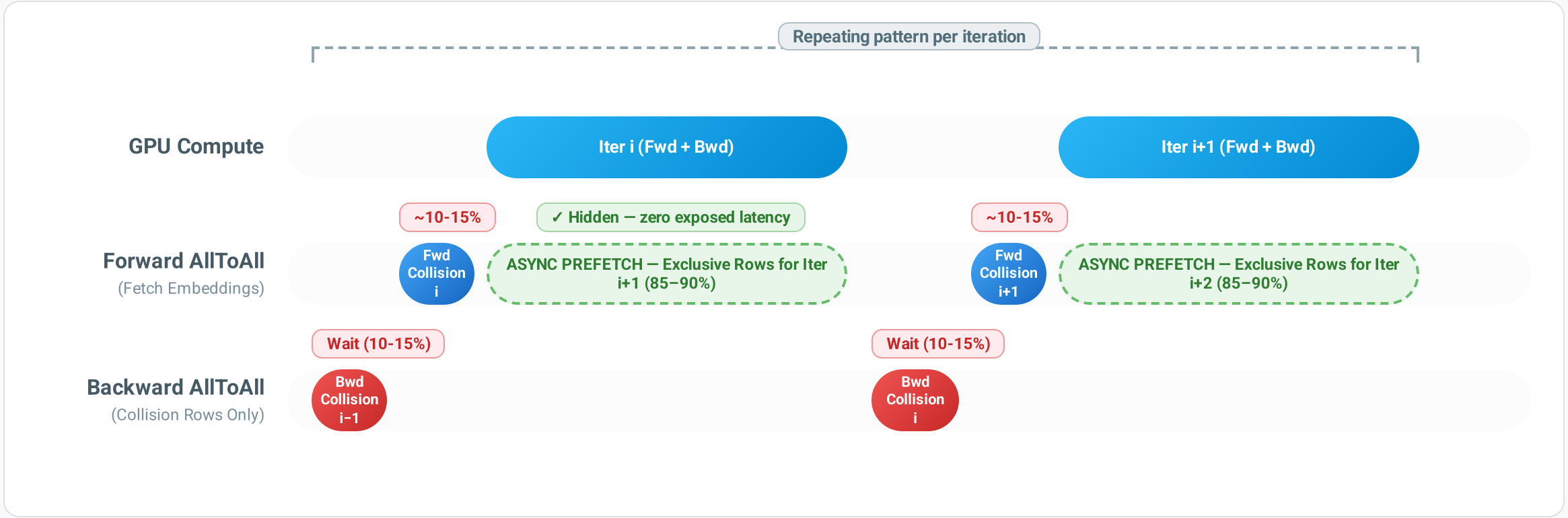
GPU 0	Very Long	(1 Sample)
GPU 1	Med, Med	(2 Samples)
GPU 2	S, S, S, S	(4 Samples)

GPUs get different # of samples, but equal total width.

Trade-off: Minimal sparsity, requires gradient scaling calibration

Prioritized Embedding Updates: Prefetch Non-Collision Rows, Block Only on Collisions

Key Insight: Decouple **exclusive rows (85–90%)** — prefetched asynchronously with zero wait — from **collision rows (10–15%)** — a small blocking wait per iteration.



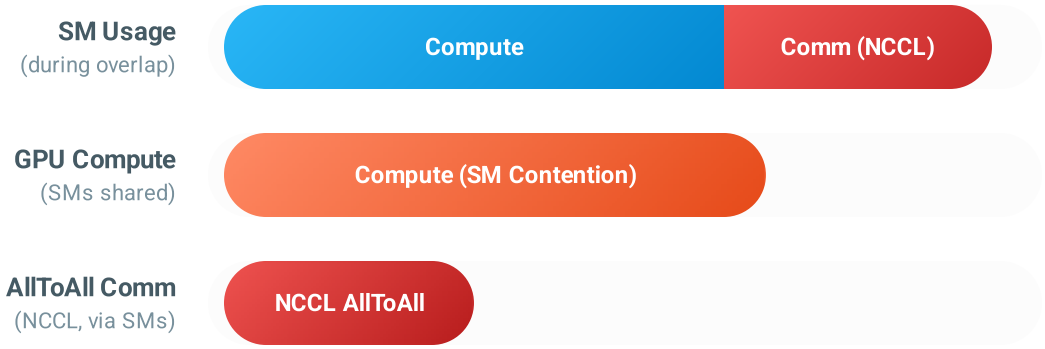
Result: Exposed blocking communication reduced to **collision rows only (~10–15%)** — a **90.3% reduction** in exposed communication latency

SM-Free Communication: Eliminate GPU Resource Contention

Standard NCCL collectives use GPU Streaming Multiprocessors (SMs) for communication, creating **SM contention** that degrades overlapped compute. FreeScale uses **CPU-driven RDMA** to bypass SMs entirely — dedicating 100% of SM capacity to compute.

Baseline: NCCL Communication

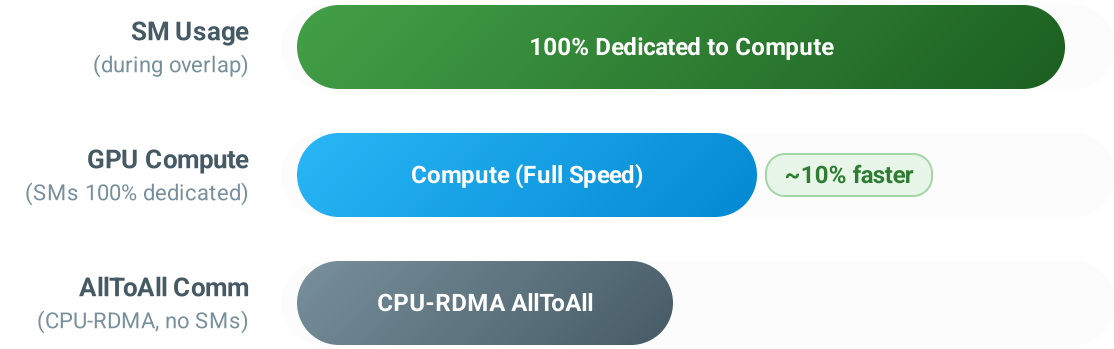
SM Contention



NCCL launches each channel as a CUDA block on its own SM → SM contention degrades overlapped compute

FreeScale: SM-Free (CPU-RDMA)

+10% MFU



Comm bypasses SMs entirely via CPU-RDMA → SMs 100% dedicated to compute → +10% MFU

Evaluation: Up to 90.3% Reduction in Computational Bubbles

HARDWARE SETUP

Up to **256 NVIDIA H100** GPUs

MODEL QUALITY

Maintains **exact numerical parity** (same Normalized Entropy)

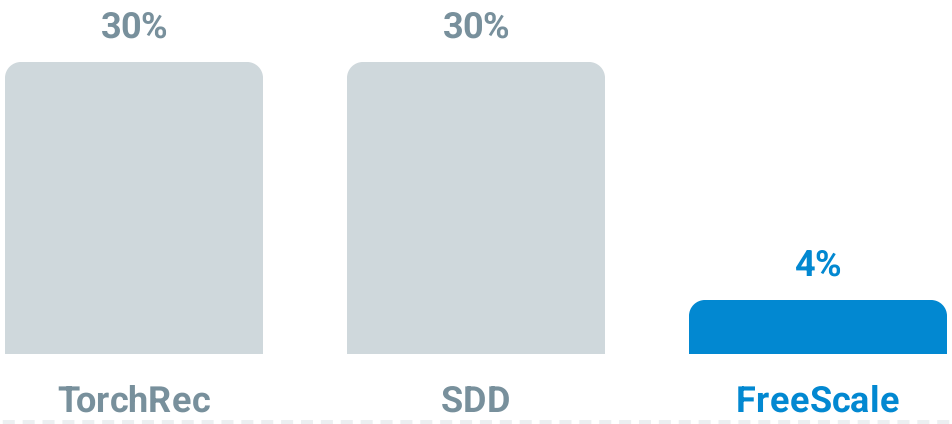
THROUGHPUT

Delivers up to **~33% QPS improvement** vs TorchRec

Straggler Percentage

Measured on 21k UIH dataset

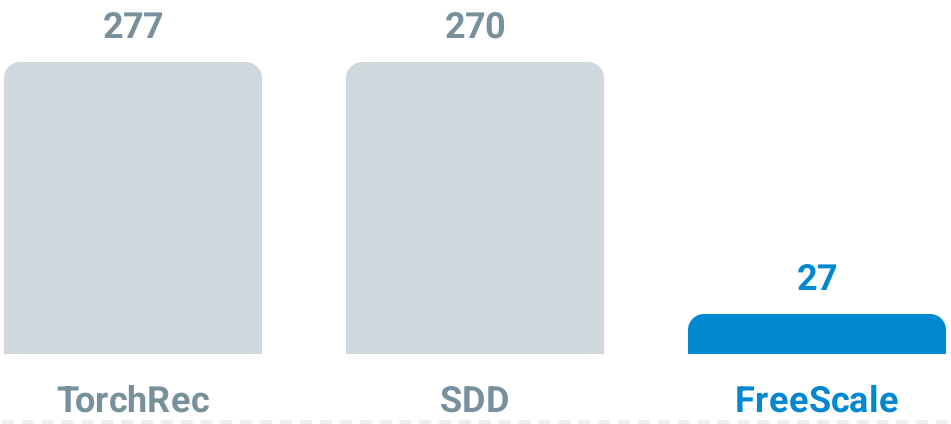
↓ 9× Reduction



Exposed Comm. Latency

Measured in milliseconds (ms)

↓ 90.3% Reduction



Triton Kernels Deliver 20–600× Speedup for Load Balancing Operations

Challenge: Jagged tensor operations (variable-length UIH) are inefficient in standard PyTorch

Specialized Triton Kernels

- **Indexed permute kernel**: apply partitioning transforms to AllGathered IDs
- **Ranged dispatch / combine kernels**: prepare tensors for ID and embedding AllToAll
- **Keyed transpose kernel**: convert feature-major ↔ batch-major representations

Performance vs. PyTorch Eager

World Size	PyTorch Dispatch	Triton Dispatch	PyTorch Combine	Triton Combine	Speedup (Max)
32	3.0 ms	0.1 ms	3.8 ms	0.1 ms	**38×**
64	6.2 ms	0.1 ms	7.6 ms	0.1 ms	**76×**
128	12.4 ms	0.2 ms	15.3 ms	0.1 ms	**153×**
256	23.8 ms	0.6 ms	29.5 ms	0.1 ms	**295×**
512	48.9 ms	2.0 ms	59.0 ms	0.1 ms	**590×**

Summary: FreeScale Achieves Holistic Efficiency for Industrial DLRM Training

Three Techniques, One Unified System

Technique	Mechanism	Result
Sequence Load Balancing	Redistribute samples by UIH complexity	**9× straggler reduction**
Prioritized Embedding Updates	Prefetch exclusive rows; block only on collisions	**9× exposed comm. reduction**
SM-Free Communication	CPU-RDMA ring for overlapped collectives	**10% additional speedup**

Combined Impact

- **90.3% reduction** in computational bubbles on 256 H100 GPUs
- **Full numerical parity** with baseline — no model quality compromise