AI4Ax: Learning to Design Application-Aware Efficient Approximate Circuits

***Abstract*—AI for Approximate computing (AI4Ax) is the first demonstration of an AI-based approach for comprehensive design space exploration in *approximate computing*. Approximate multiplier for neural network applications is taken as a case study, and we adopt an 8-bit Baugh–Wooley multiplier with multiple reconfigurable compressors, each of which can be replaced by approximate compressors. We then leverage NSGA-II, PPO, and MOEA/D to preserve near-exact accuracy while reducing power, latency, and power-delay product (PDP) of the neural network workloads such as LeNet-5 and a multi-layer perceptron. Our exploration shows that the best solutions employ a mix of approximate compressors and bring down the total power and latency to just $\sim 1\%$, demonstrating the efficacy of the approach for efficient approximate circuit design.**

***Index Terms*—Approximate Computing, Design Space Exploration, Machine Learning**

I. Introduction

The rapid growth of AI-driven applications in edge devices has amplified the need for balancing power efficiency and computational accuracy. As edge devices face stringent resource constraints, traditional fixed-precision arithmetic in neural network (NN) inference struggles to keep pace with increasing model complexity. Approximate computing, which exploits neural networks' inherent error resilience, offers a way to significantly reduce power and latency by introducing controlled approximations in arithmetic operations [1]. However, many existing approximation strategies are either coarse-grained or designed with identical approximate circuits being replicated in a fine-grain setting [2]. Exhaustive exploration of fine-grain mixing of different approximate circuits can become prohibitively large and cumbersome.

AI4Ax framework fills this gap by intelligently exploring the options to give best possible design(s). As a case study, we show the design of an 8-bit approximate Baugh-Wooley multipler for MLP (multi-layer perceptron) and LENET-5 neural network applications. In our approach, accurate compressors in specific columns can be replaced with approximate variants, including AC-series designs. Using NSGA-II, reinforcement learning (PPO), and MOEA/D, we systematically explore power–accuracy, latency–accuracy, and power-delay product (PDP) trade-offs on LeNet-5. This multi-objective optimization yields a Pareto front, highlighting designs that best balance hardware constraints and inference performance, and demonstrating the effectiveness of fine-grained, data-driven approximations for neural network workloads.

II. Methodology

Figure 1 provides a high-level overview of our methodology, which uses a reconfigurable 8-bit Baugh–Wooley multiplier

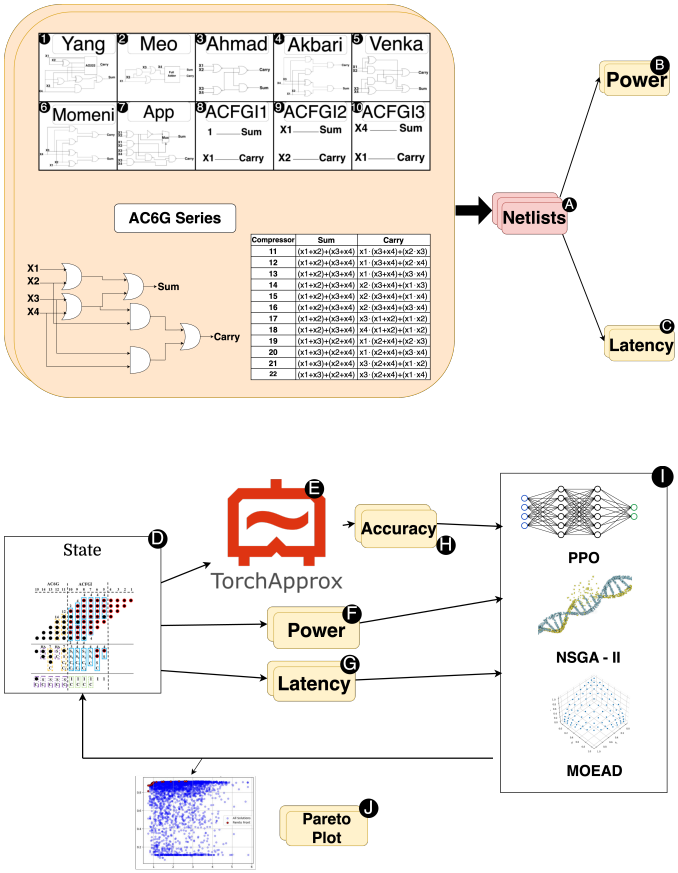

Fig. 1: Overview of the proposed AI4Ax Framework (uses approximate multiplier [4] [5]).

architecture containing 20 columns, each of which can be configured with one of 22 compressor variants. We begin by implementing every compressor in Cadence Virtuoso (A) to extract individual power (B) and latency (C) metrics. These hardware-level data points are then combined according to a configuration vector to estimate the total power and latency for the multiplier. Accuracy is evaluated by substituting the selected compressors in two neural network workloads (LeNet-5 CNN and a multi-layer perceptron) using TorchApprox [3] (E). Finally, we employ NSGA-II, PPO, and MOEA/D to search this large design space to obtain Pareto-optimal trade-offs, enabling efficient balancing of power, latency, and inference accuracy.

III. Results

As shown in Figure 3, the AC6G series (e.g., (11) to (12)) dominates the Most Significant Bit (MSB) columns, where

its slightly higher power and latency overhead is tolerated for improved accuracy. Other high-accuracy designs (e.g., Meo, Venka, and Yang) also appear in the MSB, reflecting a preference for compressors that minimize significant error propagation. By contrast, simpler wire-based compressors like ACFGI-1 to ACFGI-3, offering very low overhead, are placed near the Least Significant Bit (LSB), where errors have a reduced impact on the final result. After generating a random configuration, if AC6G or ACFGI compressors are initially placed in the LSB, the algorithms generally leave them unchanged due to negligible accuracy gains. This strategic placement underscores a critical design principle: balancing accuracy and overhead hinges on aligning compressor capabilities with the error tolerance of each bit position.

In Tables I, all three algorithms converge to solutions with comparable performance improvements, albeit via different compressor selections. The multi-compressor approach consistently outperforms the single-compressor strategy (Table II), confirming that optimal designs require a mix of approximate compressors at different bit positions. Some compressors, e.g., app compressor, Ahmad, Akbari, Meo, are rarely chosen in practice due to relatively high power and latency overhead compared to the more balanced AC6G or ACFGI families. The consistent exclusion of these compressors highlights the necessity of evaluating both error profiles and hardware metrics during early design stages. These insights provide an actionable framework for deploying approximate computing in real-world systems, where targeted precision and resource constraints are paramount.

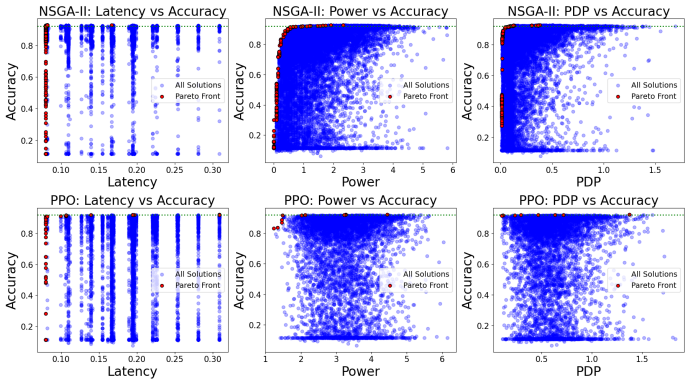

Fig. 2: Pareto Plots of Multi-Objective Optimization for Approximate Multipliers

TABLE I: Comparison of Power and Latency for CNN and MLP for Best Design and Exact Multiplier

Accuracy is reported as actual percentage of best design and power, latency, and PDP as x times improvement from exact multiplier (Exact/Best)(Green is for CNN and Orange is for MLP)*

Algorithm	Accuracy (%)	Power	Latency	PDP
NSGA-II (CNN)	90.93	85.60	51.26	4388.01
NSGA-II (MLP)	91.97	203.47	104.90	21342.94
MOEA/D (CNN)	90.67	134.18	52.58	7055.15
MOEA/D (MLP)	93.77	135.89	52.58	7145.12
PPO (CNN)	90.96	41.89	52.58	2202.72
PPO (MLP)	93.28	77.49	52.58	4074.44

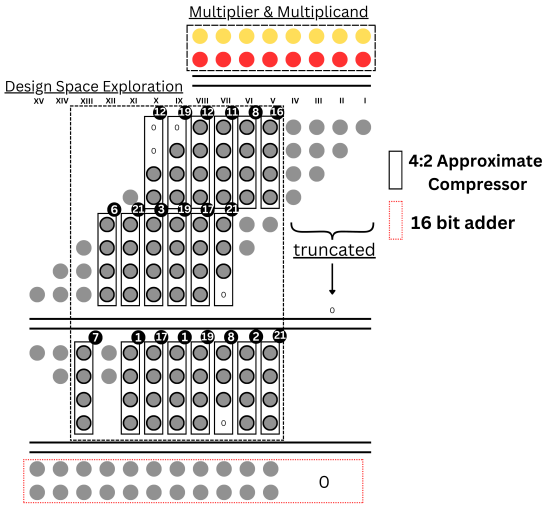

Fig. 3: Optimized design after convergence in design space exploration, with selected approximate compressors. *Numbers in circles indicate compressor indices from fig 1*

TABLE II: Comparison of Power and Latency using same compressor at all positions with best design

Accuracy is reported as actual percentage and power, latency, and PDP as x times improvement (Same/Best)

Compressor	Accuracy (%)	Power	Latency	PDP
❶	91.43	210.32	1.43	300.24
❷	92.26	205.31	3.31	680.37
❸	91.23	235.35	2.14	503.02
❻	92.18	295.44	2.72	802.69
❼	90.29	45.70	0.86	39.18

IV. Conclusion

This work demonstrates that a machine learning–based exploration framework can identify approximate multipliers that significantly reduce power, latency, and power–delay product while maintaining near-exact accuracy. By tailoring compressor choices across bit positions, the methodology finds Pareto-optimal solutions for neural network workloads, highlighting the efficacy of data-driven approaches in navigating intricate hardware design trade-offs.

References

[1] F. Guella, E. Valpreda, M. Caon, G. Masera, and M. Martina, "Marlin: A co-design methodology for approximate reconfigurable inference of neural networks at the edge," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2024.

[2] L. Sayadi, S. Timarchi, and A. Sheikh-Akbari, "Two efficient approximate unsigned multipliers by developing new configuration for approximate 4:2 compressors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 4, pp. 1649–1659, 2023.

[3] E. Trommer, B. Waschneck, and A. Kumar, "High-throughput approximate multiplication models in pytorch," in *26th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, 2023, pp. 79–82.

[4] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "Comparison and extension of approximate 4-2 compressors for low-power approximate multipliers," vol. 34, no. 1, pp. 149–162, 2015.

[5] Y. Wang, Z. Wang, S. Yin, L. Liu, and S. Wei, "Fast and high-accuracy approximate mac unit design for CNN computing," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 1, pp. 67–71, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9657057