Demo: Weaver – A System for AI-RAN Compute Sharing with Foundation Model Training

Tianxin Wang[1], Xin Zhe Khooi[2], Jiaxun Yang[1], Dheeraj Mahendiran[1], Leyang Xue[1], Mun Choon Chan[2], Mahesh K. Marina[1]
[1]The University of Edinburgh [2]National University of Singapore

Abstract

We present Weaver, a system that enables opportunistic foundation model (FM) training alongside realtime-sensitive RAN workloads on GPU-accelerated AI-RAN compute infrastructure. Weaver safely harvests spare compute and enables GPU compute sharing between RAN and FM training workloads at a fine-granularity. Weaver achieves this through a RAN-first design, by introducing (1) a RAN-centric spare compute controller integrated with the MAC scheduler to prioritize RAN performance while stabilizing GPU spare compute, and (2) an elastic, RAN-slot-bounded training policy that decomposes FM training into fine-grained work units and executes only the work that fits within each slot's available spare compute. In this demo, we showcase how Weaver allows RAN and training workloads to safely co-exist with each other within a single cell site without violating RAN deadlines, while allowing training to effectively harvest the available spare compute at a per-slot granularity, substantially improving the training throughput.

CCS Concepts

• **Networks** → **Mobile networks**.

Keywords

AI-RAN, GPU sharing, FM training, compute harvesting

ACM Reference Format:
Tianxin Wang, Xin Zhe Khooi, Jiaxun Yang, Dheeraj Mahendiran, Leyang Xue, Mun Choon Chan, and Mahesh K. Marina. 2026. Demo: Weaver – A System for AI-RAN Compute Sharing with Foundation Model Training. In *The 32nd Annual International Conference on Mobile Computing and Networking (MobiCom '26), October 26-30, 2026, Austin, TX, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3795866.3848468



This work is licensed under a Creative Commons Attribution 4.0 International License.
MobiCom '26, Austin, TX, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2505-0/2026/10
https://doi.org/10.1145/3795866.3848468

1 Introduction

As the industry moves towards 6G, GPU-accelerated RAN infrastructure is emerging as the de-facto hardware substrate [1], as evident with the Nokia-NVIDIA partnership [6]. Beyond AI-*for*-RAN (improving RAN with AI) and AI-*on*-RAN (hosting edge AI services), this shift also enables AI-*and*-RAN: sharing RAN compute with non-RAN workloads.

AI-*and*-RAN is attractive because RAN compute is often provisioned for peak load, while spatiotemporally bursty RAN traffic [8] leaves substantial GPU capacity idle. Our characterization study likewise finds substantial unused GPU compute even under realistic high loads. Therefore, the key challenge is not spare-compute availability, but exposing and using it efficiently while preserving RAN safety.

We focus on harvesting spare GPU compute for foundation model (FM) training, where FM training serves as a stress-test workload for AI-and-RAN due to its high compute demand, large persistent state, and sensitivity to resource variability. If FM training can run opportunistically alongside the primary RAN workload, then less demanding workloads, e.g., inference, can be supported as well.

We design Weaver [11], a system that enables fine-grained GPU sharing between realtime-sensitive RAN workloads and opportunistic training workloads on AI-RAN infrastructure. Weaver combines two designs (§2): (1) a RAN-centric spare-compute controller that ensures RAN safety and stabilizes per-slot spare compute, and (2) an elastic, slot-bounded training scheme that packs divided training work into current-slot spare compute. We demonstrate Weaver on a GPU-accelerated 5G platform, showing how RAN-side control and training-side elasticity interact under dynamic RAN load. Live dashboard views show how Weaver converts bursty spare GPU capacity into usable training throughput while preserving RAN performance.

2 Weaver System Design

Key challenges. Harvesting spare GPU compute from RAN infrastructure requires a sharing model different from prior RAN GPU-sharing systems such as CAORA [10] and YinYangRAN [9]. These systems partition GPU streaming multiprocessors (SMs) using mechanisms such as NVIDIA Multi-Instance GPU (MIG) and Multi-Process Service (MPS), then

resize partitions from predicted RAN demand. Such prediction-driven partitioning is too coarse for AI-and-RAN: overestimation wastes training compute, while underestimation risks violating realtime RAN deadlines. Repartitioning also takes ~7 s with MIG and ~0.3 s with MPS, far longer than a RAN slot, and may require suspending workloads or falling back to CPU processing. To that end, the core challenge is not finding spare GPU capacity, but to expose and harvest it at *per-slot timescales*, while not making it too bursty for efficient FM training. Fig. 1 shows that a bursty spare compute supply delivers only 54 TFLOPS, while a stable supply with similar average capacity delivers 142 TFLOPS. Thus, Weaver must transform dynamic per-slot spare compute into a stable, usable training resource while preserving RAN safety.

Our approach. To address this challenge, Weaver enables opportunistic training alongside realtime-sensitive RAN workloads on shared GPU-accelerated RAN infrastructure. Weaver adopts a RAN-first design with fine-grained, per-slot GPU sharing through two systems:

- **RAN-centric spare compute controller.** Weaver integrates a RAN-centric spare compute controller into the MAC scheduler. To prioritize latency-critical RAN processing, the controller uses asymmetric elasticity: it immediately expands GPU resources available to the RAN when demand rises, preventing deadline violations, while reducing RAN allocations conservatively as demand subsides. Beyond protecting RAN performance, this design smooths short-term spare compute fluctuations and turns a bursty resource signal into a stable one (Fig. 1b) that FM training can consume efficiently.
- **Elastic, RAN-slot-bounded training.** Training consumes published spare compute via CUDA Green Contexts [7], which provide SM-level hardware isolation with ~10 μs warm-start reconfiguration (vs. ~300 ms for MPS). To exploit fine-grained spare compute, Weaver decomposes training work into fine-grained units. The key observation is that FM training is dominated by General Matrix-Matrix Multiplication (GEMM), which decomposes into many independent tiles. Weaver treats these tiles as elastic work units and dynamically packs as many as possible into each RAN slot's spare compute. By executing only work that fits the current slot's compute budget and deadline, training adapts to resource fluctuations while maintaining forward progress.

3 Demo Description

This demo showcases Weaver on an NVIDIA DGX Spark running SionnaRK [3], a GPU-accelerated 5G NR stack that integrates OpenAirInterface5G [5] with GPU-based LDPC decoding. It highlights how a co-located FM training workload opportunistically shares GPU resources with live RAN processing at a single cell site.

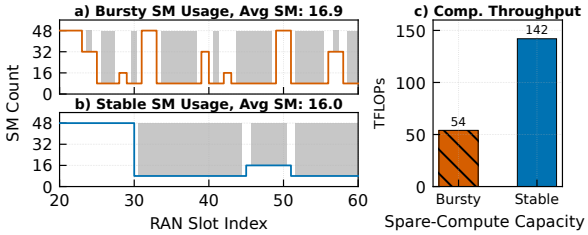


Figure 1: (a) and (b) show bursty and stable RAN SM usage patterns, with gray regions indicating spare compute capacity exposed to FM training; (c) shows their impact on the theoretical training throughput (higher is better).

The demo is driven by replayed production cellular traffic traces [4] and synthetic uplink traffic bursts [2] that emulate rapidly changing RAN load. An interactive dashboard visualizes per-slot RAN GPU demand, the spare compute capacity exposed by Weaver, realtime training throughput, and key RAN performance metrics such as HARQ deadline-met rates. This allows attendees to directly observe how RAN-load fluctuations translate into available spare GPU capacity and how Weaver opportunistically harvests these resources without impacting RAN performance.

To highlight the roles of RAN-side spare compute control and training-side elasticity, we demonstrate the following five setups:

- **Baseline:** RAN-only execution, exposing raw GPU demand from cellular traffic.
- **Peak Provisioning:** FM training uses only the GPU resources left after reserving capacity for worst-case RAN demand, yielding conservative utilization.
- **Elastic Training Only:** FM training adapts to resource availability but follows the raw spare-SM signal exposed by the RAN (Fig. 1a).
- **RAN-Side Control Only:** Weaver's RAN-centric controller stabilizes the spare-SM envelope, but training uses a non-elastic execution strategy, showing that stable spare capacity alone is insufficient when training cannot adapt its work to the current slot budget.
- **Weaver:** Combines elastic training with Weaver's RAN-centric spare compute controller, which stabilizes the spare-SM envelope (Fig. 1b) while ensuring that RAN processing always retains priority.

Key takeaway. This demo shows that neither training-side elasticity nor RAN-side spare compute control alone is sufficient, substantiating the need for both components in Weaver's design to transform dynamic per-slot spare GPU capacity into a stable and efficiently usable resource for practical FM training while preserving RAN performance.

Acknowledgments

This work was supported in part by the UKRI/EPSRC grant UKRI860.

References

[1] AI-RAN Alliance. AI-RAN Alliance Vision and Mission White Paper. https://ai-ran.org/wp-content/uploads/2024/12/AI-RAN_Alliance_Whitepaper.pdf [Accessed: Mar 2026], 2024.

[2] A. Botta, A. Dainotti, and A. Pescapé. A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks*, 56(15):3531–3547, 2012.

[3] S. Cammerer, G. Marcus, T. Zirr, F. Aït Aoudia, L. Maggi, J. Hoydis, and A. Keller. Sionna Research Kit: A GPU-Accelerated Research Platform for AI-RAN. *arXiv preprint arXiv:2505.15848*, 2025.

[4] J. Groen, M. Belgiovine, U. Demir, B. Kim, and K. Chowdhury. TRACTOR: Traffic Analysis and Classification Tool for Open RAN. In *IEEE ICC*, pages 4894–4899, 2024.

[5] F. Kaltenberger, A. P. Silva, A. Gosain, L. Wang, and T.-T. Nguyen. Openairinterface: Democratizing innovation in the 5g era. *Computer Networks*, 176:107284, 2020.

[6] Nokia. Nokia partners with NVIDIA, 2025.

[7] NVIDIA. *CUDA C++ Programming Guide*, 2024. Version 12.6.

[8] P. F. Pérez, C. Fiandrino, and J. Widmer. Characterizing and modeling mobile networks user traffic at millisecond level. In *ACM WiNTECH*, 2023.

[9] L. L. Schiavo, J. A. Ayala-Romero, A. Garcia-Saavedra, M. Fiore, and X. Costa-Perez. YinYangRAN: Resource multiplexing in GPU-accelerated virtualized RANs. In *IEEE INFOCOM*, pages 721–730. IEEE, 2024.

[10] S. D. A. Shah, M. Hafeez, A. Salama, and S. A. R. Zaidi. Proactive AI-and-RAN workload orchestration in O-RAN architectures for 6G networks. *IEEE Open Journal of the Communications Society*, 6:7939–7954, 2025.

[11] L. Xue, T. Wang, X. Z. Khooi, J. Yang, D. Mahendiran, Y. Xia, M. C. Chan, M. Lee, and M. K. Marina. Weaver: A System for AI-RAN Compute Sharing with Foundation Model Training. *arXiv preprint*, Sept 2026.