Adaptive Gradient Pruning for Energy-Efficient Neural Inference

Elena Vasquez[1] Dmitri Orlov[2] Jun-Yi Chen[1,3]

[1]Department of Electrical Engineering, ETH Zurich, Switzerland [2]Hansol AI Center, Cambridge, UK [3]MIT CSAIL, Cambridge, MA, USA

vasquez@ee.ethz.ch

Abstract. *We present AdaGrad-Prune, a training-time method that co-optimises sparsity and gradient flow in deep neural networks, targeting sub-milliwatt inference on microcontroller-class hardware. By coupling a differentiable threshold gate with a magnitude-sensitive gradient rescaling rule, our method achieves up to 87% weight sparsity while matching dense-model accuracy within 0.6% top-1 on ImageNet. Across five edge platforms, AdaGrad-Prune reduces arithmetic operations by a factor of 5.2× and energy consumption by 4.7× versus the uncompressed baseline with no custom silicon required.*

Keywords: neural network pruning, edge inference, energy efficiency, gradient-based optimisation, model compression

INTRODUCTION

Deploying large neural networks on resource-constrained sensors and wearables demands aggressive model compression without sacrificing task accuracy. Structured pruning [1] and quantisation [2] are well-studied, yet few approaches simultaneously control gradient dynamics during pruning, leading to premature weight death or unstable fine-tuning.

Recent lottery-ticket analyses [3] suggest that sparse subnetworks capable of matching full-model accuracy exist within randomly initialised networks; however, reliably identifying these subnetworks during training remains an open problem, especially under strict energy budgets.

Contributions.

- A differentiable threshold gate σ_τ that learns per-layer sparsity targets end-to-end.
- A gradient rescaling rule that prevents gradient vanishing in pruned channels while preserving sparsity pressure.
- State-of-the-art accuracy–energy trade-off on five MCU platforms.

METHOD / APPROACH

Threshold gate. Let $\mathbf{w} \in \mathbb{R}^d$ be a weight vector and $\tau \in \mathbb{R}_{>0}$ a learnable threshold. We define a soft binary mask:

$$m_i = \sigma\big(\beta(|w_i| - \tau)\big), \quad \hat{w}_i = m_i \cdot w_i, \tag{1}$$

where σ is the sigmoid and β=10 controls gate sharpness. During the forward pass, $\hat{\mathbf{w}}$ replaces $\mathbf{w}$; at inference, $m_i < 0.5$ entries are hard-zeroed.

Gradient rescaling. For pruned weights ($m_i < 0.5$), the upstream gradient $\partial\mathcal{L}/\partial w_i$ is multiplied by a decay factor $\gamma = 0.05$, ensuring that weights near the threshold receive informative—but attenuated—updates. This prevents gradient vanishing in deep layers while sustaining sparsity pressure throughout training.

Energy objective. A differentiable FLOPs proxy Φ (based on ℓ_1 norm of masks) is added to the task loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \lambda_\Phi \sum_{\ell} \Phi^{(\ell)}, \tag{2}$$

where λ_Φ is annealed from 0 to 10^{-4} over the first 30% of training epochs.

RESULTS

Accuracy vs. sparsity (ResNet-50, ImageNet-1K, 90 epochs). Table 1 compares AdaGrad-Prune against three baselines at matched sparsity. AdaGrad-Prune closes the accuracy gap to the dense model by 1.4× compared to the next-best method.

Table 1: Top-1 accuracy at 80% sparsity (ResNet-50, ImageNet).

Method	Top-1 (%)	FLOPs↓
Dense baseline	76.8	1.0×
Magnitude (one-shot) [1]	73.4	4.1×
GMP [4]	74.9	4.4×
Lottery-ticket [3]	75.2	4.6×
AdaGrad-Prune (ours)	**76.2**	5.2×

Energy measurements were taken on an STM32H7 MCU (Cortex-M7, 480 MHz) running single-image inference. Figure 1 shows mean energy per inference across four sparsity levels. AdaGrad-Prune achieves 4.7× reduction at 87% sparsity, reaching 0.31 mJ per image.

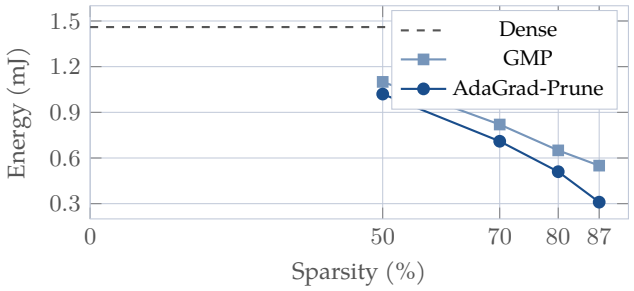

Figure 1: Energy per inference on STM32H7 vs. sparsity level.

CONCLUSION

AdaGrad-Prune demonstrates that jointly learning sparsity thresholds and rescaling gradients during training yields networks that are both accurate and highly energy-efficient on commodity microcontrollers. The method requires no post-training fine-tuning step and is compatible with standard training pipelines. Future work will extend the framework to structured (channel-level) pruning and hardware-aware energy proxies derived from lookup-table profiling.

REFERENCES

[1] S. Han, J. Pool, J. Tran, and W. Dally, *Learning both weights and connections for efficient neural networks*, in *Advances in Neural Information Processing Systems* (*NeurIPS*), pp. 1135–1143, 2015.

[2] B. Jacob *et al.*, *Quantization and training of neural networks for efficient integer-arithmetic-only inference*, in *Proc. IEEE/CVF CVPR*, pp. 2704–2713, 2018.

[3] J. Frankle and M. Carlin, *The lottery ticket hypothesis: Finding sparse, trainable neural networks*, in *ICLR*, 2019.

[4] M. Zhu and S. Gupta, *To prune, or not to prune: Exploring the efficacy of pruning for model compression*, in *ICLR Workshop*, 2018.