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

Urvij Saroliya
urvij.saroliya@tum.de
Technical University of Munich
Munich, Germany

Eishi Arima (Advisor)
eishi.arima@tum.de
Technical University of Munich
Munich, Germany

Martin Kronbichler (Advisor)
martin.kronbichler@rub.de
Ruhr University Bochum
Bochum, Germany

Martin Schulz (Advisor)
martin.w.j.schulz@tum.de
Technical University of Munich
Munich, Germany

ABSTRACT

Performance modeling enables runtime estimation, scalability analysis, and bottleneck identification without committing large-scale compute resources. However, existing methods rely on either analytical or empirical models: the former are interpretable and generalize by construction but are limited in accuracy by architectural assumptions, while the latter are accurate only where training data exists and offer no insight into performance-limiting factors. Addressing these concerns, we propose Hardware-Informed Neural Networks (HINN), applying the physics-informed neural network (PINN) methodology to multi-GPU performance modeling, where the governing "physics" consists of hardware-derived architectural laws. HINN retains established analytical models as differentiable priors and learns residual corrections subject to hardware-derived constraints enforced via automatic differentiation across the domain. Evaluated on five HPC applications on NVIDIA H100 GPUs, HINN achieves 7.6% and 9.9% median relative estimation error for kernel and communication predictions, while enabling extrapolation to unseen configurations and automated bottleneck classification.

CCS CONCEPTS

• **General and reference** → **Performance**; • **Networks** → **Network performance modeling**.

KEYWORDS

Performance Modeling, GPU Computing, MPI Communication, Machine Learning

1 INTRODUCTION

A performance model should address three concerns simultaneously: (i) prediction accuracy, to guide optimization in the correct direction; (ii) interpretability, to identify which hardware resource limits performance; and (iii) generalization, to extend to problem sizes, GPU counts, and devices unseen during training without exhaustive profiling. No existing approach delivers all three.

Analytical models such as the Roofline model cleanly separate memory-bound from compute-bound execution, but their runtime estimates fall well outside a ±10% band. On the other hand, purely data-driven models such as Gradient Boosting Regression predict accurately in-distribution but degrade sharply on unseen applications and, lacking any physical basis, produce runtimes beyond the Roofline limits – predictions the hardware cannot physically produce.

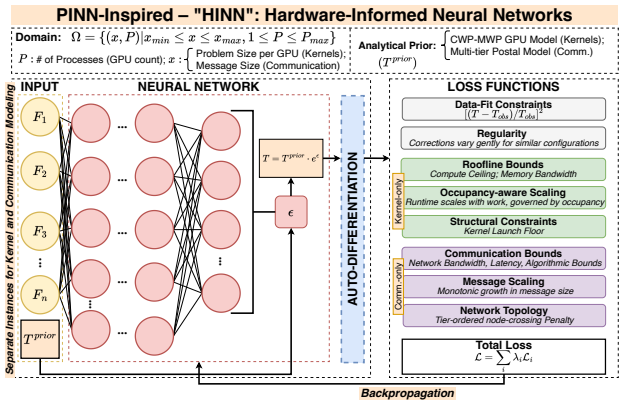


Figure 1: HINN-based Methodology.

Physics-informed neural networks [7] combine a data-fitting term with residual terms penalizing violations of governing equations at collocation points sampled across the domain. This inductive bias reduces the data required and enables extrapolation into regions with no training labels. PINNs apply where (i) governing laws exist but are alone insufficient, (ii) labeled data is scarce or expensive, and (iii) extrapolation beyond training data is critical. GPU performance modeling satisfies all three, making hardware architectural laws a natural analogue of differential equations.

2 HINN: HARDWARE INFORMED NEURAL NETWORKS

Figure 1 illustrates the HINN formulation. Kernel execution and inter-GPU communication are modeled as two independent instances of the same structure. For a given problem size n (or message size m) and GPU count P, an analytical prior T^{prior} is computed from platform parameters and application profiles – while a multilayer perceptron outputs a single scalar correction ϵ, providing

runtime estimates:

$$T = T^{prior} e^{\epsilon} \tag{1}$$

The exponential form guarantees positive predictions and is consistent with log-transformed inputs, as the multiplicative prior becomes additive in log space. As $\epsilon \to 0$ the prediction falls back onto the analytical model rather than onto zero: learning is confined to the residual, and the analytical structure is never re-learned.

We adopt the CWP-MWP model [4] as the kernel prior and the tier-aware postal model [1] as the communication prior, the latter capturing bandwidth drops across node boundaries. Treating both inputs as continuous enables constraint enforcement at arbitrary points via automatic differentiation. Hardware-derived constraints govern learning (Figure 1, right); and the total loss used for training (via backpropagation) combines these constraint penalties with weights λ_i.

Constraints are evaluated at collocation points sampled across the entire domain, including regions where no profiling data exists — this is what makes extrapolation principled rather than accidental. Additionally, gradients applied on these constraint provide diagnostics. The constraint activation:

$$\mathcal{A}_i = \frac{|\frac{\partial \mathcal{L}_i}{\partial T}|}{|\frac{\sum_j \partial \mathcal{L}_j}{\partial T}|} \tag{2}$$

quantifies how strongly constraint C_i binds the prediction at a query point; a high $\mathcal{A}_i$ means the prediction sits close to violating it. The dominant bottleneck follows as $argmax(\mathcal{A})$, requiring only the trained weights and the query point.

Implementation – Feature extraction and prior parameterization build on our unified performance modeling stack [8], integrating kernel profiling, MPI tracing, GPU specification discovery, and network micro-benchmarking.

3 RESULTS

We evaluate on the NHR@FAU Helma cluster [6] (NVIDIA H100) across five HPC applications – HPCG [3], HPL [3], HPL-MxP [3], hipBone [2], and Helmholtz [9] — against analytical (Roofline, postal) and data-only (MLP, Random Forest, GBR) baselines using identical splits and features.

Prediction Accuracy – The kernel and communication HINNs attain 7.6% and 9.9% overall median relative estimation error (REE), versus >20% for data-only baselines and ≈64% for Roofline. On held-out applications and collectives, data-only baselines exceed 80% REE while HINN remains consistent. **Bottleneck Analysis –** Constraint activations match the measured bottleneck for 14/16 kernels (87.5%) and all 2,544 MPI operations (binary latency-versus-bandwidth). **Generalization –** HINN stays within 10% REE on held-out larger problem sizes and at 16–32 GPUs when trained on 1-8. Transferred from H100 to GH200 (JUPITER Booster [5]) without new profiles, it reaches 26.9% zero-shot and 3.95% after few-shot calibration – the constraints carry the transfer, data resolves only the residual offset.

Future work includes modeling kernel–communication overlap for end-to-end runtime analysis, cross-vendor validation beyond NVIDIA Hopper architecture, and energy modeling.

ACKNOWLEDGMENTS

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.

REFERENCES

[1] Amotz Bar-Noy and Shlomo Kipnis. 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.

[2] Noel Chalmers, Abhishek Mishra, Damon McDougall, and Tim Warburton. 2022. HipBone: A performance-portable GPU-accelerated C++ version of the NekBone benchmark. *arXiv preprint arXiv:2202.12477* (2022).

[3] NVIDIA Corporation. 2026. *NVIDIA HPC Benchmarks*. https://docs.nvidia.com/nvidia-hpc-benchmarks/index.html Accessed: 2026-02-04.

[4] Sunpyo Hong and Hyesoon Kim. 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.

[5] Juelich Supercomputing Centre (JSC). 2026. JUPITER – Joint Undertaking Pioneer for Innovative and Transformative Exascale Research. https://www.fz-juelich.de/en/jsc/jupiter/tech. Accessed: August 2026.

[6] NHR@FAU. 2026. Helma Cluster. https://doc.nhr.fau.de/clusters/helma/. Accessed: August 2026.

[7] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics* 378 (2019), 686–707.

[8] Urvij Saroliya, Eishi Arima, and Martin Kronbichler. 2025. Unified Performance Modeling Stack for Distributed GPU Applications: Complementing Analytical Insights with Machine Learning. (2025).

[9] Bruno Turcksin and Daniel Arndt. 2024. 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.