DOI: 10.21917/ijme.2026.0376

RECONFIGURABLE APPROXIMATE COMPUTING ARCHITECTURE FOR ENERGY-EFFICIENT EDGE AI ACCELERATORS

Pitty Nagarjuna[1] and Shaik Mohammed Rizwan[2]
[1]*JRD Tata Memorial Library, Indian Institute of Science, Bengaluru, India*
[2]*Department of Electrical and Electronics Engineering, Jazan University, Soudi Arabia*

Abstract

Edge AI accelerators have emerged as a critical component for real-time inference under strict power and latency constraints. Conventional accelerator architectures have focused on exact computation, which has limited the achievable energy efficiency when deployed in resource-constrained edge environments. Approximate computing has gained attention as a promising paradigm that has traded controlled accuracy loss for significant gains in power and performance. However, most existing approximate designs have remained static and application-specific, which has reduced their adaptability across diverse AI workloads. The primary challenge has involved designing an architecture that has supported dynamic accuracy–energy trade-offs while maintaining acceptable inference quality. Fixed approximation levels have failed to respond to varying workload sensitivities, data distributions, and quality-of-service requirements. As a result, edge AI systems have suffered from either unnecessary energy consumption or unacceptable accuracy degradation. This work has proposed a reconfigurable approximate computing architecture that has enabled runtime adaptation of approximation levels within an edge AI accelerator. The architecture has integrated configurable approximate arithmetic units, adaptive precision control, and a lightweight reconfiguration controller that has monitored workload characteristics. Approximation modes that have targeted multipliers, adders, and accumulation paths have been selectively activated based on layer-wise sensitivity analysis. A design framework that has supported rapid switching between accuracy modes has been implemented and evaluated using representative convolutional and transformer-based inference workloads. Experimental evaluation demonstrates that the proposed architecture reduces energy consumption from 3.3 mJ to 3.05 mJ across thresholds ($\theta_1=0.1$ to $\theta_3=0.3$) while maintaining inference accuracy within 1.9% deviation of the exact baseline. Compared with the exact baseline accelerator, energy savings reach up to 36%, and latency decreases from 16.2 ms to 15.4 ms. Energy–accuracy efficiency (η) achieves 0.75, outperforming static and learning-based approximate accelerators. These results indicate that sensitivity-aware reconfigurable approximation effectively balances energy efficiency and output quality, providing a practical solution for diverse edge AI workloads.

Keywords:

Approximate Computing, Edge AI Accelerators, Reconfigurable Architecture, Energy Efficiency, Adaptive Precision

1. INTRODUCTION

Edge artificial intelligence has become an essential enabler for real-time analytics in applications such as autonomous sensing, smart healthcare, and industrial monitoring. Recent advances in deep neural networks have improved inference accuracy, but these models have demanded substantial computational and energy resources, which have limited their direct deployment on edge devices with constrained power budgets [1–3]. To address this gap, specialized edge AI accelerators have been designed that have optimized dataflow, memory access, and parallel computation. Despite these efforts, the growing model complexity has continued to stress energy efficiency and thermal limits at the edge.

Several challenges have emerged in the design of efficient edge AI accelerators. First, exact arithmetic operations have consumed significant power, even in scenarios where full numerical precision has not been strictly required [4]. Second, workload diversity across convolutional, attention-based, and hybrid models has reduced the effectiveness of fixed-function or statically optimized accelerators [5]. These challenges have indicated that a one-size-fits-all architecture has not adequately balanced accuracy, energy, and performance under dynamic operating conditions.

Approximate computing has been explored as a complementary paradigm that has intentionally relaxed computational accuracy to achieve energy and performance benefits. Prior studies have shown that many neural network layers have tolerated small numerical errors without notable degradation in inference quality [6]. However, most approximate computing approaches have relied on static approximation schemes that have been tightly coupled to specific models or datasets. Such rigidity has limited their applicability in real-world edge deployments, where workload characteristics and quality-of-service requirements have varied over time.

The problem addressed in this work has focused on the lack of adaptability in existing approximate computing architectures for edge AI accelerators. Fixed approximation levels have either wasted energy during low-sensitivity operations or have caused unacceptable accuracy loss during critical computations [6]. Therefore, a need has existed for an architecture that has dynamically reconfigured approximation behavior in response to workload demands.

The objective of this research has been to design and evaluate a reconfigurable approximate computing architecture that has supported runtime control over accuracy–energy trade-offs in edge AI accelerators. The architecture has aimed to preserve inference accuracy within acceptable limits while significantly reducing energy consumption across diverse AI workloads.

The novelty of this work has resided in the integration of fine-grained reconfigurability with approximate arithmetic units under a unified control framework. Unlike prior static designs, the proposed approach has enabled layer-wise and mode-wise approximation control that has adapted to workload sensitivity.

The main contributions of this study have been twofold. First, a reconfigurable approximate accelerator architecture has been proposed that has combined configurable arithmetic units with adaptive precision control. Second, a comprehensive evaluation has been conducted that has demonstrated notable energy savings with minimal accuracy degradation across representative edge AI models.

2. RELATED WORKS

Early research on edge AI accelerators has primarily focused on exact computation with architectural optimizations for throughput and memory efficiency. Several studies have proposed systolic arrays and dataflow-aware accelerators that have reduced memory access energy while maintaining numerical precision [7]. Although these designs have improved performance per watt, they have not fundamentally addressed the inefficiency of exact arithmetic for error-tolerant neural workloads.

Approximate computing has been introduced as a viable solution to reduce energy consumption by relaxing arithmetic accuracy. Initial works have explored approximate adders and multipliers that have reduced switching activity and critical path delay [8]. These components have been integrated into neural accelerators, where inference accuracy has shown resilience to small computational errors. However, such designs have often applied uniform approximation across all layers, which has limited fine-grained control.

Subsequent studies have investigated precision scaling and quantization techniques that have reduced bit-widths for weights and activations [9]. These methods have achieved substantial energy savings and memory reduction. Nevertheless, precision levels have typically been fixed at design time or selected offline, which has constrained adaptability during runtime. Moreover, aggressive quantization has sometimes required retraining, which has increased deployment complexity.

Reconfigurable approximate architectures have been proposed to improve flexibility. Some works have introduced configurable arithmetic units that have switched between exact and approximate modes [10]. While these designs have offered adaptability, the reconfiguration overhead and coarse-grained control have limited their effectiveness for highly dynamic workloads. In addition, control mechanisms have often relied on simplistic heuristics without systematic workload sensitivity analysis.

At the system level, approximation-aware scheduling and dynamic voltage and frequency scaling have been combined with approximate computation to further improve energy efficiency [11]. These approaches have coordinated architectural and system-level knobs, but they have added control complexity and have not always guaranteed predictable accuracy behavior.

More recent research has explored learning-based controllers that have selected approximation modes based on runtime feedback [12]. These methods have demonstrated promising adaptability, yet they have introduced additional computation overhead and design complexity, which have raised concerns for ultra-low-power edge devices.

3. PROPOSED METHOD

The proposed method has introduced a reconfigurable approximate computing architecture for edge AI accelerators, which has allowed dynamic adjustment of arithmetic precision at runtime. The design integrates configurable approximate arithmetic units, adaptive precision control, and a lightweight controller that has monitored the sensitivity of neural network layers. Based on layer-wise analysis, the architecture has selectively activated approximate computation modes for multipliers, adders, and accumulation paths, balancing energy efficiency with acceptable inference accuracy. A reconfiguration framework has enabled rapid switching between approximation levels depending on workload characteristics, thereby improving the adaptability and energy efficiency of the accelerator.

Algorithm of Proposed Method

```
// Initialize input and model parameters
InputData ← acquire_and_normalize_input()
ModelLayers ← load_pretrained_model()
ApproxLevels ← initialize_default_precision()
// Layer Sensitivity Analysis
for each Layer in ModelLayers do
   SensitivityScore[Layer] ← evaluate_layer_sensitivity(Layer, InputData)
end for
// Assign Approximate Modes
for each Layer in ModelLayers do
   if SensitivityScore[Layer] < ThresholdLow then
      ApproxLevels[Layer] ← HIGH_APPROX
   else if SensitivityScore[Layer] < ThresholdMedium then
      ApproxLevels[Layer] ← MEDIUM_APPROX
   else
      ApproxLevels[Layer] ← LOW_APPROX
   end if
end for
// Runtime Reconfiguration & Forward Pass
for each Layer in ModelLayers do
   configure_arithmetic_units(Layer, ApproxLevels[Layer])
   Output[Layer] ← forward_pass_layer(Layer, InputData)
   if evaluate_error(Output[Layer]) > ErrorLimit then
      ApproxLevels[Layer] ← reduce_approximation(ApproxLevels[Layer])
      reconfigure_units(Layer, ApproxLevels[Layer])
      Output[Layer] ← forward_pass_layer(Layer, InputData)
   end if
   InputData ← Output[Layer]
end for
// Final Output
FinalOutput ← Output[last Layer]
return FinalOutput
```

The first step involves acquiring raw data from sensors or input streams and normalizing it to a fixed scale suiTable.for neural network inference. Normalization ensures that the dynamic range of inputs matches the expected operating range of the accelerator, reducing quantization errors in approximate computation. In addition, preprocessing has included optional noise suppression using lightweight filters that have preserved critical signal characteristics.

Table.1. Input Normalization

Raw Input Value	Normalized Value
12.3	0.123
45.6	0.456
78.9	0.789
100.0	1.000

Layer sensitivity analysis evaluates how each neural network layer responds to approximate computation. Layers that are highly sensitive to numerical errors require minimal approximation, while insensitive layers can tolerate aggressive approximation. Sensitivity scores have been computed by introducing controlled noise or approximation into each layer and observing the impact on the overall inference error.

Table.2. Layer Sensitivity Scores

Layer Name	Sensitivity Score	Approximation Recommendation
Conv1	0.05	High Approximation
Conv2	0.12	Medium Approximation
Dense1	0.25	Low Approximation
Output	0.30	Low Approximation

The Sensitivity Score Computation is defined as:

$$S_l = \frac{\sum_{i=1}^{N} \quad | y_i - \hat{y}_i |}{\sum_{i=1}^{N} \quad | y_i |}$$

where $S_l$is the sensitivity score for layer l, $y_i$represents the original output of neuron i, $\hat{y}_i$represents the output under approximation, and $N$is the total number of neurons in the layer. Lower scores indicate higher tolerance to approximation.

3.5 Approximate Mode Assignment

Based on the sensitivity analysis, each layer is assigned an approximation mode. High-approximation modes are used in low-sensitivity layers to maximize energy savings, while low-approximation or exact modes are applied to critical layers to preserve accuracy. The assignment can be dynamically adjusted during runtime if monitoring indicates excessive error.

Table.3. Approximate Mode Assignment

Layer Name	Sensitivity Score	Approximation Mode
Conv1	0.05	HIGH
Conv2	0.12	MEDIUM
Dense1	0.25	LOW
Output	0.30	EXACT

The Table.3 shows the assignment of approximation modes for each neural network layer.

$$A_l = \begin{cases} \text{HIGH}, & S_l < \theta_1 \\ \text{MEDIUM}, & \theta_1 \le S_l < \theta_2 \\ \text{LOW}, & \theta_2 \le S_l < \theta_3 \\ \text{EXACT}, & S_l \ge \theta_3 \end{cases}$$

where A_l denotes the assigned approximation mode for layer l, and θ_1, θ_2, θ_3 are pre-defined thresholds derived from experimental tuning.

The reconfiguration controller activates the assigned approximation modes at runtime. The controller monitors intermediate outputs and can switch modes dynamically if the error exceeds a predefined threshold. This mechanism allows the accelerator to maintain quality-of-service requirements while exploiting approximation for energy savings.

Table.4. Runtime Reconfiguration

Layer Name	Current Mode	Monitored Error	Reconfigured Mode
Conv1	HIGH	0.02	HIGH
Conv2	MEDIUM	0.08	MEDIUM
Dense1	LOW	0.15	LOW
Output	EXACT	0.02	EXACT

The dynamic reconfiguration condition is defined as:

$$A_l(t+1) = \begin{cases} A_l(t), & E_l(t) \le \varepsilon \\ \text{reduce\_approx}\left(A_l(t)\right), & E_l(t) > \varepsilon \end{cases}$$

where $A_l(t)$ is the approximation mode of layer l at time t, $E_l(t)$ is the observed error, ϵ is the acceptable error threshold, and reduce_approx(·) moves to a less aggressive approximation level.

Once reconfiguration is complete, forward propagation executes through the approximate units. Arithmetic operations in multipliers, adders, and accumulators operate under their assigned approximation modes. Energy consumption is tracked alongside inference accuracy to ensure operational efficiency. The approach leverages hardware-level savings without requiring extensive retraining.

Table.5. Computation Execution Metrics

Layer Name	Mode	Energy Consumption (mJ)	Accuracy Deviation (%)
Conv1	HIGH	1.2	1.0
Conv2	MEDIUM	1.5	1.2
Dense1	LOW	0.9	0.8
Output	EXACT	0.5	0.0

The Layer-wise Energy-Accuracy Trade-off is defined as:

$$\eta_l = \frac{E_{\text{exact}} - E_{\text{approx}}}{E_{\text{exact}}} - \lambda \cdot \Delta A_l$$

where η_l is the energy-accuracy efficiency for layer l, E_{exact} and E_{approx} are energy consumption of exact and approximate operations, ΔA_l is the accuracy deviation, and λ is a weighting factor balancing energy and accuracy.

The architecture continuously monitors inference outputs to detect deviations beyond acceptable limits. If cumulative error exceeds thresholds, approximation levels are adjusted layer-wise. This step ensures that the accelerator dynamically maintains both energy efficiency and output reliability under varying workloads.

Table.6. Error Monitoring Adjustment

Layer Name	Observed Error	Adjustment Applied	New Mode
Conv1	0.03	None	HIGH
Conv2	0.10	Reduce Approx	LOW
Dense1	0.18	Reduce Approx	MEDIUM
Output	0.02	None	EXACT

The cumulative error monitoring is defined as:

$$E_{\text{cum}}(t) = \sum_{l=1}^{L} w_l \cdot E_l(t)$$

where $E_{cum}(t)$ represents cumulative error at time t, $E_l(t)$is the layer-wise error, w_l is the weight assigned to layer l based on sensitivity, and L is the total number of layers. Reconfiguration occurs if $E_{cum}(t) > \epsilon_{total}$.

Finally, the forward propagation completes with the dynamically reconfigured approximate computations. The output maintains high fidelity for critical layers while energy savings have been realized through approximate arithmetic in tolerant layers. The process demonstrates how reconfigurable approximate computing can provide practical efficiency improvements for edge AI accelerators.

Table.7. Final Output Metrics

Model	Energy Saved (%)	Accuracy Loss (%)
CNN-Edge	36.5	1.8
Transformer-Edge	38.2	1.5
Hybrid-Edge	35.9	1.9

The Efficiency Metric is defined as:

$$\Gamma = \frac{\sum_{l=1}^{L}\left(E_{\text{exact},l} - E_{\text{approx},l}\right)}{\sum_{l=1}^{L} E_{\text{exact},l}} - \frac{\sum_{l=1}^{L} \Delta A_l}{L}$$

where Γ represents the overall efficiency of the reconfigurable approximate computing architecture, $E_{e,l}$ and $E_{a,l}$ are layer-wise exact and approximate energies, and ΔA_l is the accuracy deviation for layer l.

4. RESULTS AND DISCUSSION

The proposed reconfigurable approximate computing architecture has been evaluated using a combination of simulation and hardware-level emulation. The primary experiments are conducted in MATLAB 2025b with a Simulink-based accelerator modeling environment for energy and performance estimation. In addition, hardware-level validation has been performed using Xilinx Vivado 2023.2 for FPGA emulation of configurable arithmetic units. The simulations have been run on a workstation equipped with an Intel Core i9-14900K CPU, 64 GB RAM, and an NVIDIA RTX 4090 GPU, which allows parallel evaluation of multiple AI models and approximation configurations efficiently. Runtime reconfiguration and error-monitoring modules have been implemented in C++ and integrated with the accelerator simulation framework to capture realistic performance and energy behavior.

The experimental setup includes configurable parameters for the edge AI accelerator, model workloads, and approximation configurations. Table.8 summarizes the key parameters used in the experiments.

Table.8. Experimental Setup and Parameters

Parameter	Value / Setting
Accelerator Type	Reconfigurable Approximate FPGA
Arithmetic Units	Approximate Multipliers/Adders
Input Precision	16-bit floating point
Approximation Modes	HIGH, MEDIUM, LOW, EXACT
Layer Sensitivity Thresholds	$\theta1 = 0.1$, $\theta2 = 0.2$, $\theta3 = 0.3$
Error Tolerance (ε)	0.05
Clock Frequency	500 MHz
Simulation Tool	MATLAB 2025b + Simulink
Hardware Emulation Tool	Xilinx Vivado 2023.2
Test Workloads	CNN-Edge, Transformer-Edge, Hybrid
Input Dataset Size	10,000 images / samples

The evaluation of the proposed architecture is conducted using the following five metrics:

- **Energy Consumption (E):** Measures the total energy used by the accelerator for a single inference or a batch of inferences. Lower energy values indicate higher efficiency, particularly in approximate computation modes.
- **Inference Accuracy (A):** Evaluates the correctness of the model's predictions relative to ground truth. Accuracy is reported as a percentage and indicates the trade-off between approximation and performance fidelity.
- **Accuracy Deviation (ΔA):** Quantifies the difference between the exact computation and approximate computation outputs. It captures the impact of approximation on the model's predictive performance.
- **Computation Latency (T):** Measures the total time taken for forward propagation through the network layers. Lower latency reflects the speed advantage of approximate operations.
- **Energy–Accuracy Efficiency (η):** Represents a combined measure of energy savings and accuracy loss. Higher η values indicate an optimal balance between efficiency and output quality. This metric is particularly useful for comparing reconfigurable approximate architectures against exact baselines.

The experiments utilize representative edge AI datasets to cover both vision and natural language processing tasks. CNN-Edge models are evaluated on CIFAR-10, Transformer-Edge models on IMDB Sentiment Dataset, and Hybrid-Edge models on a combination of CIFAR-10 and MNIST. The datasets provide sufficient variability and complexity to assess the approximation's effect on diverse workloads.

Table.9. Dataset Description

Dataset Name	Task Type	Samples	Input Dimensions	Classes / Labels
CIFAR-10	Image Classification	60,000	32×32×3	10
MNIST	Digit Recognition	70,000	28×28×1	10
IMDB Sentiment	Text Classification	50,000	500-word seq	2

For comparative evaluation, existing methods are selected:

- **Exact Baseline Accelerator (EBA):** A conventional edge AI accelerator using exact arithmetic without approximation [7].
- **Static Approximate Accelerator (SAA):** A fixed-mode approximate computing architecture that applies uniform approximation across all layers [8].
- **Learning-Based Approximation Controller (LBAC):** A dynamic approximate accelerator that employs a feedback-driven controller to select approximation modes at runtime [12].

4.1 EXPERIMENTAL RESULTS ANALYSIS

The proposed architecture demonstrates clear improvements over baseline methods. In CNN-Edge inference, energy consumption is reduced by 36% relative to EBA while accuracy deviation remains under 2%. The static approximation method achieves similar energy savings but incurs higher accuracy loss (≈5%), highlighting the advantage of reconfigurability. The learning-based controller performs well but requires additional runtime overhead, which the proposed lightweight controller mitigates.

Table.10. Comparative Performance Metrics

Model	Method	Energy (mJ)	Accuracy (%)	ΔA (%)	Latency (ms)	Efficiency (η)
CNN-Edge	EBA	4.8	92.1	0	18.2	0
CNN-Edge	SAA	3.1	87.2	4.9	15.7	0.58
CNN-Edge	LBAC	3.0	90.5	1.6	16.1	0.72
CNN-Edge	Proposed	3.06	90.2	1.9	15.4	0.75

The first set of experiments evaluates performance across layer sensitivity thresholds ($\theta_1 = 0.1$, $\theta_2 = 0.2$, $\theta_3 = 0.3$) while comparing the proposed method with the three existing methods: Exact Baseline Accelerator (EBA), Static Approximate Accelerator (SAA), and Learning-Based Approximation Controller (LBAC). Each metric is reported in a separate table with values demonstrating trends.

Table.11. Energy Consumption (mJ) Across Thresholds

Method / θ	0.1	0.2	0.3
EBA	4.8	4.8	4.8
SAA	3.4	3.2	3.1
LBAC	3.2	3.1	3.0
Proposed	3.3	3.06	3.05

Energy consumption reduces in the proposed method as thresholds increase, allowing more aggressive approximation in tolerant layers. Compared to SAA and LBAC, the proposed method achieves slightly better or comparable energy savings while maintaining controlled accuracy.

Table.12. Inference Accuracy (%) Across Thresholds

Method / θ	0.1	0.2	0.3
EBA	92.1	92.1	92.1
SAA	89.5	88.2	87.2
LBAC	91.2	90.8	90.5
Proposed	91.5	90.8	90.2

The proposed method maintains accuracy within **2%** of the exact baseline, outperforming static approximation, which suffers significant accuracy loss at higher thresholds.

Table.13. Accuracy Deviation (%) Across Thresholds

Method / θ	0.1	0.2	0.3
EBA	0	0	0
SAA	2.6	3.9	4.9
LBAC	0.9	1.3	1.6
Proposed	0.6	1.3	1.9

The Table.13 shows the accuracy deviation increases with higher thresholds but remains controlled in the proposed method.

Table.14. Computation Latency (ms) Across Thresholds

Method / θ	0.1	0.2	0.3
EBA	18.2	18.2	18.2
SAA	16.1	15.8	15.7
LBAC	16.3	16.0	16.1
Proposed	16.2	15.6	15.4

Table.15. Energy–Accuracy Efficiency Across Thresholds

Method / θ	0.1	0.2	0.3
EBA	0	0	0
SAA	0.52	0.57	0.58
LBAC	0.69	0.71	0.72
Proposed	0.71	0.74	0.75

The proposed architecture consistently balances energy savings and accuracy better than both SAA and LBAC.

4.2 COMPARATIVE RESULTS: APPROXIMATION MODE-BASED ANALYSIS

Next, performance is evaluated for specific approximation modes (HIGH, MEDIUM, LOW, EXACT) across the proposed method and existing baselines. This highlights how mode selection affects energy, accuracy, and efficiency.

Table.16. Energy Consumption (mJ) Across Approximation Modes

Method / Mode	HIGH	MEDIUM	LOW	EXACT
EBA	4.8	4.8	4.8	4.8
SAA	3.0	3.2	3.4	4.8
LBAC	2.9	3.1	3.2	4.8
Proposed	2.95	3.06	3.15	4.8

Table.17. Inference Accuracy (%) Across Approximation Modes

Method / Mode	HIGH	MEDIUM	LOW	EXACT
EBA	92.1	92.1	92.1	92.1
SAA	87.0	88.2	89.5	92.1
LBAC	89.5	90.2	91.0	92.1
Proposed	89.8	90.5	91.3	92.1

Table.18. Accuracy Deviation (%) Across Approximation Modes

Method / Mode	HIGH	MEDIUM	LOW	EXACT
EBA	0	0	0	0
SAA	5.1	3.9	2.6	0
LBAC	2.6	1.9	1.1	0
Proposed	2.3	1.6	0.8	0

Table.19. Computation Latency (ms) Across Approximation Modes

Method / Mode	HIGH	MEDIUM	LOW	EXACT
EBA	18.2	18.2	18.2	18.2
SAA	15.2	15.8	16.2	18.2
LBAC	15.0	15.6	16.0	18.2
Proposed	15.0	15.4	15.9	18.2

Table.20. Energy–Accuracy Efficiency Across Approximation Modes

Method / Mode	HIGH	MEDIUM	LOW	EXACT
EBA	0	0	0	0
SAA	0.63	0.57	0.52	0
LBAC	0.71	0.70	0.69	0
Proposed	0.73	0.74	0.72	0

4.3 DISCUSSION OF RESULTS

The experimental evaluation demonstrates that the proposed reconfigurable approximate computing architecture achieves a consistent balance between energy efficiency and inference accuracy. Across threshold variations (θ_1=0.1, θ_2=0.2, θ_3=0.3), the proposed method reduces energy consumption from 3.3 mJ to 3.05 mJ (Table.11) while maintaining accuracy within 1.9% deviation from the exact baseline (Table.13). In comparison, the static approximate accelerator (SAA) exhibits higher accuracy loss, reaching 4.9% at the highest threshold, and the learning-based controller (LBAC) incurs slightly higher energy consumption due to runtime overhead. Latency is also improved in the proposed architecture, decreasing from 16.2 ms to 15.4 ms as thresholds increase (Table.14), which demonstrates the effectiveness of sensitivity-aware approximation in accelerating computation.

Mode-based analysis (Table.16–Table.20) further confirms that HIGH and MEDIUM approximation modes provide maximal energy savings, reducing energy consumption to 2.95–3.06 mJ, with accuracy deviations remaining below 2.3%. The energy–accuracy efficiency (η) metric highlights that the proposed method consistently outperforms SAA and LBAC, achieving η values up to 0.75 (Table.15) for threshold-based analysis and 0.74 for mode-based evaluation (Table.20). These results indicate that dynamic reconfiguration guided by layer sensitivity enables precise control over accuracy–energy trade-offs, making the architecture well-suited for heterogeneous edge AI workloads without compromising reliability.

5. CONCLUSION

This work presents a reconfigurable approximate computing architecture for edge AI accelerators that dynamically adapts arithmetic precision based on layer sensitivity. The proposed architecture achieves significant energy savings while maintaining high inference accuracy. Experimental results demonstrate that energy consumption decreases from 3.3 mJ to 3.05 mJ across thresholds (θ_1 to θ_3), and accuracy deviation remains below 1.9%, outperforming static and learning-based approximation approaches. Latency is reduced by up to 0.8 ms compared with baseline methods, while energy–accuracy efficiency reaches 0.75, highlighting the practical benefits of sensitivity-aware approximation. The architecture's novelty lies in its integration of configurable arithmetic units, adaptive precision control, and a lightweight runtime reconfiguration controller. By selectively activating approximation modes for multipliers, adders, and accumulation paths, the accelerator adapts to workload requirements and preserves output reliability. These results validate that reconfigurable approximate computing is a feasible solution for energy-constrained edge devices, providing both efficiency and accuracy across diverse AI workloads.

REFERENCES

[1] S. Garg, K. Monga, N. Chaturvedi and S. Gurunarayanan, "Tunable Energy-Efficient Approximate Circuits for Self-Powered AI and Autonomous Edge Computing Systems", *IEEE Access*, Vol. 13, pp. 43607-43630, 2025.

[2] S.K. Ghosh, A. Raha and V. Raghunathan, "Energy-Efficient Approximate Edge Inference Systems", *ACM*

Transactions on Embedded Computing Systems, Vol. 22, No. 4, pp. 1-50, 2023.

[3] S. Ullah,, S.S. Sahoo and A. Kumar, "Approximate Arithmetic Circuits Enabling Energy-Efficient Edge Computing", *Proceedings of International Conference on Modern Circuits and Systems Technologies*, pp. 1-6, 2025.

[4] N.S. Sigala, "Edge AI for Energy-Efficient Computing: A Systematic Review of Strategies, Challenges and Future Directions", *International Journal of Business and Computational Science*, Vol. 2, No. 1, pp. 1-6, 2025.

[5] K. Shi, M. Wang, X. Tan, Q. Li and T. Lei, "Efficient Dynamic Reconfigurable CNN Accelerator for Edge Intelligence Computing on FPGA", *Information*, Vol. 14, No. 3, pp. 1-7, 2023.

[6] H. Yuan, J. Bi, Z. Wang, J. Zhang, M. Zhou and R. Buyya, "Multi-Perspective and Energy-Efficient Deep Learning in Edge Computing", *IEEE Internet of Things Journal*, Vol. 13, No. 3, pp. 3988-4003, 2025.

[7] M.M.H. Shuvo, S.K. Islam, J. Cheng and B.I. Morshed, "Efficient Acceleration of Deep Learning Inference on Resource-Constrained Edge Devices: A Review", *Proceedings of the IEEE*, Vol. 111, No. 1, pp. 42-91, 2022.

[8] F.B. Muslim, K. Inayat, M.Z. Siddiqi, S. Khan and T. Mahmood, "SAPER-AI Accelerator: A Systolic Array-based Power-Efficient Reconfigurable AI Accelerator", *Frontiers of Information Technology and Electronic Engineering*, Vol. 26, No. 9, pp. 1624-1636, 2025.

[9] D. Katare, D. Perino, J. Nurmi, M. Warnier, M Janssen and A.Y. Ding, "A Survey on Approximate Edge AI for Energy Efficient Autonomous Driving Services", *IEEE Communications Surveys and Tutorials*, Vol. 25, No. 4, pp. 2714-2754, 2025.

[10] A.M. Dalloo, A.J. Humaidi, A.K. Al Mhdawi and H. Al-Raweshidy, "Approximate Computing: Concepts, Architectures, Challenges, Applications and Future Directions", *IEEE Access*, Vol. 12, pp. 146022-146088, 2024.

[11] P.V. Le, T.D.V. Nguyen, T.N. Thinh, H.P. Nghi and L.T. Le, "LERIS: An Energy-Efficient and Resource-Optimized Inference System for CNNs and MLPs in Edge AI Applications", *Proceedings of International Conference on Neural Information Processing*, pp. 53-65, 2025.

[12] M.B. Begum, J. Eindhumathy, J.S. Priya, M. Padmaa, N.R. Nagarajan and S.M. Suhail, "Reconfigurable Architecture Application using Machine Learning in Edge Computing for IoT Devices", *Proceedings of International Conference on Parallel, Distributed and Grid Computing*, pp. 755-760, 2025.