Scaling Out *hls4ml* for ASICs: A Surrogate Model for Neural Network Accelerator Synthesis Metrics

Arghya Ranjan Das[1] **Jason Weitz**[2] **Kyle Woodworth**[7] **Hershel Weiner**[3]
Benjamin Hawks[7] **Aaron Young**[4] **Angelos Ioannou**[5] **Javier Duarte**[2]
Davide Braga[7] **Mia Liu**[1] **David Burnette**[6] **Farah Fahim**[7]
Nhan Tran[7] **Giuseppe Di Guglielmo**[7]

[1]Purdue University [2]University of California San Diego [3]The University of Texas
[4]Oak Ridge National Laboratory [5]Lawrence Berkeley National Laboratory
[6]Siemens EDA [7]Fermi National Accelerator Laboratory

Abstract

Custom ASIC accelerators deliver large power and performance gains for machine learning in scientific and edge computing, but each design needs hours of synthesis to obtain area and timing, making design-space exploration and agentic design flows infeasible. A surrogate model predicts these metrics directly from a design specification and synthesis directives, sidestepping the bottleneck. Such models exist for *hls4ml* accelerators on FPGAs [1, 2], but extending them to ASICs and generalizing beyond the training distribution remain open. We present a dataset of over half a million dense neural network accelerators generated by *hls4ml* and synthesized with Siemens Catapult HLS on Nangate $45\,\mathrm{nm}$, and train Transformer and Graph Neural Network surrogates for area and latency. Our best model reaches latency $R^2 = 0.9997$ and area $R^2 = 0.9991$ on held-out designs, and a sum-decomposition strategy sustains $R^2 = 0.9954$ and 0.9797 when generalizing to networks deeper than any seen in training. Finally, fine-tuning the Nangate $45\,\mathrm{nm}$ model to GlobalFoundries 22FDX with only 10% of the data establishes open-library pre-training as a scalable path to nodes whose PDKs are not yet ready for large synthesis campaigns, such as cryogenic 22FDX for in-cryostat qubit readout.

1 Introduction

Machine learning in scientific and edge computing increasingly relies on custom ASIC accelerators for their power and performance advantages, and superconducting qubit readout is a demanding instance, requiring real-time, low-latency classification of qubit states. Current workflows deploy *hls4ml*-generated neural network accelerators [3, 4] on Xilinx RFSoC FPGAs at room temperature [5, 6]. Moving the readout electronics into the cryostat, closer to the qubits, requires custom ASICs on cryo-compatible nodes such as GlobalFoundries 22 nm FD-SOI (GF22FDX) [7]. Cryo-CMOS ASICs have demonstrated in-fridge control and readout for both superconducting and semiconductor qubits [8–14], but real-time neural-network state discrimination as a cryogenic ASIC remains largely unexplored. *hls4ml* offers an automated path from trained networks to ASIC implementations through Siemens Catapult HLS [15], but a single synthesis run takes minutes to hours, making exhaustive exploration of depth, width, bitwidth, and reuse factor infeasible and prohibitive for agentic design flows whose optimization loops may evaluate thousands of candidates. Surrogate models that predict synthesis metrics from a design specification cut this evaluation from hours to milliseconds. Such models exist for *hls4ml* on FPGAs [1, 2], but two challenges separate that work from the cryo-ASIC setting. First, ASIC metrics (standard-cell area, latency, and initiation interval) differ from FPGA resource counts, and no large-scale ASIC dataset for *hls4ml* exists, so both a dataset and new models

Submitted to 7th Fast Machine Learning for Science Conference (FastML 2026). Do not distribute.

are required. Second, the surrogate must generalize beyond its training distribution along two axes: across nodes, since extending PDKs such as GF22FDX to cryogenic operation is slow, so we train at scale on the open Nangate $45\,\mathrm{nm}$ library and transfer to room-temperature GF22FDX; and across depth, where a surrogate trained on shallow networks must predict deeper ones, a setting in which standard models degrade sharply. We address both challenges and make the following contributions:

- A dataset of ≈500,000 *hls4ml*/Catapult designs on Nangate $45\,\mathrm{nm}$ spanning depth, width, bitwidth, and reuse factor, the first at this scale for ASIC-targeted NN accelerators.
- Transformer and Graph Neural Network surrogates reaching latency $R^2 = 0.9997$ and area $R^2 = 0.9991$; throughput is computed exactly as $\mathrm{II} = \max_i \mathrm{II}_i$, with no learned component.
- A *sum-decomposition* strategy that restores accuracy on networks deeper than any seen in training, reaching latency $R^2 = 0.9954$ and area $R^2 = 0.9797$.
- Cross-node transfer: a $45\,\mathrm{nm}$ model fine-tunes to room-temperature GF22FDX with ≈50,000 designs, reaching latency $R^2 = 0.9984$ and area $R^2 = 0.9393$.

2 Dataset

Design space. Each design point is a dense MLP accelerator defined by five axes: network depth $L \in \{1, \ldots, 10\}$; input and per-layer widths $w_i \in \{4, 8, 16, 32, 64\}$; a uniform quantization bitwidth $b \in \{4, 6, 8, 10, 12, 14\}$ on all weights and activations; per-layer activation $\sigma_i \in \{\mathrm{ReLU}, \mathrm{sigmoid}, \tanh\}$; and the *hls4ml*/Catapult reuse factor $\mathrm{RF} \in \{1, 4, 8, 16\}$, which trades area for throughput by time-sharing multiply-accumulate units. The number of unique (w, b, σ) architectures grows from 450 at $L = 1$ to $\approx 1.5 \times 10^6$ at $L = 4$, so we combine two sampling strategies.

Sweep strategy. For $L \leq 3$ the full Cartesian product is tractable: we synthesize all 450, $6{,}750$, and $101{,}250$ width/bitwidth/activation architectures, each at all four reuse factors ($1{,}800$, $27{,}000$, and $405{,}000$ designs). For $L \in \{4, 5, 6, 8, 10\}$ we instead draw $N = 2{,}500$ unique configurations per depth with a Latin Hypercube Sampler [16] over the $(2 + 2L)$-dimensional space, snapping each sample to the discrete grid and discarding collisions; synthesizing at all four reuse factors yields $10{,}000$ designs per depth.

Infrastructure. For each architecture a QKeras [17] model is built through *hls4ml*'s API and synthesized with Siemens Catapult HLS targeting the Nangate Open Cell Library 45 nm [18], with no manual intervention. Jobs run on NERSC Perlmutter under SLURM with 300 concurrent synthesis slots; each report is parsed for total cell area ($\mu\mathrm{m}^2$), latency (clock cycles), and initiation interval, then archived per design. Throughput is derived analytically from the reuse factor and clock period and cross-checked against the Catapult report.

Statistics. The Nangate 45 nm portion comprises $483{,}772$ designs at $L = 1$–10; a GF22FDX portion of $43{,}521$ designs at $L = 1$–3 supports the transfer experiments (Sec. 4), for $527{,}293$ in total (Table 1). Cartesian sweeps give uniform coverage at $L \leq 3$, while LHS coverage beyond is broad rather than exhaustive. Median synthesis time is ≈20 min per design, with a heavy tail: $\mathrm{RF} = 1$ designs at maximum width and bitwidth reach several hours under full unrolling.

Table 1: Dataset composition. "Cartesian" is exhaustive enumeration; "LHS" is Latin Hypercube Sampling ($N = 2{,}500$ architectures per depth, each at four reuse factors).

Technology	Depth(s)	Strategy	Designs
Nangate 45 nm	$L = 1, 2, 3$	Cartesian	433,800
Nangate 45 nm	$L = 4, 5, 6, 8, 10$	LHS	49,972
GF22FDX	$L = 1$	Cartesian	1,800
GF22FDX	$L = 2, 3$	LHS	41,721
Total			**527,293**

3 Surrogate Models

We use two model families, a Transformer [19] and Graph Neural Networks (GNNs), predicting area and latency as learned regression targets while computing throughput analytically.

Design encoding. All designs are depth-L MLPs, so we represent each as an ordered set of L layer tokens (Fig. 1). Each token is a 33-dimensional feature vector: standardized numerical attributes (input and output widths, bitwidth b, reuse factor RF) and one-hot categorical attributes (layer type, activation). Two design-global *hls4ml* directives, i.e. strategy (Latency/Resource) and the I/O type (parallel/stream), form a 4-dimensional per-design vector concatenated at the readout head rather than added to each token. The Transformer consumes the tokens as a length-L sequence; the GNNs consume them as a line graph whose edges follow the dataflow between consecutive layers.

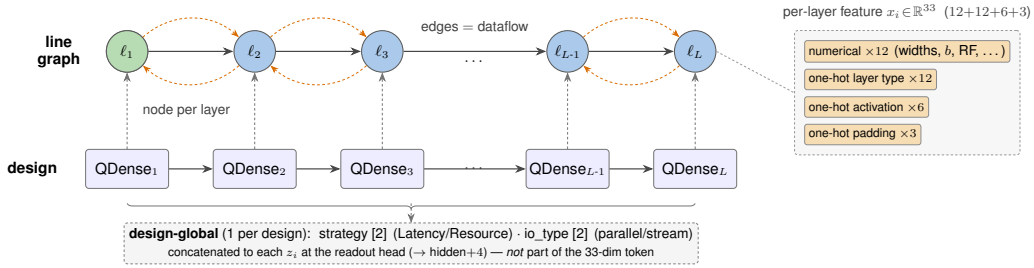

Figure 1: Design encoding. Each layer becomes a node in a directed line graph carrying a 33-dimensional feature token; the *hls4ml* directives are concatenated to the node embeddings at the readout head.

Architectures. The Transformer is a self-attention encoder over the layer tokens with a regression head (∼3M parameters). The GNNs combine one of three encoder configurations (as shown in Appendix) with one of two readouts. The encoders are a GRAPHSAGE [20] backbone, a lightweight GATv2 [21] that averages attention heads, and a full GATv2 that concatenates heads with a residual projection. The readouts are a *pooling* head that aggregates the node embeddings, and a *sum-decomposition* head that applies a weight-shared MLP h to each embedding z_i and sums the non-negative per-layer costs,

$$\hat{y} = \sum_{i=1}^{L} \exp\big(h(z_i)\big). \tag{1}$$

This yields four GNN surrogates: GRAPHSAGE with sum-pooling (∼0.6M; Fig. A.2), the full GATv2 with multi-pooling (∼54M; Fig. A.1), and two sum-decomposition models, an averaged-head GATv2 (∼1.7M; Fig. A.3) and a concat-head GATv2 (∼54M; Fig. A.4).

Throughput as an exact formula. The pipelined accelerators from *hls4ml* run as a sequence of streaming stages, so the steady-state initiation interval is set by the slowest stage, $\mathrm{II} = \max_i \mathrm{II}_i$, with each II_i fixed by the layer width, bitwidth b, reuse factor RF, and clock. We therefore compute throughput analytically from a per-layer lookup rather than learning it, which is exact ($R^2 = 1.000$) and leaves only area and latency to the learned models.

Training and in-distribution accuracy. Models minimize mean-squared error in normalized log-space with AdamW [22] and early stopping, on a random 70/15/15 split of the $L = 1, 2, 3$ designs (303,660 train, 65,070 test). The 54.47M GATv2 attains latency $R^2 = 0.9997$ and area $R^2 = 0.9991$ (Table 2), with throughput exact throughout. The pooling readouts interpolate well but fail on networks deeper than any seen in training (Sec. 4), where the sum-decomposition readout succeeds.

4 Generalizing Beyond the Training Distribution

A surrogate model must predict networks deeper than those it trained on, and a surrogate ported to a new ASIC technology node must do so from little data. We test both, writing $\{\cdot\} \to L$ for the training depths and held-out test depth (e.g. $\{1, 2\} \to 3$).

Table 2: In-distribution accuracy (random $70/15/15$ split of $L = 1, 2, 3$ designs, 65,070 test). Values are the mean over 4 seeds; throughput is computed analytically and exact ($R^2 = 1.000$) for all models.

Model	Params	Latency R^2	Area R^2
Transformer	3.2M	0.9935	0.9905
GraphSAGE (sum-pool)	0.59M	0.9925	0.9919
GATv2 (multi-pool)	54.47M	**0.9997**	**0.9991**

Deeper networks. Despite near-perfect in-distribution accuracy (Table 2), the pooled GATv2 collapses on networks deeper than any seen in training: $\{1, 2\} \to 3$ falls to latency $R^2 = 0.634$ and area $R^2 = 0.789$, with both targets negative (worse than predicting the mean) by $L = 10$ (Table 3).

Sum-decomposition. The sum-decomposition readout (Eq. 1) is built for this regime. Because latency and area accumulate across layers and h predicts only an in-range per-layer term, a deeper design adds one more such term and never extrapolates to an unseen magnitude. This restores $\{1, 2\} \to 3$ to latency $R^2 = 0.9954$ and area $R^2 = 0.9797$ and holds far deeper, keeping latency $R^2 = 0.9853$ at $L = 10$ where the pooled model reaches -1.15. The gap is the readout, not capacity: the two 54M models share an encoder, yet only sum-decomposition extrapolates.

Table 3: Prediction on networks deeper than any seen in training (R^2, mean over 4 seeds). *Pooled* = 54.47M GATv2 with pool→MLP readout; *Sum-Decomp* = exp-sum readout. $\{1, 2\} \to L$ denotes training on depths $1, 2$ and testing at depth L.

	Pooled GATv2 54.47M		Sum-Decomp 1.70M		Sum-Decomp 54M	
Train → Test	Lat.	Area	Lat.	Area	Lat.	Area
$\{1, 2\} \to 3$	0.634	0.789	**0.9954**	**0.9797**	0.977	0.962
$\{1, 2\} \to 6$	-0.448	0.239	0.983	0.949	0.980	0.947
$\{1, 2\} \to 10$	-1.152	-0.181	0.9853	0.9252	0.9816	0.9343
$\{1, 2, 3\} \to 10$	-0.615	0.273	**0.9998**	0.9739	0.9894	**0.9924**

Cross-node transfer. We then transfer the $45\,\mathrm{nm}$ sum-decomposition models to a room-temperature GF22FDX set of 43,521 designs at $L = 1$–3, about 10% of the $45\,\mathrm{nm}$ corpus. Latency and throughput are cycle counts and transfer near zero-shot; only area is node-dependent, as GF22FDX cells are $\approx 3.5\times$ denser. Fine-tuning only the readout head (Fig. A.5; 2.5% of parameters) from a deep $45\,\mathrm{nm}$ prior reaches latency $R^2 = 0.995$ and area $R^2 = 0.970$, matching or beating from-scratch training without seeing a $22\,\mathrm{nm}$ 3-layer design (Table 4).

Table 4: Cross-node transfer to GF22FDX (R^2, mean over 4 seeds). Train/fine-tune (FT) and test are both on $22\,\mathrm{nm}$, holding out depth 3 (train on $\{1, 2\}$, test on the 8,000-design 3-layer set); FT initializes from a $45\,\mathrm{nm}$ prior and trains only the readout head (2.5% of parameters).

Experiment Setup			Sum-Decomp 1.70M		GATv2 Sum-Decomp 54M	
Train / FT → Test	Initialization	Training	Lat. R^2	Area R^2	Lat. R^2	Area R^2
$22\,\mathrm{nm}$ $\{1, 2\} \to 3$	random	all params	0.978	0.950	0.945	0.912
$22\,\mathrm{nm}$ $\{1, 2\} \to 3$	$45\,\mathrm{nm}$ $\{1, 2\}$	head only	0.956	0.945	0.924	0.888
$22\,\mathrm{nm}$ $\{1, 2\} \to 3$	$45\,\mathrm{nm}$ $\{1\text{–}5\}$	head only	**0.995**	**0.970**	**0.966**	**0.961**

5 Conclusions

We presented a large-scale ASIC synthesis dataset for *hls4ml* neural-network accelerators and Transformer and GNN surrogates that predict area and latency to $R^2 > 0.999$ in distribution. A sum-decomposition readout extends this accuracy to networks far deeper than any seen in training, and open-library pre-training transfers a $45\,\mathrm{nm}$ surrogate to GF22FDX from $\approx 10\%$ of the data. Future work extends the design space to deeper and convolutional models and to additional technology nodes, including the cryogenic GF22FDX PDK targeted for in-cryostat qubit readout.

References

[1] M. M. Rahimifar et al. rule4ml: An Open-Source Tool for Resource Utilization and Latency Estimation for ML Models on FPGA, 2024.

[2] B. Hawks et al. wa-hls4ml: A Benchmark and Surrogate Models for hls4ml Resource and Latency Estimation, 2025. URL `https://arxiv.org/abs/2511.05615`.

[3] FastML Team. fastmachinelearning/hls4ml, 2025. URL `https://github.com/fastmachinelearning/hls4ml`.

[4] J. Duarte et al. Fast inference of deep neural networks in FPGAs for particle physics. *Journal of Instrumentation*, 13(07):P07027, 2018. doi: 10.1088/1748-0221/13/07/P07027.

[5] L. Stefanazzi et al. The QICK (Quantum Instrumentation Control Kit): Readout and control for qubits and detectors. *Review of Scientific Instruments*, 93(4):044709, 2022. doi: 10.1063/5.0076249. arXiv:2110.00557.

[6] G. Di Guglielmo et al. End-to-end workflow for machine learning-based qubit readout with QICK and hls4ml, 2025.

[7] R. Carter et al. 22nm FDSOI technology for emerging mobile, Internet-of-Things, and RF applications. In *2016 IEEE International Electron Devices Meeting (IEDM)*, pages 2.2.1–2.2.4, December 2016. doi: 10.1109/IEDM.2016.7838029.

[8] J. C. Bardin et al. Design and Characterization of a 28-nm Bulk-CMOS Cryogenic Quantum Controller Dissipating Less Than 2 mW at 3 K. *IEEE Journal of Solid-State Circuits*, 54(11):3043–3060, 2019. doi: 10.1109/JSSC.2019.2937234.

[9] S. Van Winckel et al. A 28nm 6.5–8.1GHz 1.16mW/qubit Cryo-CMOS System-on-Chip for Superconducting Qubit Readout. In *ESSCIRC 2022 - IEEE 48th European Solid State Circuits Conference (ESSCIRC)*, pages 61–64. IEEE, 2022. doi: 10.1109/ESSCIRC55480.2022.9911493.

[10] K. Kang et al. A 40-nm Cryo-CMOS Quantum Controller IC for Superconducting Qubit. *IEEE Journal of Solid-State Circuits*, 57(11):3274–3287, 2022. doi: 10.1109/JSSC.2022.3198663.

[11] J. Yoo et al. Design and Characterization of a <4-mW/Qubit 28-nm Cryo-CMOS Integrated Circuit for Full Control of a Superconducting Quantum Processor Unit Cell. *IEEE Journal of Solid-State Circuits*, 58(11):3044–3059, 2023. doi: 10.1109/JSSC.2023.3309317.

[12] X. Xue et al. CMOS-Based Cryogenic Control of Silicon Quantum Circuits. *Nature*, 593:205–210, 2021. doi: 10.1038/s41586-021-03469-4.

[13] J. Park et al. A Fully Integrated Cryo-CMOS SoC for State Manipulation, Readout, and High-Speed Gate Pulsing of Spin Qubits. *IEEE Journal of Solid-State Circuits*, 56(11):3289–3306, 2021. doi: 10.1109/JSSC.2021.3115988.

[14] A. Ruffino et al. A Cryo-CMOS Chip That Integrates Silicon Quantum Dots and Multiplexed Dispersive Readout Electronics. *Nature Electronics*, 5:53–59, 2021. doi: 10.1038/s41928-021-00687-6.

[15] Siemens Digital Industries Software. Siemens Simplifies Development of AI Accelerators for Advanced System-on-Chip Designs with Catapult AI NN. Press release, May 2024. URL `https://news.siemens.com/en-us/siemens-catapult-ai-nn/`.

[16] M. D. McKay et al. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245, 1979. doi: 10.1080/00401706.1979.10489755.

[17] Claudionor N. Coelho et al. Automatic Heterogeneous Quantization of Deep Neural Networks for Low-Latency Inference on the Edge for Particle Detectors. *Nature Machine Intelligence*, 3(8):675–686, 2021. doi: 10.1038/s42256-021-00356-5.

[18] NanGate, Inc. NanGate FreePDK45 Open Cell Library. https://si2.org/open-cell-and-process-design-kit, 2008.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[20] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[21] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations (ICLR)*, 2022.

[22] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

A Appendix: Model Architectures and Encodings

This appendix collects the model-architecture and training diagrams referenced in the main text. Each GNN surrogate is drawn as a full tower (encoder + readout).

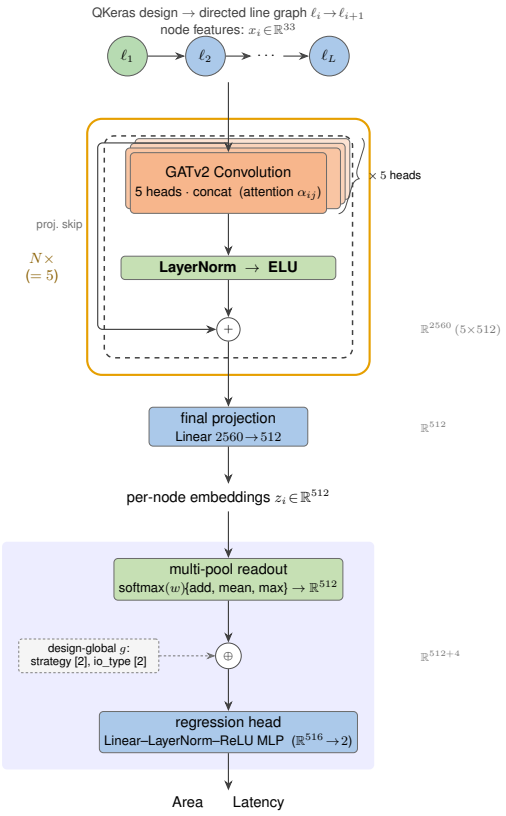

Figure A.1: The GATv2 *lui-gnn* surrogate: a GATv2 encoder (5 layers, 5 heads, concat + residual) with a learnable multi-pool readout and an MLP head (54M parameters).

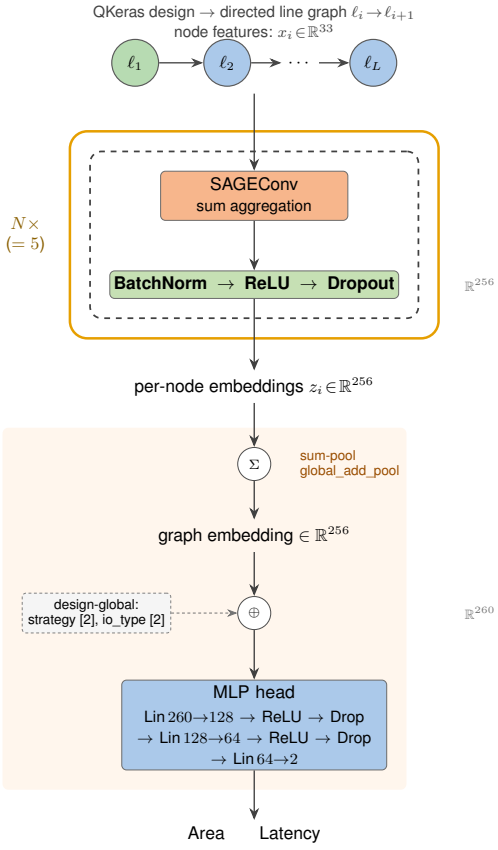

Figure A.2: The GraphSAGE baseline: a SAGEConv encoder (sum aggregation) with a sum-pool readout and an MLP head (0.6M parameters).

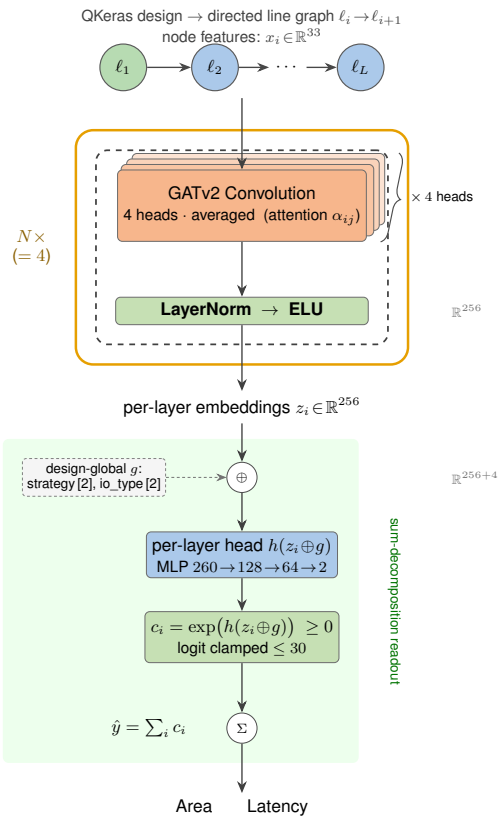

Figure A.3: The small (1.70M) sum-decomposition surrogate: a GATv2 encoder with the per-layer $\exp \to \Sigma$ readout.

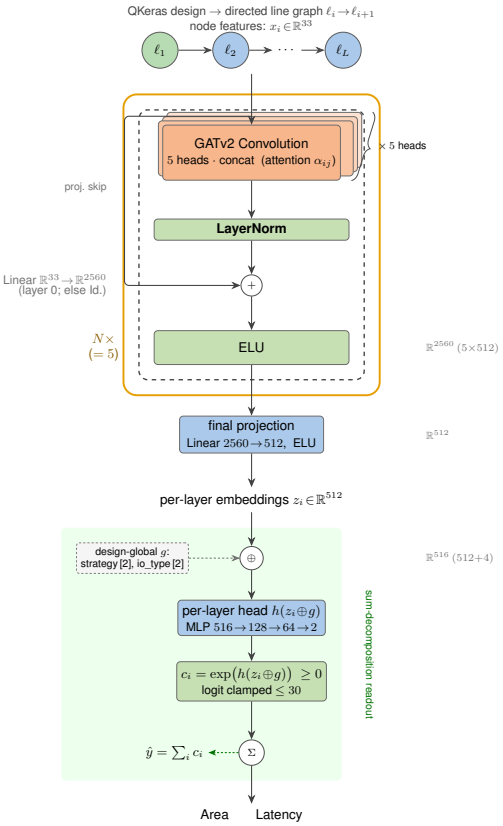

Figure A.4: The 54M sum-decomposition surrogate: the *lui-gnn* GATv2 encoder with the per-layer $\exp \to \Sigma$ readout.

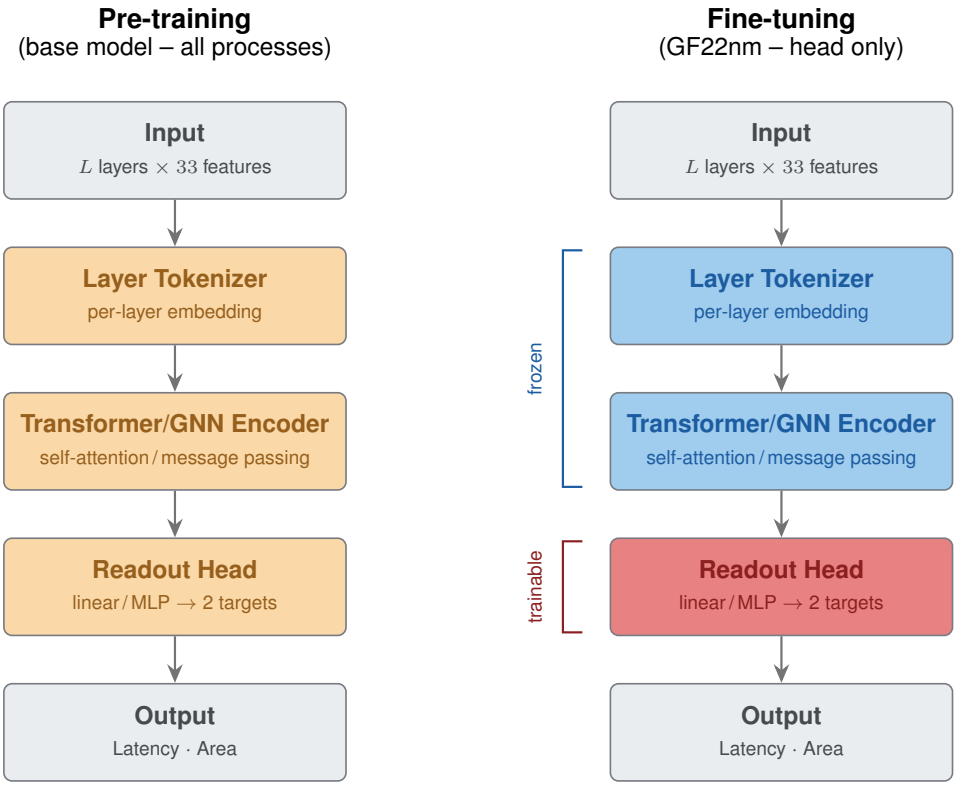

Figure A.5: Head-only cross-node fine-tuning. A model pre-trained on $45\,\mathrm{nm}$ (all parameters trainable, left) is adapted to $22\,\mathrm{nm}$ by *freezing the encoder* (blue) and training *only the readout head* (red), a small fraction of the parameters.