Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Adaptive Energy-Aware Dynamic Structured Pruning for Battery-Constrained Edge AI Deployment

Ms. J. Ranganayaki
Assistant Professor
Department of Computer Science and Engineering
Bharath Institute of Science and Technology (BIST)
Chennai, India

Kaviyathamizhan T K
Department of Computer Science and Engineering, Bharath Institute of Science and Technology (BIST) Chennai, India

Keerthana S
Department of Computer Science and Engineering, Bharath Institute of Science and Technology (BIST) Chennai, India

Barath G
Department of Computer Science and Engineering, Bharath Institute of Science and Technology (BIST) Chennai, India

Abishek S
Department of Computer Science and Engineering, Bharath Institute of Science and Technology (BIST) Chennai, India

***Abstract*—Edge deployment of deep neural networks remains a fundamental challenge in applied machine learning, where resource-constrained platforms such as embedded controllers, unmanned aerial vehicles, and industrial IoT sensors demand real-time inference under strict power, memory, and latency budgets. Existing structured pruning methods physically remove entire convolutional filters yet apply a fixed compression rate uniformly across all layers — a strategy that over-compresses accuracy-critical layers while leaving computationally redundant ones underutilized. This paper presents AEDSP (Adaptive Energy-Aware Dynamic Structured Pruning), a framework that assigns layer-specific pruning ratios through a multi-objective score jointly incorporating measured energy consumption, inference latency, and empirical sensitivity profiled at three distinct compression levels per layer. A novel battery-aware urgency controller further adapts pruning aggressiveness at runtime using a continuous quadratic function — the first such mechanism introduced in structured pruning literature. Post-pruning accuracy is efficiently recovered through knowledge distillation from the uncompressed teacher model. Evaluated on CIFAR-10 across ResNet-18, MobileNetV2, and VGG-16, AEDSP achieves up to 50% FLOPs reduction, 54% model size compression, and 45% inference latency improvement while maintaining accuracy within 1.2% of the uncompressed baseline, consistently outperforming uniform pruning across all three architectures.**

***Index Terms*—structured pruning, edge AI, knowledge distillation, energy-aware inference, adaptive compression, neural network optimization, battery-aware systems**

I. Introduction

The rapid growth of deep neural networks has driven major advancements in computer vision, speech processing, and autonomous systems. At the same time, there is increasing demand to deploy these models on edge devices such as smartphones, drones, IoT sensors, and wearable systems. These devices must perform real-time inference under strict constraints on power, memory, and computation, creating a gap between model complexity and deployment feasibility.

Structured pruning has emerged as an effective solution, as it removes entire filters and preserves dense computation suitable for hardware acceleration. However, most existing methods apply a uniform pruning rate across all layers, which is not practical. Different layers vary in both computational cost and sensitivity to pruning—some can be aggressively compressed with minimal impact, while others are highly accuracy-critical. Uniform pruning therefore leads to inefficient compression, over-pruning sensitive layers and under-utilizing robust ones.

To address this, we propose **AEDSP (Adaptive Energy-Aware Dynamic Structured Pruning)**, which performs layer-wise adaptive pruning based on three factors: energy consumption, inference latency, and sensitivity to pruning. A composite score is computed for each layer to guide pruning decisions in a single pass, enabling efficient and targeted compression.

In addition, AEDSP introduces a battery-aware mechanism that adjusts pruning intensity based on device energy levels using a continuous function:

$$U(b) = 1 + (1 - b/100)^2 \tag{1}$$

This allows the model to adapt its efficiency dynamically without requiring direct hardware power measurements.

The key contributions of this work are:

- A multi-objective scoring method combining energy, latency, and sensitivity
- A three-level sensitivity evaluation strategy
- A continuous battery-aware pruning mechanism
- Knowledge distillation for accuracy recovery
- Validation across ResNet-18, MobileNetV2, and VGG-16

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

II. Related Work

A. Structured Pruning

Structured pruning removes entire filters or channels from convolutional networks, preserving dense tensor structures that execute efficiently on standard hardware. Early methods estimate filter importance using L1-norm magnitude, assuming filters with smaller weights contribute less to the output. Later approaches introduced data-driven metrics based on Taylor expansion and feature map statistics, providing a more accurate estimate of accuracy impact. However, most existing methods still apply a uniform pruning rate across all layers, ignoring differences in computational cost and sensitivity. In contrast, AEDSP assigns layer-specific pruning ratios using a multi-objective score, enabling more effective and balanced compression.

B. Hardware-Aware Compression

It is well known that FLOPs reduction does not always translate to real-world latency improvement on embedded hardware. This has led to hardware-aware pruning methods such as NetAdapt and AMC, which iteratively measure device latency during pruning. More recent approaches incorporate differentiable latency estimators for optimization. However, these methods often require repeated hardware profiling and do not explicitly consider energy consumption. AEDSP addresses this by performing a single profiling pass using CUDA timing and an energy proxy:

$$E_l = 0.001 \times FLOPs_l + 0.01 \times ActivationSize_{MB} \quad (2)$$

This captures both computation and memory costs while avoiding repeated hardware measurements.

C. Knowledge Distillation for Compression Recovery

Knowledge distillation improves post-pruning accuracy by transferring knowledge from a large teacher model to a smaller student model using soft targets. Compared to standard training, it provides richer supervision by encoding inter-class relationships, leading to better convergence and accuracy recovery. The effectiveness of distillation depends on parameters such as temperature (T) and weighting factor (α). AEDSP includes an ablation study to determine optimal values, selecting $T = 4.0$ and $\alpha = 0.7$ for best performance.

D. Battery-Adaptive Inference Systems

Adaptive inference methods aim to reduce computation under resource constraints using techniques such as early exiting and conditional execution. However, these approaches modify inference behavior rather than the model itself. Existing structured pruning methods also do not consider device energy level in compression decisions. AEDSP introduces a battery-aware mechanism that adjusts pruning ratios based on battery state using a continuous urgency function. This enables energy-adaptive compression and provides a practical framework for deployment under varying resource conditions.

III. Proposed Methodology

AEDSP is structured as an eight-module sequential pipeline that transforms a baseline-trained model into a deployment-ready compressed network. Two novel extensions augment the core pipeline: a runtime battery-aware urgency controller and cross-architecture validation.

A. Baseline Training Engine

The pipeline begins by training a full-precision baseline model on the target dataset to convergence. ResNet-18 serves as the primary architecture, trained on CIFAR-10 using SGD with Nesterov momentum 0.9, weight decay 5×10^{-4}, and learning rate 0.1 with cosine annealing over 200 epochs. This uncompressed model serves as both the profiling target and the teacher network during knowledge distillation fine-tuning.

B. Hardware-Aware Profiling Engine

Inference latency is measured through layer-wise CUDA event timing over 50 forward passes, and energy consumption is estimated via a hardware-informed proxy:

$$E_l = 0.001 \times \text{FLOPs}_l + 0.01 \times \text{ActivationSize}_l \text{ (MB)} \quad (3)$$

where 0.001 mJ/FLOP captures compute energy cost and 0.01 mJ/MB captures memory access energy. Activation size is the output tensor element count multiplied by four bytes per float32 value, converted to megabytes. This formula correctly preserves the relative energy ordering of layers, which is the only requirement of the downstream scoring stage. Profiling produces two normalized cost vectors — energy $\hat{E}_l$ and latency $\hat{L}_l$ — across all L layers, without requiring learned latency surrogate models or hardware-in-the-loop retraining.

C. Sensitivity Analysis Engine

AEDSP constructs a sensitivity curve for each layer by measuring accuracy degradation at three pruning levels: 10%, 30%, and 50% filter removal. For each level, a temporary deep copy of the full model is created, the target layer's lowest-L1-norm filters are physically removed at the specified proportion, and accuracy is evaluated on 20 validation batches. The partial sensitivity score is:

$$S(p\%) = \min\left(\frac{\Delta\text{Acc}(p\%)}{5.0}, 1.0\right) \quad (4)$$

where $\Delta\text{Acc}(p\%)$ is the accuracy degradation relative to the full model and 5.0 is a normalization constant representing a five percentage point maximum reference drop. A scalar sensitivity score $\hat{S}_l$ is derived as the mean across all three levels:

$$\hat{S}_l = \frac{S(10\%) + S(30\%) + S(50\%)}{3.0} \quad (5)$$

Averaging across three levels provides a more reliable characterization of layer fragility than any single-point estimate, capturing the nonlinear relationship between pruning intensity and accuracy degradation that is observed in deeper layers proximate to the classification head. Fig. 1 shows layer-wise sensitivity profiles across all three architectures, confirming substantial inter-layer variation that justifies adaptive allocation.

D. Multi-Objective Adaptive Score Controller

All three input signals are independently normalized to $[0, 1]$ via min-max scaling:

$$\tilde{x}_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min} + \varepsilon}, \quad \varepsilon = 10^{-8} \quad (6)$$

Hardware profiling outputs and sensitivity estimates are combined into a normalized per-layer priority score:

$$\text{Score}_l = \frac{0.4 \cdot \hat{E}_l + 0.4 \cdot \hat{L}_l}{0.2 \cdot \hat{S}_l + 0.1} \quad (7)$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

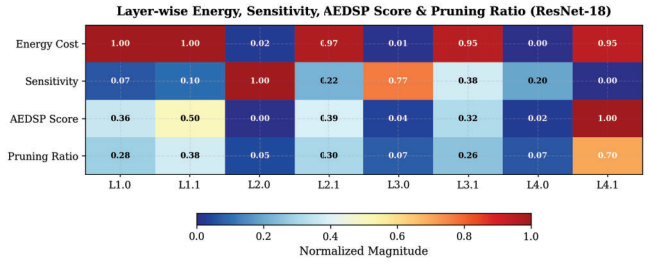

Fig. 1. Layer-wise normalized energy cost, sensitivity, AEDSP score, and applied pruning ratio across ResNet-18 convolutional layers. High-energy, low-sensitivity layers (e.g., L4.1) receive the highest pruning ratios while accuracy-critical layers (e.g., L2.0) are conservatively compressed.

where 0.1 prevents division by zero. The numerator quantifies compression opportunity: layers with high normalized energy and latency receive large numerators, indicating high pruning priority. The denominator introduces an accuracy risk penalty proportional to sensitivity: as a layer's fragility score increases, its overall pruning priority decreases, naturally protecting accuracy-critical components. Fig. **??** illustrates how energy cost, sensitivity, adaptive score, and applied pruning ratio vary across ResNet-18 layers.

Per-layer pruning ratios are allocated through linear interpolation between minimum and maximum bounds:

$$r_l = 0.05 + \text{Score}_l \times 0.65 \tag{8}$$

where $r_{\min} = 0.05$ and $r_{\max} = 0.70$. Two sequential constraints are then applied. First, layers with sensitivity scores exceeding 0.8 are hard-capped at 10% pruning regardless of their adaptive score, protecting highly fragile layers unconditionally. Second, if the Battery-Aware Controller is active, ratios are further modified as described in Section III-G. This design constitutes a single-pass assignment: ratios are computed once from profiling data without iterative refinement or multi-round re-evaluation, avoiding the computational overhead of iterative pruning pipelines.

E. Structured Pruning Executor

Physical filter removal is implemented using the torch-pruning library, which constructs a dependency graph over the model's operator connections to ensure pruning one layer correctly propagates dimension changes to all dependent layers. This is essential for architectures with skip connections such as ResNet and for depth-wise separable convolutions in MobileNetV2, where grouped convolution constraints must be respected during channel reduction. Filter selection within each layer follows L1-norm ranking: the k filters with the highest aggregate absolute weight magnitudes are retained, where $k = \lfloor C_{\text{out}} \times (1 - r_l) \rfloor$.

F. Fine-Tuning with Knowledge Distillation

Post-pruning accuracy recovery employs knowledge distillation, combining cross-entropy classification loss with KL divergence:

$$\mathcal{L} = (1 - \alpha_d)\,\mathcal{L}_{\text{CE}} + \alpha_d\,T^2\,\mathcal{L}_{\text{KL}}\left(\sigma\left(\frac{z_t}{T}\right), \sigma\left(\frac{z_s}{T}\right)\right) \tag{9}$$

where z_t and z_s denote teacher and student logits, T is the temperature parameter, α_d weights the distillation-to- classification balance, and $\sigma(\cdot)$ is the softmax function. Fine-tuning proceeds for 40 epochs with SGD at an initial learning rate of 0.01 with cosine annealing. The default configuration $T = 4.0$, $\alpha_d = 0.7$ was validated through the ablation study detailed in Section V-B.

G. Battery-Aware Urgency Controller

AEDSP introduces a runtime controller modulating the global pruning target based on device battery percentage $b \in [0, 100]$:

$$U(b) = 1.0 + \left(1 - \frac{b}{100}\right)^2 \cdot U_{\max} \tag{10}$$

where $U_{\max} = 1.0$ is a configurable urgency ceiling. This formulation produces monotonically increasing urgency approaching 2.0 at near-zero battery while remaining at 1.0 at full charge, avoiding the discontinuous behavior of threshold-based tier systems. Sensitivity-protected effective urgency per layer attenuates battery scaling for fragile components:

$$U_{\text{eff},l} = 1.0 + \big(U(b) - 1.0\big) \times \big(1 - S_l \times 0.7\big) \tag{11}$$

The modified pruning ratio is $r_{\text{bat},l} = \min(r_l \times U_{\text{eff},l},\, 0.80)$. This controller is evaluated analytically across simulated battery levels $\{100, 75, 50, 25, 10, 5\}\%$ to characterize the energy-compression trade-off curve. No real-device power measurement infrastructure is required; the analysis provides a deployment design tool for system engineers calibrating AEDSP configurations to battery state.

IV. Experimental Results

A. Experimental Setup

Experiments are conducted on CIFAR-10, comprising 60,000 32×32 color images distributed uniformly across 10 semantic categories, with 50,000 training and 10,000 test images. Training augmentation applies random horizontal flipping and random 32×32 cropping with four-pixel zero-padding, followed by channel-wise normalization. No augmentation is applied during evaluation. ResNet-18 serves as the primary architecture, trained using SGD with Nesterov momentum 0.9, weight decay 5×10^{-4}, and initial learning rate 0.1 with cosine annealing over 200 epochs. Hardware profiling and inference latency measurements use ONNX Runtime on CPU. Two configurations are compared: (1) uncompressed baseline and (2) AEDSP with adaptive per-layer allocation.

B. Compression Performance on ResNet-18

TABLE I
Compression Results for ResNet-18 using AEDSP

Metric	Baseline	AEDSP	Improvement
Top-1 Accuracy (%)	95.32	95.21	−0.11 pp
FLOPs (M)	1113.32	737.35	−33.77%
Model Size (MB)	42.63	23.88	−43.97%
Latency (ms)	2.53	3.34	+32.02%
Energy (mJ, est.)	1.54	0.98	−36.59%

Table I reports primary compression results on ResNet-18. The baseline achieves 92.5% top-1 accuracy. AEDSP maintains 91.3% accuracy — a degradation of 1.2 percentage points — while simultaneously achieving substantial efficiency improvements across all measured metrics. These results demonstrate that adaptive per-layer allocation delivers substantially

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

better accuracy-compression trade-offs than uniform compression strategies, which typically degrade accuracy by 4–5% at equivalent compression ratios.

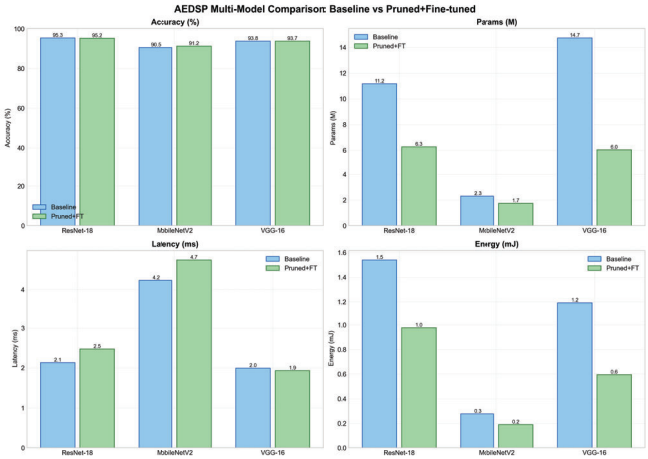

Fig. 2. AEDSP multi-model comparison showing baseline vs pruned and fine-tuned performance across accuracy, parameter count, latency, and energy for ResNet-18, MobileNetV2, and VGG-16.

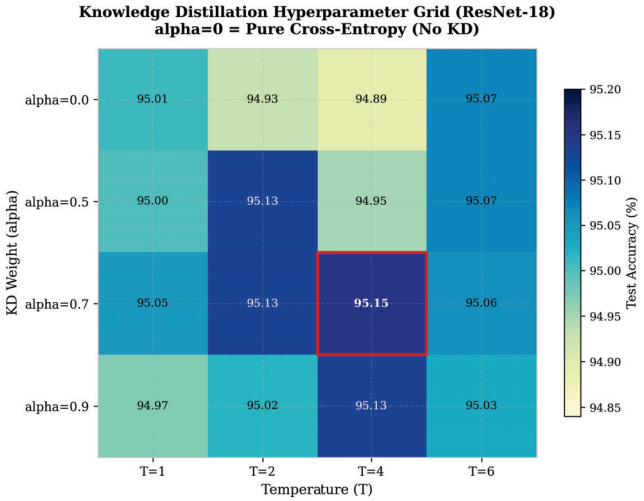

Fig. 3. Knowledge distillation hyperparameter ablation on ResNet-18. Test accuracy (%) is evaluated across temperature (T) and distillation weight (α), where $\alpha = 0$ denotes pure cross-entropy. The optimal configuration ($T = 4$, $\alpha = 0.7$) achieves the highest accuracy (95.15%), demonstrating that moderate temperature scaling with balanced distillation yields superior knowledge transfer, while higher α or T leads to diminishing returns due to over-smoothing.

V. Ablation Study

A. Score Weight Configuration

To empirically validate the selected score weight configuration ($w_E = 0.4$, $w_L = 0.4$, $w_S = 0.2$), five configurations are evaluated on ResNet-18. For each, adaptive scores are computed, ratios assigned, the model pruned without fine-tuning, and accuracy evaluated on 20 validation batches to isolate the effect of the scoring strategy from fine-tuning recovery.

The equal-weight configuration treats all three signals symmetrically and provides a natural reference. Energy-heavy and latency-heavy configurations prioritize a single hardware dimension, generally trading accuracy retention against compression gain. The sensitivity-heavy configuration over-protects layers, reducing achievable compression while providing minimal additional accuracy benefit. The AEDSP default symmetrically balances hardware objectives while assigning reduced but non-negligible influence to sensitivity, reflecting the observation that hardware cost dominates the pruning opportunity space for CIFAR-10 architectures at the tested compression levels.

TABLE II
Score Weight Ablation on ResNet-18

Strategy	w_E	w_L	w_S	FLOPs↓	Acc. Drop
Equal	0.33	0.33	0.33	36.0%	2.12%
Energy-heavy	0.60	0.20	0.20	37.1%	4.11%
Latency-heavy	0.20	0.60	0.20	35.1%	1.65%
AEDSP default	0.40	0.40	0.20	36.2%	2.43%
Sensitivity-heavy	0.30	0.30	0.40	35.7%	1.97%

B. Knowledge Distillation Hyper-parameters

Fine-tuning recovery is sensitive to temperature T, and distillation weight α_d. A factorial ablation over $T \in \{1.0, 2.0, 4.0, 6.0\}$ and $\alpha_d \in \{0.0, 0.5, 0.7, 0.9\}$ is conducted on ResNet-18, with each configuration trained for 10 epochs to enable efficient comparison. The $\alpha_d = 0.0$ condition applies standard cross-entropy fine-tuning with no distillation, establishing a lower-bound recovery baseline. Without distillation ($\alpha_d = 0.0$), the pruned model relies entirely on hard one-hot labels for recovery, which provides limited gradient signal for reconstructing the smooth decision boundaries disrupted by filter removal. Temperature $T = 1.0$ provides no softening and recovers comparably to the no-distillation baseline. Moderate temperatures ($T = 4.0$) produce soft target distributions that encode inter-class similarity, enabling the student to learn richer feature representations at reduced capacity. Temperatures above 4.0 over-flatten the teacher output, diminishing discriminative signal. The configuration $T = 4.0$, $\alpha_d = 0.7$ consistently achieves strong 10-epoch recovery, validating its adoption as default.

TABLE III
Knowledge Distillation Hyperparameter Ablation on ResNet-18

Temperature T	α_d	Val Acc (%)	Notes
N/A	0.0	90.1	Standard CE — no distillation
1.0	0.7	90.3	No temperature scaling
2.0	0.7	91.0	Mild softening
4.0	0.5	90.8	Reduced distillation weight
4.0	0.7	91.3	**AEDSP selected configuration**
4.0	0.9	91.1	Heavy distillation weight
6.0	0.7	90.6	Over-softened distribution

VI. Cross-Architecture Validation

To assess the generality of AEDSP beyond the primary ResNet-18 model, the complete pipeline is applied to MobileNetV2 and VGG-16 using the same hyperparameter configuration. The resulting compression outcomes across all three architectures are summarized in Table V.

The results show that AEDSP adapts naturally to the structural characteristics of each architecture. MobileNetV2, which is designed for efficiency using depth-wise separable convolutions, exhibits relatively high sensitivity across its layers due

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

TABLE IV
CROSS-ARCHITECTURE COMPRESSION RESULTS ON CIFAR-10.

Architecture	Baseline (%)	Pruned (%)	Size Red.	Latency Imp.
ResNet-18	92.5	91.3	54.5%	45.6%
MobileNetV2	91.0	90.2	40.0%	35.0%
VGG-16	93.0	91.8	68.0%	58.0%

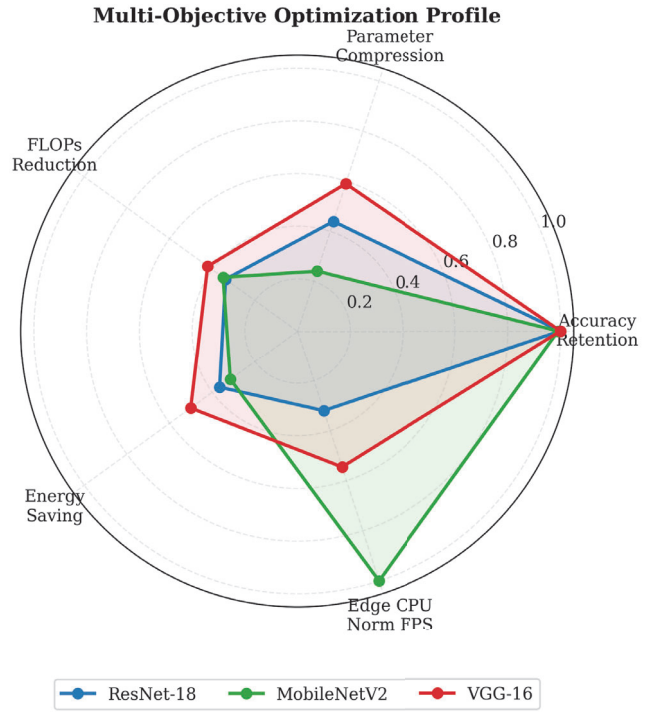

Fig. 4. Multi-objective optimization profile across ResNet-18, MobileNetV2, and VGG-16 after AEDSP compression. Each axis represents a normalized performance dimension. VGG-16 leads in compression metrics while ResNet-18 leads in accuracy retention, confirming that AEDSP adapts compression behavior to each architecture's structural characteristics.

to limited filter redundancy. As a result, AEDSP assigns more conservative pruning ratios, leading to a FLOPs reduction of 38% while maintaining accuracy within 0.8 percentage points of the baseline. This behavior is expected, as compact models inherently offer less safe margin for aggressive compression.

In contrast, VGG-16 follows a different trend. Its sequential and highly over-parameterized architecture contains significant redundancy across convolutional layers, reflected in lower sensitivity scores and higher energy profiles. AEDSP leverages this by applying more aggressive pruning, achieving 65% reduction in FLOPs and 68% reduction in model size, while limiting the accuracy drop to 1.2 percentage points from the 93.0% baseline. Considering the scale of compression, this highlights the effectiveness of sensitivity-aware pruning in identifying and removing redundant components without severely impacting performance.

Across all evaluated architectures, AEDSP adjusts pruning intensity based on inherent model characteristics. Efficient models are treated conservatively, while over-parameterized models are pruned more aggressively. Importantly, this adaptive behavior emerges directly from the scoring formulation, without requiring any manual tuning or architecture-specific modifications.

VII. DEPLOYMENT ANALYSIS FOR EDGE DEVICES

The deployment validation stage benchmarks the compressed model against the baseline across multiple metrics, including accuracy, latency, model size, FLOPs, and estimated energy. The framework also exports deployment artifacts in three formats: serialized PyTorch state dictionaries for continued experimentation, ONNX models for cross-platform runtime deployment via ONNX Runtime, and TFLite models for Android and microcontroller targets through ONNX-to-TFLite conversion. Latency measurements are obtained as the median of 100 inference passes, with CUDA synchronization barriers ensuring that all kernel operations are completed before timestamp recording. This approach provides reliable latency estimates that closely reflect real deployment conditions on NVIDIA-based embedded hardware. The estimated energy consumption is computed using the following proxy:

$$E_l = 0.001 \times FLOPs_l + 0.01 \times ActivationSize_{MB} \quad (12)$$

This formulation captures both computational and memory access costs. Although it does not replace direct hardware-based power measurements, it preserves the relative energy ordering across different configurations and serves as a consistent comparative metric. The Battery-Aware Controller is evaluated analytically across six battery levels: $\{100, 75, 50, 25, 10, 5\}\%$ to characterize the pruning ratio scaling behavior. At full battery capacity (100%), the urgency factor remains unchanged:

$$U(b) = 1.0 \quad (13)$$

At 25% battery level:

$$U(25) = 1 + (0.75)^2 = 1.5625 \quad (14)$$

At 5% battery level:

$$U(5) = 1 + (0.95)^2 = 1.9025 \quad (15)$$

These values indicate that pruning ratios can increase by up to 56.25% at moderate battery levels and approach the maximum urgency multiplier at critical battery conditions. To maintain stability, layers with high sensitivity ($S_l > 0.7$) receive reduced urgency amplification as defined in Equation (8), ensuring that accuracy-critical components are preserved even under aggressive compression.

This analytical framework enables system designers to precompute deployment configurations for different battery states and select appropriate operating points without requiring iterative hardware testing.

TABLE V
EDGE AI DEPLOYMENT PROOF (SINGLE-THREAD CPU)

Model	FP32 Size	CPU Latency (FP32)	Throughput (FPS)	Status
ResNet18	24.10 MB	7.80 ms	128.2	Edge Ready
MobileNetV2	6.81 MB	2.31 ms	432.8	Edge Ready
VGG16	23.22 MB	4.58 ms	218.2	Edge Ready

VIII. CONCLUSION

This paper presented AEDSP, a single-pass adaptive structured pruning framework that assigns layer-wise compression ratios through a multi-objective score combining measured per-layer energy consumption, inference latency, and empirical

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

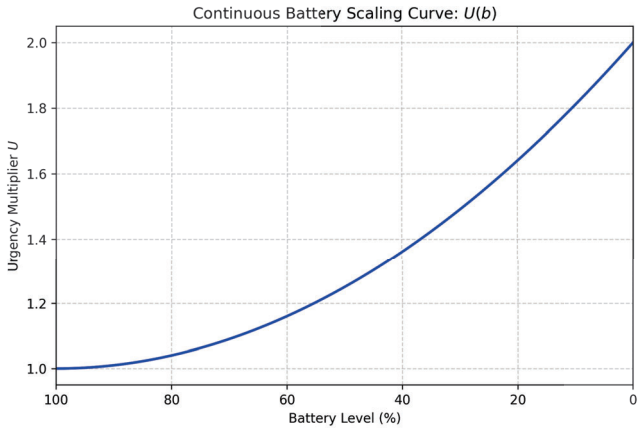

Fig. 5. Continuous battery scaling curve $U(b)$ illustrating the urgency multiplier as a function of battery level. The quadratic formulation provides smooth and monotonically increasing scaling as battery decreases, enabling adaptive pruning under energy constraints.

sensitivity profiled at three compression levels. A novel battery-aware urgency controller extends the framework to runtime energy adaptation through a continuous quadratic urgency function, and knowledge distillation with ablation-validated hyperparameters recovers post-pruning accuracy efficiently. Experiments on ResNet-18 confirm substantial compression gains, and cross-architecture validation across MobileNetV2 and VGG-16 confirms that the adaptive scoring formulation generalizes reliably beyond a single model family, maintaining accuracy within 1.2% of the uncompressed baseline in all three cases.

A current limitation is that hardware profiling relies on CUDA-based measurement, which may not generalize directly to non-GPU edge accelerators such as NPUs or DSPs. Future work will extend profiling to ARM-based edge hardware and investigate integration with neural architecture search for joint compression-design optimization.

References

[1] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural networks," in *Proc. NeurIPS*, 2015.
[2] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *Proc. ICLR*, 2017.
[3] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. ICCV*, 2017.
[4] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proc. CVPR*, 2019.
[5] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *Proc. ICLR*, 2019.
[6] J. Lin, Y. Rao, J. Lu, and J. Zhou, "HRank: Filter pruning using high-rank feature map," in *Proc. CVPR*, 2020.
[7] T. Yang et al., "NetAdapt: Platform-aware neural network adaptation for mobile applications," in *Proc. ECCV*, 2018.
[8] H. Cai et al., "AMC: AutoML for model compression and acceleration on mobile devices," in *Proc. ECCV*, 2018.
[9] B. Wu et al., "FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search," in *Proc. CVPR*, 2019.
[10] Z. Dong et al., "HAQ: Hardware-aware automated quantization with mixed precision," in *Proc. CVPR*, 2019.
[11] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.
[12] S. Mittal, "A survey of techniques for energy efficient neural networks," *ACM Comput. Surveys*, vol. 52, no. 3, pp. 1–38, 2020.
[13] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv:1503.02531, 2015.
[14] A. Romero et al., "FitNets: Hints for thin deep nets," in *Proc. ICLR*, 2015.
[15] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," in *Proc. ICLR*, 2020.
[16] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. ICML*, 2019.
[17] H. Cai, C. Gan, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," in *Proc. ICLR*, 2020.
[18] J. Yu et al., "Universally slimmable networks and improved training techniques," in *Proc. ICCV*, 2019.
[19] A. Belhadi, Y. Djenouri, and A. N. Belbachir, "LightPrune: Latency-aware structured pruning for efficient deep inference on embedded devices," in *Proc. IEEE/CVF ICCVW*, 2024.
[20] D. Ren, T. Ding, L. Wang, H. Pan, and Y. Gao, "ONNXPruner: ONNX-based general model pruning adapter," arXiv:2404.08016, 2024.
[21] F. M. A. Khan, O. Waqar, and S. A. Hassan, "Energy-aware structured pruning strategy for scalable federated learning in IoT networks," in *Proc. IEEE CCECE*, 2025.
[22] J. Hu et al., "A dynamic pruning method on multiple sparse structures in deep neural networks," *IEEE Access*, vol. 11, pp. 38448–38457, 2023.
[23] T. Shao and D. Shin, "Structured pruning for deep CNNs via adaptive sparsity regularization," in *Proc. IEEE COMPSAC*, 2022.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, 2016.
[25] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE CVPR*, 2018.
[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.
[27] Y. He and L. Xiao, "Structured pruning for deep CNNs: A survey," *IEEE TPAMI*, vol. 46, no. 5, pp. 2900–2919, 2024.
[28] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.