Resource Scheduler for Real-time Machine Learning

Suyash Vardhan Singh [1] Iftakhar Ahmad [1] David Andrews [2]
Miaoqing Huang [2] Austin Downey [1] Jason D. Bakos [1]

[1]University of South Carolina [2]University of Arkansas

Abstract

Data-driven physics models offer the potential for substantially increasing the sample rate for applications in high-rate cyber-physical systems, such as model predictive control, structural health monitoring, and online smart sensing. Making this practical requires new model deployment tools that search for networks with maximum accuracy while meeting both real-time performance and resource constraints. Tools that generate customized architectures for machine learning models, such as HLS4ML and FINN, require manual control over latency and cost trade-offs for each layer. This poster describes the proposed end-to-end framework that combines Bayesian optimization for neural architecture search with Integer Linear Optimization of layer cost-latency trade-off using HLS4ML "reuse factors".

Introduction

There is increasing interest in the development of "high-rate" cyber-physical systems, which are defined as systems that make decisions at a rate exceeding 1 KHz. These systems often require a model that makes predictions of physical phenomena, but their execution time constraints make the use of physics-based models impractical. For this reason, there is an increasing interest in using lightweight machine learning models as surrogates. There is a general expectation that these models should simultaneously deliver prediction accuracy equivalent to $> 30\,dB$ of signal-to-noise ratio with $< 1\,ms$ latency, depending on the exact nature of the application targeted. At this time, there are limited options for tool flows that are capable of delivering this level of performance while guaranteeing a bounded inference time and meeting resource constraints when using an embedded-class FPGA. Achieving these goals requires co-optimization of network hyperparameters while optimizing the hardware reuse factor in each layer of the deployed architecture.

Tool Flow and Neural Network used for Dropbear

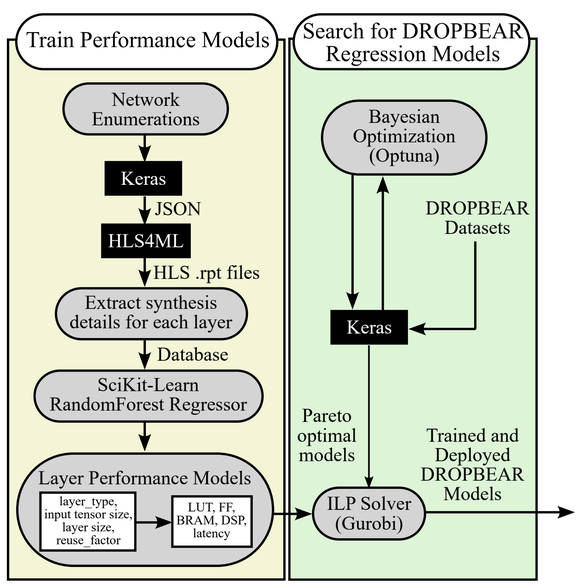

Figure 1. Overview of the tool flow used.

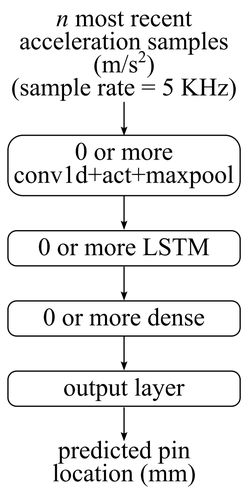

Figure 2. Neural network structure used for DROPBEAR.

Cost Trade-OFF

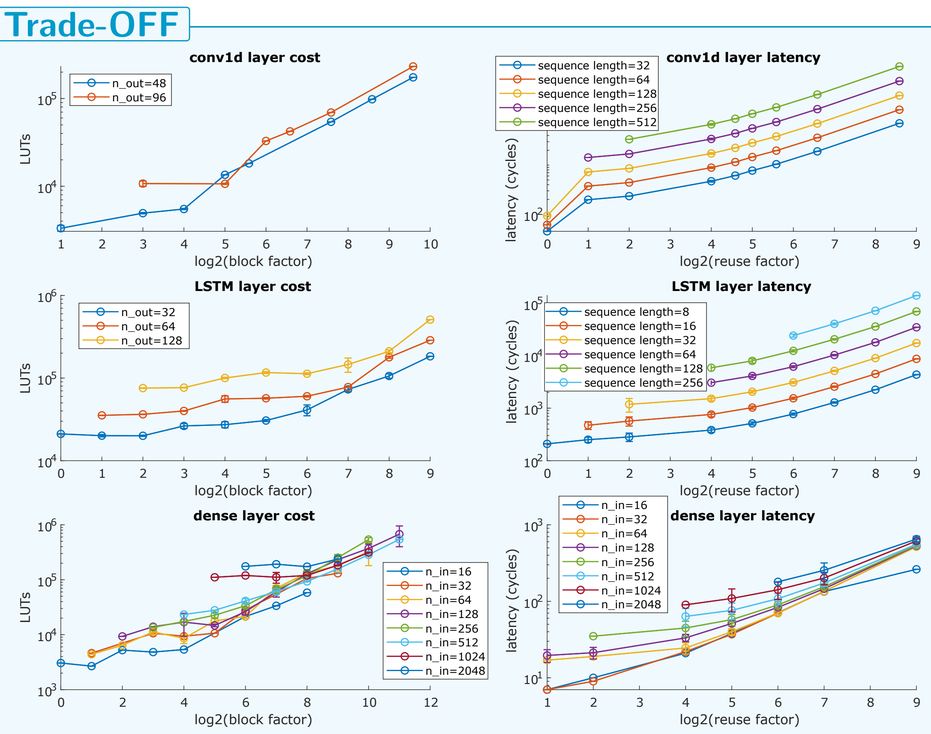

Figure 3. LUT cost and latency when scaling size of hardware GEMV unit (as block factor = number of scalar multipliers) for three types of HLS4ML layers. Note that each data point represents a set of observations, as many layer configurations map to the same independent variables shown in the plots; error bars indicate standard deviation.

Network Hyperparameter Search

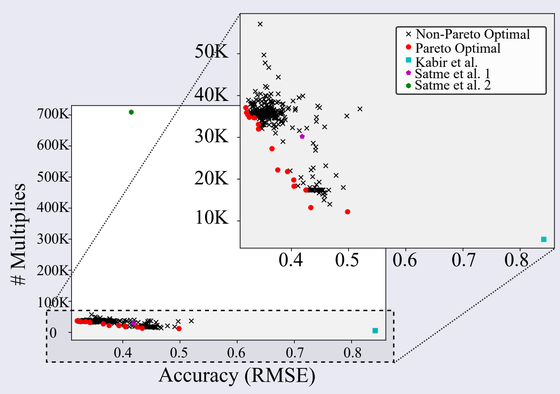

Figure 4. Pareto optimal model configurations for accuracy and cost. Included are the positions of Satme et al. network 1 and 2 (purple and green dots) Satme et al. 2022 and Kabir et al. (cyan square) Kabir et al. 2023.

Background : DROPBEAR

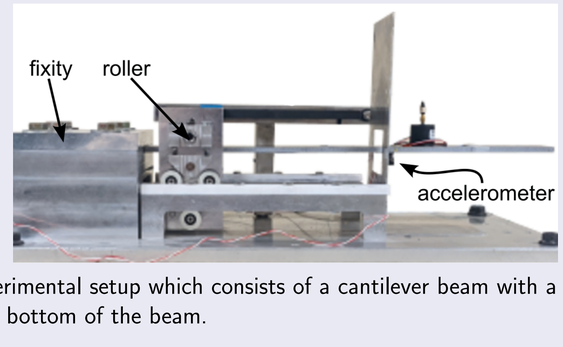

Figure 5. The DROPEAR experimental setup which consists of a cantilever beam with a movable roller and an accelerometer mounted on the bottom of the beam.

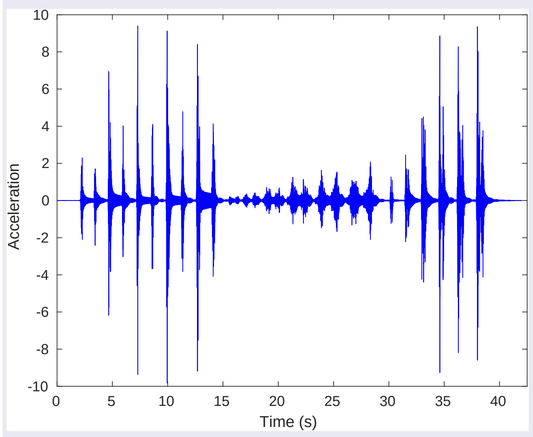

Figure 6. DROPEAR acceleration data, which results from the roller movements but is treated as an input into a model that predicts the roller location given the acceleration signal.

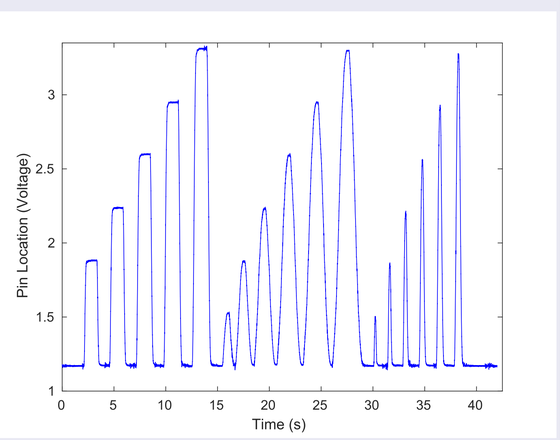

Figure 7. DROPEAR roller position, which moves to simulate a moving boundary condition for the cantilever beam. Essentially, it sets the root of the cantilever beam.

Automated Model Deployment

Accuracy	Multiplies	# LUTs	# DSPs	Latency (µs)	Optimized RF
0.169	11.9K	18999	10	168.83	48, 768, 384, 768, 384, 64
0.1433	12.2K	24808	17	169.14	48, 384, 384, 384, 768, 64, 16, 16, 16, 4
0.1339	12.3K	24807	17	169.14	48, 768, 768, 384, 768, 64, 25, 25, 25, 5
0.119	12.6K	24807	17	169.14	48, 384, 768, 384, 768, 512, 32, 32, 32, 4
0.1161	13.7K	26375	16	171.82	48, 768, 768, 768, 768, 384, 162, 162, 18
0.1134	15.7K	26375	16	171.82	48, 768, 768, 768, 768, 384, 162, 162, 18
0.1095	16.8K	27125	14	171.82	60, 600, 1200, 300, 1200, 1360, 289, 289, 17
0.1065	21.7K	63052	40	193.92	78, 2028, 1014, 2028, 2028, 1768, 289, 289, 17
0.1029	25.0K	63052	40	193.92	90, 2700, 2700, 2700, 2700, 2040, 289, 289, 17
0.0982	25.6K	30836	24	170.59	24, 192, 384, 768, 384, 1824, 1444, 38
0.0958	33.0K	44702	30	176.81	24, 192, 384, 384, 768, 4512, 2209, 2209, 2209, 2209, 47
0.0939	34.4K	63052	40	194.94	123, 5043, 5043, 5043, 5043, 3116, 361, 361, 19
0.0851	36.6K	80227	58	174.88	24, 192, 768, 768, 384, 5600, 2500, 2500, 2500, 50
0.0828	41.4K	91708	66	176.96	24, 192, 768, 768, 768, 336, 2916, 2916, 2916, 2916, 54
0.0813	70.5K	91702	66	176.96	24, 192, 768, 768, 768, 13200, 5625, 5625, 5625, 5625, 75
0.0792	74.9K	94960	78	193.26	24, 192, 192, 192, 768, 14592, 5776, 5776, 5776, 5776, 76

Model Deployment Optimizer

Stochastic Search			Proposed ILP Search		ILP vs Stochastic	
Trials	Search Time (s)	Design Latency (μ s)	Search Time (s)	Design Latency (μ s)	Search Speedup	Latency Speedup
1K	5.03	343.06	4.8	189.84	1.05	1.81
10K	47.67	233.82			9.93	1.23
100K	490.68	227.95			102.23	1.20
1M	4965.65	204.768			1034.51	1.08

Table 1. HLS4ML Deployment Optimizer Versus Stochastic Search

References

Kabir, E., D. Coble, J. N. Satme, A. R. Downey, J. D. Bakos, D. Andrews, and M. Huang (2023). "Accelerating LSTM-based High-Rate Dynamic System Models". In: *2023 33rd International Conference on Field-Programmable Logic and Applications (FPL)*. IEEE, pp. 327–332.

Satme, J., D. Coble, B. Priddy, A. R. Downey, J. D. Bakos, and G. Comert (2022). "Progress Towards Data-Driven High-Rate Structural State Estimation on Edge Computing Devices". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 86311. American Society of Mechanical Engineers, V010T10A017.

 contact:ss121@email.sc.edu