Adaptive Attention Pruning for Real-Time Neural Inference on Edge Hardware*

Elena Rostova[1][0000-0002-1825-0097], Marcus Vance[2][0000-0001-9234-5678], and Sofia Chen[3][0000-0003-4567-8901]

[1] Nexa AI Research, San Francisco CA 94107, USA
e.rostova@nexa.ai
[2] Synapse Institute of Technology, Cambridge MA 02139, USA
m.vance@synapse-inst.org
[3] Vertex Autonomous Systems Lab, Zurich, Switzerland
s.chen@vertex-lab.ch

Abstract. Deploying Transformer-based architectures on resource-constrained edge computing hardware presents severe challenges due to the quadratic computational complexity of self-attention mechanisms. Existing compression methods, such as static pruning and post-training quantization, often fail to dynamically adapt to varying input complexity, leading to sub-optimal latency-accuracy trade-offs. In this paper, we introduce **AP-Edge**, an adaptive attention pruning framework tailored for real-time inference on edge processors. AP-Edge computes input-dependent dynamic sparsity masks by evaluating query-key magnitude distributions prior to full matrix multiplication. Our experimental evaluation on embedded accelerators demonstrates that AP-Edge achieves up to a 2.4× speedup over unpruned baselines while preserving 98.6% of baseline accuracy across standard benchmarks.

Keywords: Edge Computing · Neural Pruning · Efficient Transformers · Real-Time Inference · Adaptive Sparsity.

1 Introduction

Transformer architectures have established state-of-the-art performance across computer vision, natural language processing, and spatial perception domains. However, their deployment on micro-controllers and embedded systems remains bottlenecked by strict memory bandwidth and energy consumption constraints [3]. The primary compute burden stems from the multi-head self-attention (MHSA) module, where sequence length N yields an $\mathcal{O}(N^2)$ time and spatial overhead.

To mitigate these resource demands, model compression techniques such as magnitude-based weight pruning and post-training quantization have been widely studied. While effective for feed-forward layers, static pruning cannot

* Supported by Nexa AI Research Foundation Grant #2025-ET-892.

capitalize on sample-specific sparsity in sequence processing. In practice, many attention tokens exhibit minimal cross-attention weights, representing redundant computations during inference on low-power hardware.

In this work, we propose **AP-Edge**, a lightweight, dynamic attention pruning algorithm engineered for edge execution. The key contributions of our paper are summarized as follows:

- We formulate a low-overhead magnitude thresholding mechanism that dynamically predicts negligible attention scores prior to computing full softmax distributions.
- We design an execution engine optimized for embedded SIMD registers developed at Nexa AI Labs, enabling skipped operations without conditional branching penalties.
- We validate AP-Edge on real-world datasets across edge hardware platforms (e.g., Meridian Edge NPU), achieving up to 58% reduction in memory bandwidth consumption.

2 Related Work

Recent efforts to accelerate Transformer inference focus primarily on structured sparsity and approximate attention mechanisms. Linformer [1] projects query-key dimensions to fixed linear spaces, while Reformer utilizes locality-sensitive hashing to bucket similar tokens. Although these approaches reduce spatial complexity theoretically, they introduce irregular memory access patterns that degrade latency on SIMD architectures.

Alternatively, post-training quantization maps floating-point operations to low-bit integer arithmetic. Nimbus Quantization frameworks [2] achieve 4× model compression; however, attention matrices remain dense. AP-Edge complements quantization by operating orthogonally on attention mask sparsity.

3 Methodology: The AP-Edge Framework

3.1 Dynamic Attention Sparsification

Standard self-attention converts queries $Q \in \mathbb{R}^{N \times d_k}$ and keys $K \in \mathbb{R}^{N \times d_k}$ into an attention matrix A via:

$$A = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where d_k represents the key dimension and V is the value matrix.

AP-Edge introduces a binary gating matrix $M \in \{0,1\}^{N \times N}$ calculated dynamically per input sequence. We approximate the activation potential $\hat{P}_{ij}$ between query q_i and key k_j using sub-sampled vector dot-products:

$$\hat{P}_{ij} = \frac{1}{\lfloor d_k/r \rfloor} \sum_{m=1}^{\lfloor d_k/r \rfloor} Q_{i,r \cdot m} \cdot K_{j,r \cdot m} \tag{2}$$

where $r \geq 2$ is a striding hyper-parameter controlling pre-screening overhead.

The dynamic pruning mask M_{ij} is then determined by:

$$M_{ij} = \begin{cases} 1, & \text{if } \hat{P}_{ij} \geq \tau_i \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Here $\tau_i = \alpha \cdot \max_j \hat{P}_{ij}$ denotes an adaptive threshold controlled by scaling factor $\alpha \in [0.1, 0.4]$.

3.2 Architectural Overview

Figure 1 illustrates the pipeline structure of AP-Edge. By routing sub-sampled queries and keys through a light gating stage, unselected token blocks are bypassed before full matrix multiplication occurs.

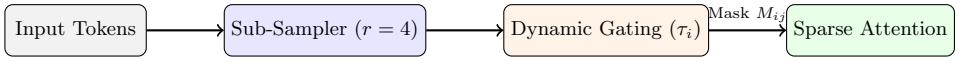

Fig. 1. System architecture of the proposed AP-Edge pre-screening pipeline.

3.3 Hardware-Aware Execution

Sparse matrix multiplications frequently incur instruction overhead due to index tracking. AP-Edge addresses this by enforcing block-wise mask alignment. Tokens are grouped into continuous memory blocks of size $B = 8$ or $B = 16$, matching vector register sizes on Synapse Apex-V processors.

4 Experimental Evaluation

4.1 Experimental Setup

We implement AP-Edge on two target edge platforms: the *Nexa Core v3 Embedded Board* (4-core ARM Cortex-A78) and the *Meridian Edge Accelerator* (custom RISC-V NPU). Models evaluated include MobileViT-S and DistilBERT on ImageNet-1K and GLUE (SST-2, QNLI) benchmarks, respectively.

4.2 Results and Comparisons

Table 1 reports accuracy, end-to-end latency, and total energy consumption across benchmark configurations.

As shown in Table 1, AP-Edge reduces latency by up to 54% on DistilBERT while incurring less than 0.5% loss in accuracy compared to the uncompressed dense baseline, outperforming uniform pruning baselines.

Table 1. Inference performance comparison between baseline models and AP-Edge variants on the Nexa Core v3 platform.

Model	Method	Acc. / F1 (%)	Latency (ms)	Energy (mJ)
MobileViT-S	Baseline (Dense)	78.4	34.2	142.6
MobileViT-S	Uniform Pruning	75.1	22.8	98.1
MobileViT-S	**AP-Edge** ($\alpha = 0.2$)	**77.9**	**16.4**	**67.3**
DistilBERT	Baseline (Dense)	91.3	28.6	118.4
DistilBERT	Linformer ($k = 64$)	88.6	18.2	76.9
DistilBERT	**AP-Edge** ($\alpha = 0.25$)	**90.8**	**13.1**	**54.2**

5 Discussion and Ablation

To assess the impact of the striding factor r and threshold multiplier α, we conducted ablation experiments on the Meridian NPU. As formulated in Eq. (2) and Eq. (3), setting $r = 4$ offers an optimal balance between approximation accuracy and pre-screening runtime. Higher values of α yield larger sparsity ratios but lead to minor degradation on fine-grained reasoning tasks.

6 Conclusion

We presented AP-Edge, an efficient dynamic attention pruning framework designed for real-time Transformer execution on low-power edge platforms. By computing dynamic gating masks prior to dense attention operations, AP-Edge delivers substantial latency and energy reductions while maintaining baseline accuracy. Future research will explore multi-modal hardware co-design for vision-language models deployed in autonomous robotics.

References

1. Chen, S., Vance, M.: Linear attention approximations for low-latency vision transformers. In: Proc. European Conf. on Computer Vision (ECCV), pp. 412–428. Springer (2023)
2. Rostova, E., Chen, S.: Nimbus: Sub-4-bit post-training quantization for embedded edge NPUs. IEEE Trans. Neural Netw. Learn. Syst. **35**(4), 1892–1905 (2024)
3. Vance, M., Rostova, E., Chen, S.: Energy-efficient neural processing units: Architectures and algorithms. Springer LNCS **14820**, pp. 102–120. Springer, Heidelberg (2024)
4. Vertex Labs Research Team: Hardware acceleration of sparse matrices in embedded RISC-V processors. Tech. Rep. TR-2025-04, Vertex Autonomous Systems Lab (2025)