Learning Within the Laws of the Hardware:
Rethinking Multi-GPU Performance Modeling and Workload Characterization

Urvij Saroliya[1], Eishi Arima[1](Advisor), Martin Kronbichler[2](Advisor), Martin Schulz[1](Advisor)
[1]Chair of Computer Architecture and Parallel Systems, Technical University of Munich, Germany
[2]Faculty of Mathematics, Ruhr University Bochum, Germany

1. INTRODUCTION

ENABLING MULTI-GPU APPLICATION DIAGNOSTICS VIA PREDICTIVE MODELING

CHALLENGES:

- Characterizing untested app configurations requires a performance model that delivers:
  🎯 ***Accurate Predictions.*** 💡 ***Bottleneck Diagnosis.*** 🔭 ***Extrapolation Beyond Data.***
- Analytical models (e.g., Roofline in Fig. 1.1) are interpretable and generalize, but accuracy requires ***in-depth architectural considerations***.
- ML models (e.g., Gradient Boosting Regression in Fig. 1.1) are accurate where data exists but offer **no insights** and **fail to generalize** for OOD samples.

No existing modeling approach delivers all three simultaneously!

PHYSICS-INFORMED NEURAL NETWORK (PINN) [1] PARADIGM:

- Neural networks trained not only on data, but on *governing equations*.
- *Where it applies?* (i) governing laws exist but are alone insufficient, (ii) labeled data is scarce or expensive, (iii) extrapolation beyond training data is critical.

GPU performance modeling satisfies all three!

OUR CONCEPT: HARDWARE-INFORMED NEURAL NETWORKS (HINN) INSPIRED BY PINN

Embed hardware architectural laws as constraints within a learnable model!

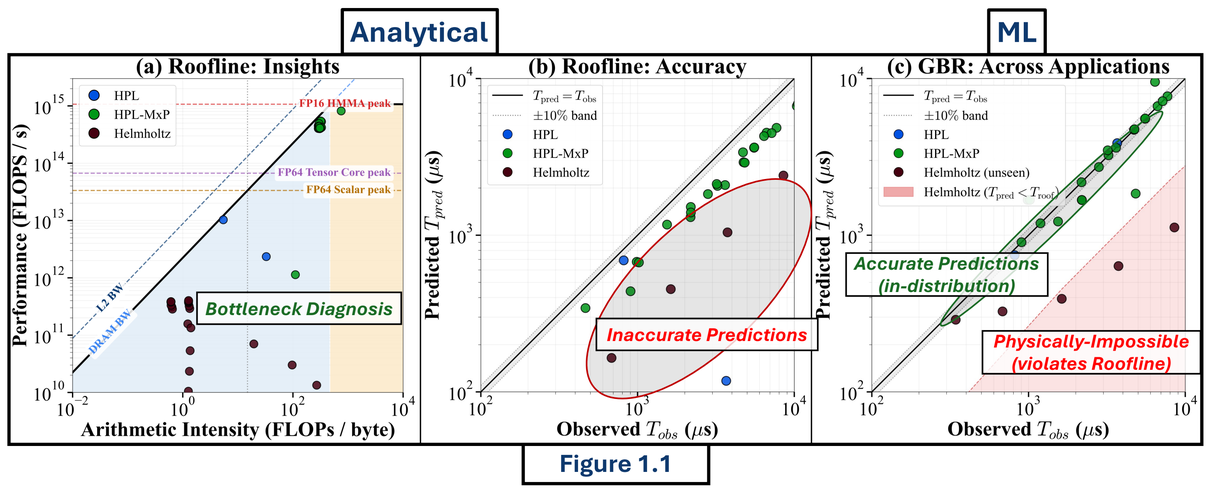


Figure 1.1

KEY CONTRIBUTIONS:

- ✓ *Novel Performance Modeling*: First application of the PINN methodology to GPU applications performance modeling for **accuracy**, **interpretability**, and **generalization**.
- ✓ Evaluation on 5 HPC applications (NVIDIA H100), with cross-device transfer validated on JUPITER (GH200).

2. HARDWARE-INFORMED NEURAL NETWORKS

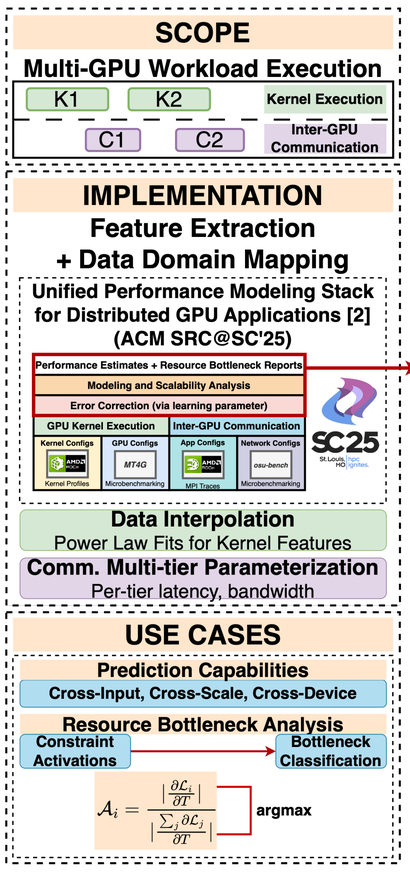
KEY IDEA: Retain analytical models as prior and learn residual constrained by hardware laws.

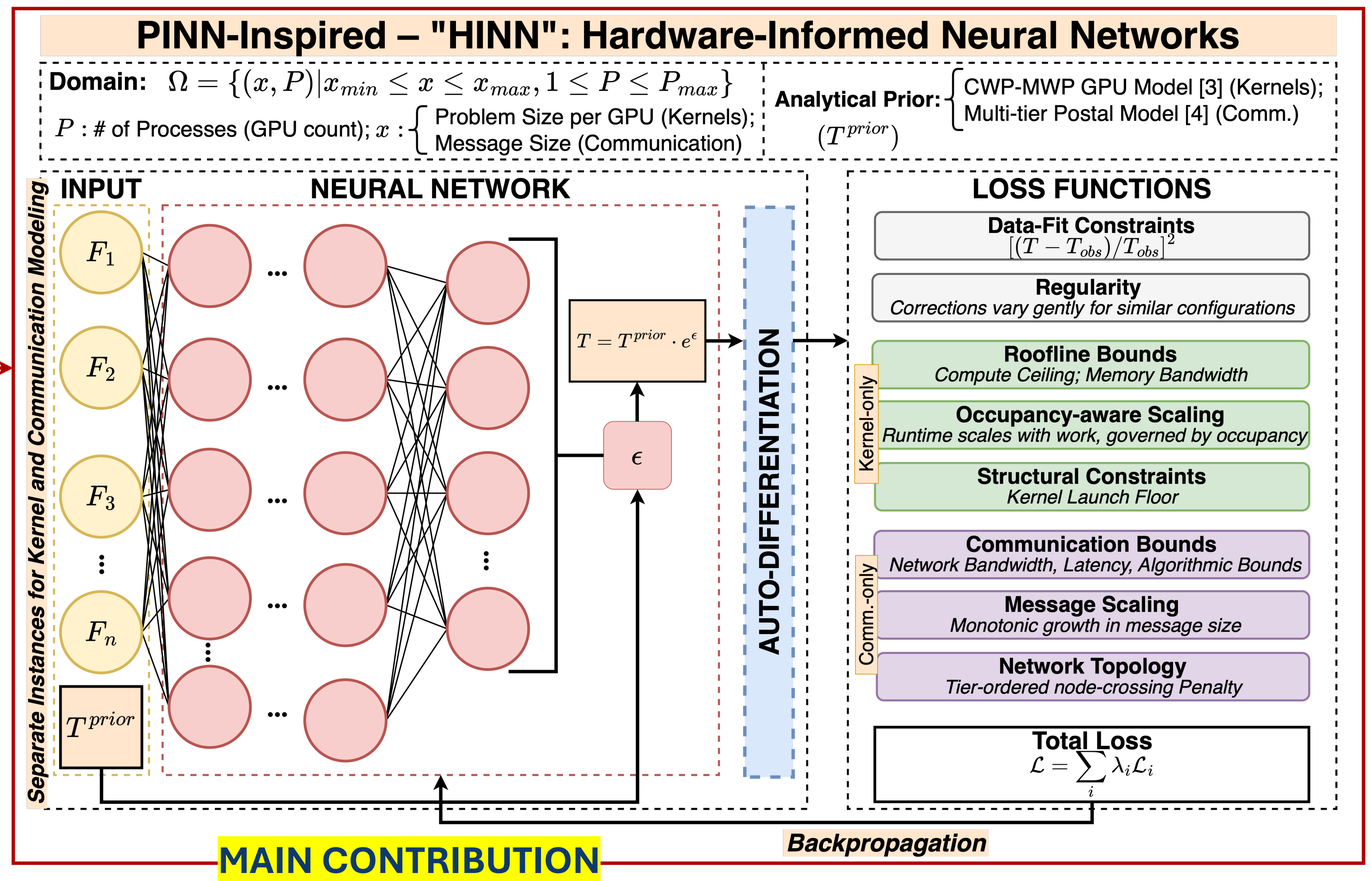
→ *Data is spent on the correction, not on rediscovering known physics.*



CLOSING THE TRADE-OFF:

- **Accuracy –** learned ϵ absorbs what analytical models miss: occupancy, cache effects, stalls, communication overheads.
- **Bottleneck Diagnosis –** constraint activations $\mathcal{A}_i$ reveal bottleneck
  → *A high activation means the prediction sits close to violating constraint.*
- **Generalization –** constraints are enforced across the full domain, not just at data points
  → *Predictions stay physically valid at unseen configs.*





3. EXPERIMENTAL RESULTS

Experimental Setup

Platform 1	Helma [5] (4x NVIDIA H100 94GB HBM2e; NVLink 4.0 / Infiniband)
Platform 2	JUPITER Booster [6] (4x NVIDIA GH200 96GB HBM3; NVLink 4.0 / Infiniband)
Apps	HPCG [7], HPL [7], HPL-MxP [7], hipBone [8], Helmholtz [9]
Baselines	Roofline (Kernel Analytical), Postal (Comm. Analytical), MLP, Random Forest, GBR

EVALUATION METHODOLOGY:

- **Metric:** *median* ***REE*** (relative estimation error) pooled across all configs per app.
- **Dataset:** 5 apps / 16 kernels → 677 train / 170 val (kernel configs); 5.88×10^4 raw MPI events → 2,057 train / 515 val after bucketing per (op, message size, GPU count).
- **Held-out (OOD) Splits:** Helmholtz kernels and hipBone Allgather are excluded from training entirely → tests generalization to unseen applications and unseen collectives.
- All ML baselines use **identical train/val splits and input features** as HINN.

PREDICTION ACCURACY:

- **Kernel HINN: (Fig. 3.1)** ***7.6% overall median REE*** — under 10% on every training app.
- **Comm. HINN: (Fig. 3.2)** ***9.9% overall median REE*** across all MPI operations.
- On unseen kernels/MPI ops, every data-only baseline *collapses*
  - HINN remains consistent.

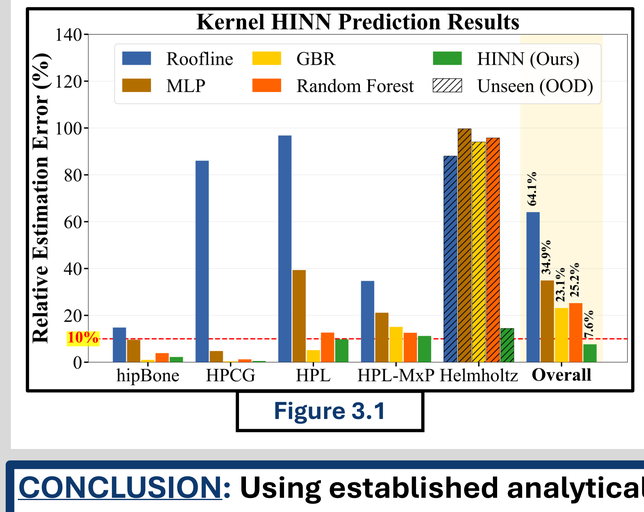


Figure 3.1

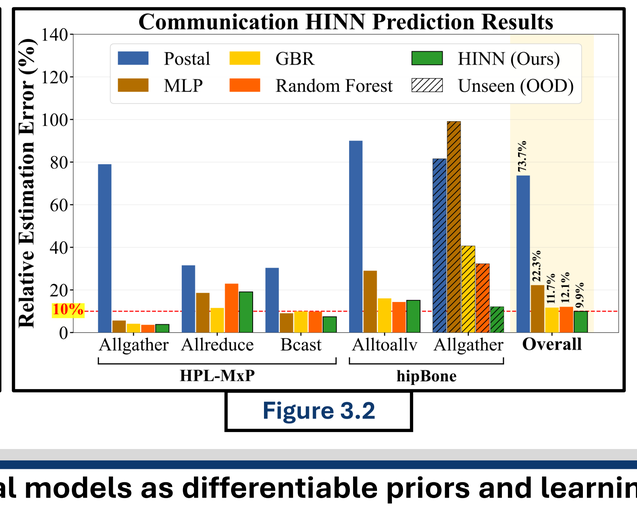


Figure 3.2

BOTTLENECK DIAGNOSIS:

- **Kernel Characterization:**

Classification Rate: 14 / 16 kernels ***(87.5%)***

Kernel	App	Compute	DRAM	L2	Tensor	Match
_occa_hipBoneAx_0	hipBone	2.01×10^{-26}	**0.95**	5.49×10^{-2}	0	✓
sellmv_v1_2D_kernel	HPCG	2.71×10^{-108}	**1.00**	4.72×10^{-4}	0	✓
xmma::gemm::kernel_lds_pipeline	HPL	8.47×10^{-6}	**1.00**	2.19×10^{-7}	2.64×10^{-31}	✗
nvjet_hss_320x128_64x3_1x2_h	HPL-MxP	2.75×10^{-13}	0.40	0.14	**0.47**	✓
LocalHelmholtzOperator	Helmholtz	0	**1.00**	8.32×10^{-9}	0	✓

- **Communication Characterization:**

Classification Rate: 2,544 / 2,544 MPI ops ***(100%)***

Operation	App	BW	LAT	Match
MPI_Alltoallv (m=144MB)	hipBone	**0.997**	3.0×10^{-3}	✓
MPI_Alltoallv (m=31KB)	hipBone	6.7×10^{-2}	**0.933**	✓
MPI_Bcast (m=26MB)	HPL-MxP	**0.952**	4.8×10^{-2}	✓
MPI_Bcast (m=1KB)	HPL-MxP	2.0×10^{-3}	**0.998**	✓
MPI_Allreduce (m=807KB)	HPL-MxP	0.478	**0.522**	✓

GENERALIZATION TO UNSEEN CONFIGURATIONS:

- **Cross-Input: (Fig. 3.3)** trained on smaller problem sizes, evaluated on larger held-out sizes: HINN REE **≤10% for all n**, while ML baselines degrade sharply.
- **Cross-Scale: (Fig. 3.4)** trained on GPU count ∈ {1,2,4,8}, evaluated on {16,32}: **HINN maintains low REE**.
- **Cross-Device: (Fig. 3.5)** Helma (H100) → JUPITER Booster (GH200)
  - *No new hardware counter profiles collected on the JUPITER Booster system*
  - Zero-shot: **26.9%** REE, Few-shot: a handful of wall-clock runtimes used to fine-tune Helma-trained model, reduces REE to **3.95%**.

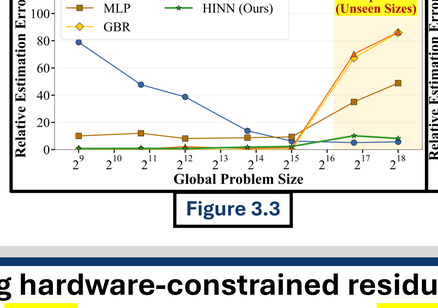


Figure 3.3

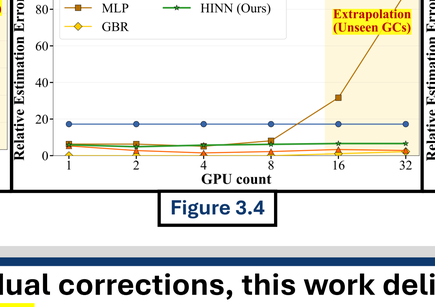


Figure 3.4

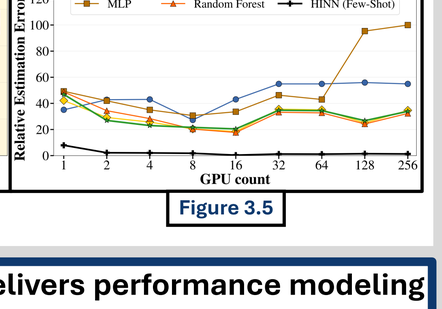


Figure 3.5

CONCLUSION: **Using established analytical models as differentiable priors and learning hardware-constrained residual corrections, this work delivers performance modeling that is accurate, interpretable, and generalizable at once, achieving estimation errors of *7.6%* for GPU kernels and *9.9%* for inter-GPU communications.**

FUTURE WORK: **(1) Model kernel–communication overlap for end-to-end runtime analysis; (2) Cross-device validation beyond NVIDIA Hopper; (3) Energy modeling.**

MODELING STACK DETAILS (SC'25)

REFERENCES:
[1] M. Raissi et al. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, vol. 378, 686–707.
[2] Urvij Saroliya et al. 2025. Unified Performance Modeling Stack for Distributed GPU Applications: Complementing Analytical Insights with Machine Learning. In ACM SRC at The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC Poster), November 2025.
[3] Sunpyo Hong et al. 2009. An analytical model for a GPU architecture with memory-level and thread-level parallelism awareness. In Proceedings of the 36th annual international symposium on Computer architecture. 152–163.
[4] A. Bar-Noy et al. 1992. Designing broadcasting algorithms in the postal model for message-passing systems. In Proceedings of the fourth annual ACM symposium on Parallel algorithms and architectures. 13–22.
[5] NHR@FAU. Helma Cluster. https://doc.nhr.fau.de/clusters/helma/ Accessed: 2026-08-08.
[6] Juelich Supercomputing Centre (JSC). JUPITER – Joint Undertaking Pioneer for Innovative and Transformative Exascale Research. https: //www.fz-juelich.de/en/jsc/jupiter/tech Accessed: 2026-08-08.
[7] NVIDIA Corporation. 2026. NVIDIA HPC Benchmarks. https://docs.nvidia.com/ nvidia-hpc-benchmarks/index.html Accessed: 2026-08-08.
[8] N. Chalmers et al. 2022. Hipbone: A performance-portable gpu-accelerated c++ version of the nekbone Benchmark. arXiv preprint arXiv:2202.12477.
[9] B. Turcksin et al. The deal.II tutorial: Step-64 – Solving a Helmholtz problem with matrix-free methods on the GPU. https://dealii.org/current/doxygen/deal.II/step_64.html Accessed: 2026-08-08.

ACKNOWLEDGEMENT
This work has been supported by the German Federal Ministry of Research, Technology, and Space (BMFTR) through the SCALEXA initiative and the PDExa project (16ME0641). Special thanks extended to the Erlangen National High Performance Computing Center (NHR@FAU, Germany) and Juelich Supercomputing Centre (JSC, Germany), for providing access to the compute resources.

With funding from the: Federal Ministry of Research, Technology and Space

SC26 Chicago, hpc unites