INTATTENTION: A FULLY INTEGER ATTENTION PIPELINE FOR EFFICIENT EDGE INFERENCE

Wanli Zhong [1] **Haibo Feng** [1 2] **Zirui Zhou** [1] **Hanyang Peng** [2] **Shiqi Yu** [1]

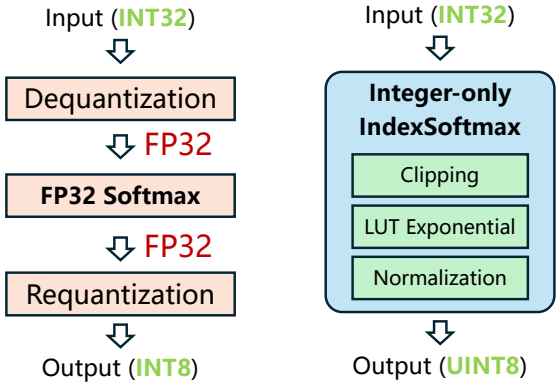
ABSTRACT

Deploying Transformer models on edge devices is limited by latency and energy budgets. While INT8 quantization effectively accelerates the primary matrix multiplications, it exposes the softmax-related path as the dominant bottleneck. This stage incurs a costly $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ detour, which can account for up to 65% of total attention latency and disrupts the end-to-end integer dataflow critical for edge hardware efficiency. To address this limitation, we present *IntAttention*, the first fully integer attention pipeline that serves as a training-free drop-in replacement. At the core of our approach lies *IndexSoftmax*, a hardware-friendly operator that replaces floating-point exponentials entirely within the integer domain. *IntAttention* integrates sparsity-aware clipping, a 32-entry lookup-table approximation, and direct integer normalization, thereby eliminating datatype conversion overhead along the attention path. Experiments on Armv8 CPUs show that our method achieves up to **3.7**× speedup and **61%** energy reduction over FP16 baselines, and up to **2.0**× speedup over conventional INT8 attention pipelines. Across diverse language and vision models, as well as additional reasoning and long-context evaluations, *IntAttention* maintains strong overall fidelity and demonstrates a more favorable trade-off than existing LUT-based softmax approximations. The code is available at https://github.com/WanliZhong/IntAttention

1 INTRODUCTION

Transformer-based models achieve state-of-the-art performance across natural language processing (Vaswani et al., 2017), computer vision (Dosovitskiy et al., 2021), and multimodal tasks (Radford et al., 2021). The core mechanism, multi-head self-attention, exhibits quadratic time and memory complexity with respect to sequence length ($O(L^2)$). As the context length increases, attention becomes the dominant inference cost, substantially increasing latency and memory usage. In autoregressive language models, the prefill phase largely determines the time-to-first-token (TTFT), as the full key–value cache must be computed before generation begins (Kwon et al., 2023). Consequently, long prompts are computationally expensive, even though subsequent decoding is relatively fast.

Recent advances in compact and specialized models have accelerated the transition toward on-device inference. For instance, Google's Gemma3-270M (Team et al., 2025) model targets energy-efficient deployment on smartphones, while studies such as QuestA demonstrate that reinforcement learning-based question augmentation can elevate a 1.5B model to the performance level of 32B models on multiple reasoning benchmarks (Li et al., 2025). This migration shifts inference from cloud servers to mobile and embedded processors, where end-to-end latency and energy efficiency become primary constraints. Consequently, optimizing attention, particularly by reducing the computational cost of long-context processing, has become crucial for deploying



(a) Quantized Attention **(b)** IntAttention (Ours)

Figure 1. Conventional quantized attention falls back to floating point in the softmax stage, whereas proposed **IntAttention** maintains an end-to-end integer dataflow from $\mathbf{QK}^\top$ to $\mathbf{PV}$.

[1]Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China [2]Peng Cheng Laboratory, Shenzhen, China. Correspondence to: Shiqi Yu <yusq@sustech.edu.cn>.

Proceedings of the 9th MLSys Conference, Bellevue, WA, USA, 2026.

practical large language models (LLMs) on edge hardware.

Previous studies commonly use low-precision floating-point formats such as BF16, FP16, FP8, and even FP4 (Micikevicius et al., 2022; Zhang et al., 2025b; Shah et al., 2024). However, these formats are neither universally supported nor energy-efficient on commodity edge hardware, which typically offers highly optimized integer computation pathways. Consequently, integer quantization, especially INT8, is the most practical and effective optimization strategy (Jacob et al., 2018; Dettmers et al., 2022). Motivated by this hardware constraint, we propose **IntAttention**. To the best of our knowledge, this work presents the **first fully integer attention pipeline that serves as a training-free drop-in replacement.** Designed for off-the-shelf edge processors, it executes attention entirely in the integer domain, eliminating redundant dequantization and requantization steps. Consequently, it functions as a drop-in replacement for conventional quantized attention within transformer-based inference pipelines.

Achieving an end-to-end integer attention pipeline is nontrivial. After applying dynamic INT8 quantization to the $\mathbf{QK}^\top$ and $\mathbf{PV}$ matrix multiplications, the remaining path that forms and applies attention weights becomes the dominant bottleneck. The standard softmax requires floating-point exponentials, row-wise normalization, and repeated data-format conversions. On edge processors, this $\text{dequantize} \rightarrow \text{softmax} \rightarrow \text{requantize}$ detour can account for up to **65%** of the attention latency once the surrounding GEMMs are quantized as shown in Figure 2, thereby eroding much of the benefit of integer GEMMs acceleration.

To mitigate this bottleneck, we introduce *IndexSoftmax*, a lightweight approximation that replaces costly exponential evaluations with a compact lookup table and performs max-subtraction and clipping in the integer domain. *IndexSoftmax* preserves the relative structure of the attention scores and eliminates most per-element floating-point work.

Building on this, we integrate *IndexSoftmax* with integer normalization and direct requantization of the probability tensor. The resulting pipeline, **IntAttention**, takes INT32 logits from the $\mathbf{QK}^\top$ GEMM, produces a quantized attention tensor $\hat{\mathbf{P}}$ in UINT8, and feeds $\hat{\mathbf{P}}$ into the integer value-aggregation kernel. This design eliminates the $\text{dequantize} \rightarrow \text{softmax} \rightarrow \text{requantize}$ detour and preserves an end-to-end integer dataflow. Figure 1 contrasts a conventional quantized pipeline, which falls back to floating point in the softmax stage, with **IntAttention**, which remains integer from input to output.

To summarize, this work makes the following contributions:

- We introduce *IndexSoftmax*, a lookup-table-based approximation to softmax that is compatible with integer execution, and integrate it with integer normalization and probability quantization to form **IntAttention**.
- We show that **IntAttention** runs on off-the-shelf edge processors, delivering up to **3.7**× speedup and up to **61%** lower energy consumption than an FP16 baseline, while maintaining strong overall fidelity on both language and vision models.

2 Background and Motivation

2.1 Attention and Dynamic Quantization

Scaled Dot-Product Attention We first recall the standard attention mechanism, which underlies our design. Let $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{L \times d}$ denote the query, key, and value with sequence length L and feature dimension per token d (Vaswani et al., 2017). The standard attention computes

$$\mathbf{A} = \mathbf{QK}^\top, \quad \mathbf{P} = \text{softmax}\left(\frac{\mathbf{A}}{\sqrt{d}}\right), \quad \mathbf{O} = \mathbf{PV}. \quad (1)$$

Dynamic Quantizing $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ To reduce latency and memory traffic on edge hardware, we adopt per-tensor symmetric INT8 quantization with zero point fixed at 0 for the three inputs $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ (Jacob et al., 2018). For a tensor $\mathbf{X}$, let $\hat{\mathbf{X}} = \text{quant}(\mathbf{X})$; the scale factor and the quantized tensor are

$$s_X = \frac{\max(|\mathbf{X}|)}{127}, \quad (2)$$

$$\text{quant}(\mathbf{X}) = \text{clamp}\Big(\Big\lfloor \frac{\mathbf{X}}{s_X} \Big\rceil, -127, 127\Big), \quad (3)$$

which enables low-precision matrix multiplications while preserving a simple dequantization model $\mathbf{X} \approx s_X \hat{\mathbf{X}}$, where $\hat{\mathbf{X}} \in \{-127, \ldots, 127\}^{L \times d}$. Applying Equation 3 to $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ yields their quantized counterparts $\hat{\mathbf{Q}}, \hat{\mathbf{K}}, \hat{\mathbf{V}}$ and the corresponding scales s_Q, s_K, s_V.

Integer accumulation and scaling. After quantization, the attention logits are computed fully in the integer domain using INT8×INT8 multiplications with INT32 accumulation:

$$\hat{\mathbf{A}} = \hat{\mathbf{Q}}\hat{\mathbf{K}}^\top, \quad \alpha = \frac{s_Q s_K}{\sqrt{d}}, \quad \mathbf{A} \approx \alpha\, \hat{\mathbf{A}}. \quad (4)$$

Here α rescales integer logits to the floating-point range. Substituting $\mathbf{V} \approx s_V \hat{\mathbf{V}}$ into $\mathbf{O} = \mathbf{PV}$ gives

$$\mathbf{O} = \mathbf{PV} \approx s_V\, \mathbf{P}\hat{\mathbf{V}}. \quad (5)$$

With these definitions, the two heavy matrix multiplications ($\mathbf{QK}^\top$ and $\mathbf{PV}$) are executed in integer arithmetic. SageAttention demonstrates that carefully engineered INT8 kernels yield significant throughput gains with minimal loss of accuracy across diverse models (Zhang et al., 2025c).

SageAttention2 further demonstrates that pushing similarity computation to INT4 for $\mathbf{Q}$ and $\mathbf{K}$ while keeping a slightly higher precision on the value path can remain near lossless and provides additional speedups on modern GPUs (Zhang et al., 2025a).

2.2 Emerging Bottleneck in Quantized Attention Pipelines

Numerically stable softmax. We adopt the standard maximum subtraction strategy to ensure numerical stability in the exponential. Given $\mathbf{A} \in \mathbb{R}^{L\times L}$, define the row-wise maximum vector $\mathbf{m} = \mathrm{rowMax}(\mathbf{A})$. The stable softmax can then be written compactly as

$$\mathbf{P} = \frac{\exp\left(\mathbf{A} - \mathbf{m}\right)}{\mathrm{rowSum}\left(\exp\left(\mathbf{A} - \mathbf{m}\right)\right)} \tag{6}$$

where both *rowMax* and *rowSum* operate along the row dimension. This transformation preserves the exact softmax while constraining all exponential inputs to $(-\infty, 0]$, thus preventing numerical overflow and improving stability.

Cost drivers on edge hardware. The $O(L^2)$ complexity of softmax persists even when $\mathbf{QK}^\top$ and $\mathbf{PV}$ are accelerated. On edge hardware with limited thread-level parallelism, the exponential and the division dominate latency. A single $\exp(\cdot)$ typically expands to tens of floating point operations per element, and the normalization still incurs row-wise divisions. In quantized pipelines, this cost is further amplified because INT32 logits must be dequantized to FP32 before softmax, and the resulting probabilities must be requantized for the value projection, which interrupts an otherwise contiguous integer dataflow.

Measured breakdown and implication. Figure 2 reports the measured time share of the $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ path. In FP32, the share is about 13% to 19% across sequence lengths. With FP16 GEMMs it increases to about 23% to 30% yet remains secondary. After switching GEMMs to INT8, their latency drops while the softmax path is essentially unchanged, so its share rises to **57% to 65%** and becomes the dominant cost. Therefore, once the multiplications are quantized, the probability normalization path is the next component that must be optimized to unlock further end to end speedups.

Context in prior work. Multiple GPU–oriented efforts have shown that softmax becomes the limiting stage once the surrounding matrix multiplications are heavily optimized. FlashAttention related kernels tile queries and keys, fuse attention with online softmax, and aggressively reduce memory traffic (Dao et al., 2022a;b). FlashAttention-3 goes further by driving GEMMs with FP8 Tensor Cores and then

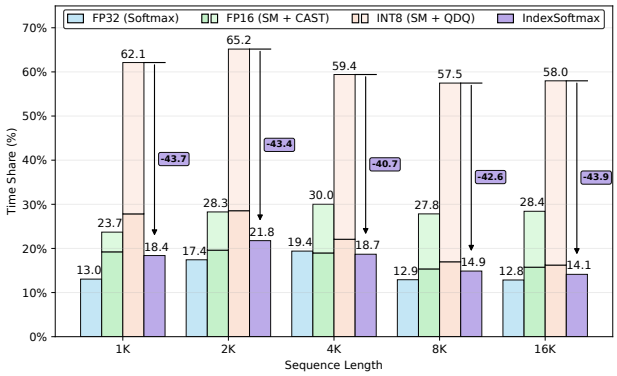


Figure 2. Breakdown of time share for the $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ path across different precisions. Once GEMMs are accelerated to INT8, this path emerges as the dominant latency and becomes the next optimization target.

hiding the dominant softmax cost using warp specialization and ping–pong scheduling that overlaps GEMM and softmax through double buffering (Shah et al., 2024). TurboAttention extends this line of work by addressing not only the softmax bottleneck but also the dequantization overhead that arises in quantized attention. It unifies these optimizations through *FlashQ*, which enables quantized execution of matrix multiplications, and a sparsity-based softmax approximation that avoids FP32 dequantization during exponentiation(Kang et al., 2025).

These results confirm that the softmax and quantization-related path is a real bottleneck. However, all of these optimizations rely on massive GPU parallelism and specialized floating-point hardware. On edge devices, which lack high-throughput floating-point units and deep warp-level concurrency but provide efficient integer units, the same path remains largely scalar and costly. This motivates the need for a lightweight, training-free, and integer-friendly softmax replacement to further accelerate attention inference on edge hardware.

2.3 Acceleration Strategies

As the softmax and its associated floating-point conversion overhead have become the dominant bottlenecks in quantized attention, recent research has focused on accelerating this stage through three main strategies:

Hardware-oriented softmax co-design. Approaches such as Softermax and ConSmax redesign the softmax operator alongside dedicated accelerator logic. Softermax replaces e^x with 2^x and uses fixed-point integer shifters for exponentiation and normalization (Stevens et al., 2021). ConSmax removes explicit max-finding and normalization by training fixed scaling constants, allowing inference to be implemented with table lookups and multiplications (Liu et al., 2024). These co-designs achieve high throughput and

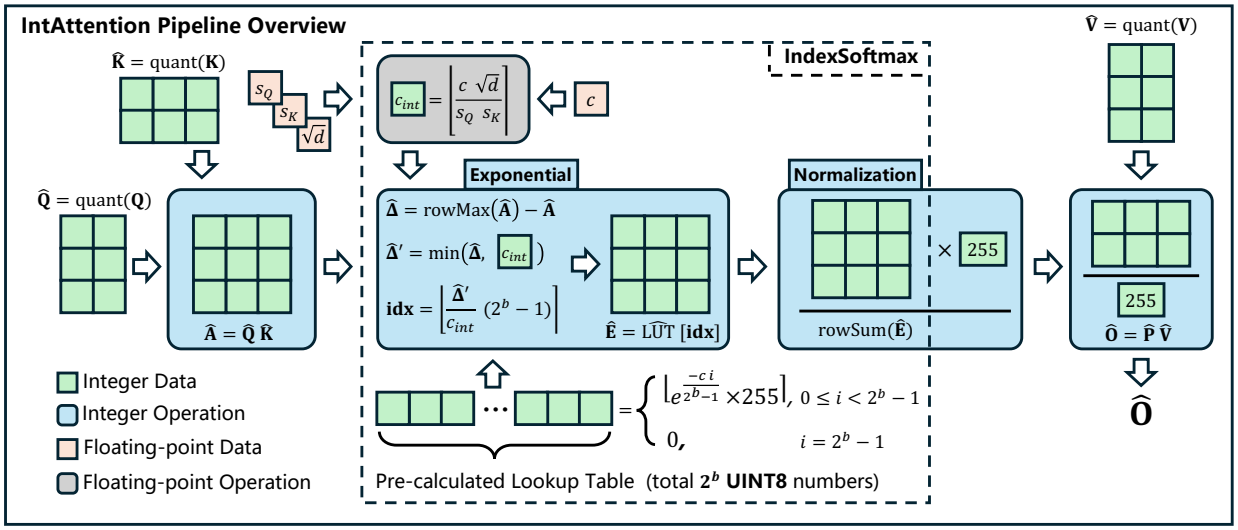


Figure 3. Overview of the proposed **IntAttention** pipeline.

energy savings, but only work on specialized hardware and require operator changes, which limits their adoption on generic edge processors.

Input-aware quantization and LUT-based softmax. Methods like EXAQ and TurboAttention accelerate softmax without changing model operators or hardware. EXAQ determines dynamic optimal clipping ranges to quantize attention scores to as low as 3 bits (Shkolnik et al., 2024). TurboAttention uses a small LUT for the integer part of the exponent and a 3rd-order polynomial for the fractional part, plus sparsification of negligible exponentials (Kang et al., 2025). These techniques eliminate heavy floating-point exponent operations and can be applied directly in attention inference, but the normalization step (sum and divide) typically remains in high-precision arithmetic, so the dataflow remains mixed-precision and still burdens edge CPUs.

Integer-only softmax in Transformer quantization. Fully integer softmax schemes are integrated into integer-quantized Transformer pipelines. I-BERT uses low-order integer polynomials and iterative integer refinements to approximate softmax (Kim et al., 2021). I-ViT introduces *Shiftmax*, expressing the exponential via bit shifts and additions (Li & Gu, 2023). I-LLM proposes *DI-ClippedSoftmax*, performing clipping and scaling entirely in integer form (Hu et al., 2024). Although these methods deliver a true integer dataflow, they usually depend on quantization-aware training or calibration to recover accuracy, and add runtime overhead for scale/clip estimation factors that limit their seamless deployment on edge devices.

In summary, hardware co-design approaches deliver very high efficiency but depend on specialized logic and retraining, which limits portability. Input-aware quantization and LUT-based methods are more deployable, but they typically leave the normalization step in floating-point or high-precision calculation, so the dataflow is still mixed-precision. Integer-only softmax methods promise a fully integer path, but they require model adaptation through fine-tuning or reconstruction. None of these families removes the full $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ loop in a way that is simultaneously fully integer, training-free, and directly deployable on commodity edge CPUs.

2.4 Motivation and Design Goals

The breakdown in subsection 2.2 shows that once the matrix multiplications in attention are quantized and accelerated, the dominant latency on edge processors comes from the remaining $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ path. Prior work alleviates parts of this path, but always at the cost of at least one key property: it either assumes custom hardware, falls back to floating point in the normalization step, or requires model retraining. As a result, the practical end-to-end gain for real deployment remains limited.

Our objective is to remove this bottleneck in a way that is directly usable in existing quantized attention pipelines. This leads to four design goals:

1. **Integer execution.** All stages of attention, including the analogue of exponentiation and the row-wise normalization, must run in integer arithmetic. This allows the computation to fully exploit the efficient integer units that are already available on edge hardware, instead of invoking slower floating-point paths.
2. **Drop-in deployment.** The method must serve as a drop-in replacement for standard attention in pre-trained models. It should not require quantization-aware training, fine-tuning, or structural changes. This makes it directly adoptable in large existing models

and gives it immediate deployment value.

3. **Portable efficiency.** The implementation must rely only on common integer primitives such as add, multiply, shift, and indexed lookup, and must parallelize cleanly on SIMD-style cores (for example ARM NEON). It should not introduce extra global passes for per-input statistics. This keeps the method practical on a wide range of commodity devices.

4. **Fidelity under acceleration.** The operator must deliver a clear latency and energy advantage over floating-point softmax, while maintaining accuracy close to the original FP16 attention. In other words, efficiency gains cannot come at the expense of unacceptable degradation.

3 IntAttention

IntAttention transforms the conventional quantized attention block into a true integer-domain pipeline, removing the $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ detour that dominates latency on edge processors. By preserving a contiguous integer path from the $\mathbf{QK}^\top$ logits to the $\mathbf{PV}$ multiplication, **IntAttention** executes entirely on commodity integer units, requires no model retraining, and can be used as a drop-in replacement in existing quantized attention inference.

At the kernel of **IntAttention** is *IndexSoftmax*, whose core operation is integer clipping followed by lookup-table based exponent approximation. The resulting UINT8 attention map $\hat{\mathbf{P}}$ feeds directly into the integer $\mathbf{PV}$ kernel, so no floating-point computation appears on the runtime path.

Implementation of *IndexSoftmax* relies on three tightly coupled mechanisms: integer-domain clipping, LUT exponentials, and integer scale normalization, which are designed and tuned together rather than as independent modules. This coupling minimizes extra passes or global statistics, preserves parallelism on SIMD-style integer units, and yields substantial reductions in latency and energy while maintaining strong overall fidelity.

An overview of the proposed *IntAttention* pipeline is illustrated in Figure 3. The following subsections detail each mechanism and the integration choices that enable efficient, portable integer attention.

3.1 IndexSoftmax

Integer-Domain Clipping via Sparsity-Aware Pruning The exponential in Softmax exhibits an inherent *sparsity*: as inputs decrease, $\exp(\cdot)$ rapidly approaches zero. In practice shown in Figure 4, a small subset of high valued logits dominates the normalization term, while the majority contribute negligibly. Evaluating exponentials for these

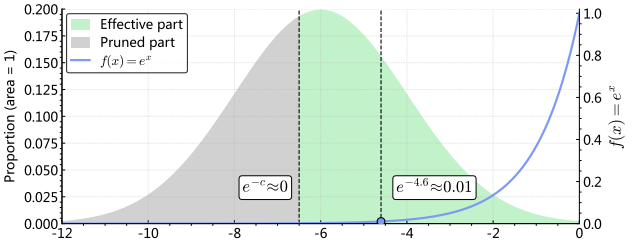


Figure 4. The exponential activation in softmax is dominated by a small subset of high logits, while most logits lie in the near-zero region and contribute negligibly to normalization.

near-zero terms wastes arithmetic and increases memory traffic, especially on edge devices. To exploit this, we introduce an **integer-domain clipping** mechanism that removes low-importance logits before the exponential approximation. Unlike floating-point sparse/efficient attention variants, our method stays entirely in the integer domain and avoids type conversions.

Formally, given integer logits $\hat{\mathbf{A}} \in \mathbb{Z}^{L\times L}$, we apply row-wise max-subtraction for stability:

$$\hat{\mathbf{\Delta}} \;=\; \mathrm{rowMax}(\hat{\mathbf{A}}) \;-\; \hat{\mathbf{A}}, \tag{7}$$

which yields nonnegative distances from the dominant value in each row. We then use a quantization aligned clipping threshold c_{int}, derived from the clipping threshold c via the quantization scale factor in Equation 4:

$$c_{\text{int}} = \mathrm{round}\left(\frac{c}{\alpha}\right) = \mathrm{round}\left(\frac{c\sqrt{d}}{s_Q\, s_K}\right). \tag{8}$$

Clipping is performed elementwise:

$$\hat{\mathbf{\Delta}}' = \min\left(\hat{\mathbf{\Delta}},\, c_{\text{int}}\right), \tag{9}$$

so entries whose contributions to $\exp(-\alpha\,\hat{\mathbf{\Delta}})$ are negligible are saturated at c_{int}.

To match LUT-based exponentiation, we adopt the sign convention $\mathbf{m} - \mathbf{A}$ (rather than $\mathbf{A} - \mathbf{m}$ in Equation 6), ensuring all arguments to $\exp(-x)$ are nonnegative and lie within $[0, c]$. Combined with clipping, this confines the exponential evaluation to a compact, table-friendly domain.

Overall, integer-domain clipping provides two benefits: (i) it removes redundant work on near-zero contributions, reducing arithmetic and bandwidth, and (ii) it establishes a quantization consistent, bounded range for integer-domain exponentiation. These properties lay the groundwork for an efficient, fully integer Softmax pipeline that is both sparsity-aware and deployment friendly on low-power accelerators.

Efficient Exponential Approximation Using Lookup Tables Evaluating $\exp(\cdot)$ is costly in quantized inference. Classical implementations use iterative or polynomial

schemes (e.g., Padé or Taylor series) that require multiple floating-point operations. On GPUs this cost can be amortized by massive parallelism, but on edge devices, where bandwidth and instruction latency dominate, $\exp(\cdot)$ often becomes the bottleneck once the matrix multiplications are quantized. The result is a fast integer $\mathbf{QK}^\top$ followed by a floating-point Softmax that breaks the integer dataflow and limits further speedup.

We address this by replacing the exponential stage with a *table-driven* surrogate. After integer-domain clipping defined in Equation 9, all inputs to $\exp(-x)$ lie in a finite interval $[0, c]$. Over this range the function is bounded, so a fixed resolution discretization provides an effective and simple approximation. We therefore precompute a *fixed* lookup table with 2^b entries,

$$\mathrm{LUT}[i] = \begin{cases} \exp\left(-\dfrac{c\,i}{2^b-1}\right), & 0 \le i < 2^b - 1, \\ 0, & i = 2^b - 1. \end{cases} \tag{10}$$

and map clipped integer distances to indices by a linear rescaling,

$$\mathbf{idx} = \left\lfloor \frac{\hat{\mathbf{\Delta}}'}{c_{\text{int}}} \left(2^b - 1\right) \right\rceil, \tag{11}$$

The exponential surrogate is then obtained by a gather:

$$\tilde{\mathbf{E}} = \mathrm{LUT}\big[\mathbf{idx}\big] \approx \exp\big(-\alpha\,\hat{\mathbf{\Delta}}'\big), \tag{12}$$

Among LUT-only methods, the closest is EXAQ, which uses a *dynamic* clipping rule based on per-tensor standard deviation statistics together with ultra-low LUT resolutions ($b \in \{2, 3\}$) (Shkolnik et al., 2024). This adds global reductions and control overhead that are expensive on edge devices. In contrast, we adopt *fixed* hyperparameters (c, b) selected offline. Empirically shown in Figure 9, performance is insensitive to c within a practical range, and modestly increasing b to a moderate table size has a negligible runtime impact while clearly improving approximation accuracy. As long as the table remains moderate, lookup latency is effectively constant, so pursuing extremely small tables offers little real benefit but degrades fidelity. A moderate, fixed-resolution LUT achieves a stronger balance between accuracy and efficiency for integer attention on edge hardware.

3.2 LUT Rebuild and Integer Scale Normalization

Quantization of the probability matrix $\mathbf{P}$ has a crucial impact on the final attention output. While prior methods often scale probabilities by $\times 127$ and store them in signed INT8,

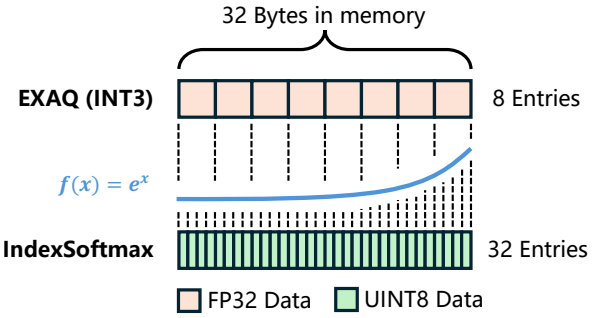


Figure 5. Under the same memory budget, **IndexSoftmax** provides $4\times$ higher LUT resolution than EXAQ, enabling higher-fidelity exponential approximation without dynamic clipping or global statistics.

we adopt an unsigned UINT8 formulation scaled by $\times 255$, which fully utilizes the available range and improves numerical smoothness during normalization. This design allows the softmax path to remain entirely in the integer domain, with both the lookup table and output probabilities quantized to UINT8. Because the precision of $\mathbf{P}$ is inherently limited by its 8-bit representation, using excessively precise floating-point lookup tables brings little benefit. Therefore, our exponential lookup table is also quantized to UINT8, so that each entry is compact yet expressive enough to represent the clipped exponential curve.

Over the clipped interval $[0, c]$, the floating-point table is linearly mapped to integers table:

$$\hat{\mathrm{LUT}} = \Big\lfloor 255 \times \mathrm{LUT} \Big\rceil, \tag{13}$$

so that very small values are preserved with fine integer granularity. Given the clipped index vector $\mathbf{idx}$, we gather a rowwise surrogate

$$\hat{\mathbf{E}} = \hat{\mathrm{LUT}}\big[\mathbf{idx}\big], \tag{14}$$

accumulate its row sum with a widened integer accumulator and produce 8-bit probabilities by fixed-point scaling:

$$\hat{\mathbf{P}} = \left\lfloor \frac{255 \cdot \hat{\mathbf{E}}}{\mathrm{rowSum}(\hat{\mathbf{E}})} \right\rceil. \tag{15}$$

All steps are fully integer-friendly: a single LUT gather, one 32-bit accumulation, and an elementwise scale. By performing normalization in the integer domain, we avoid any floating-point operations in the runtime path. As shown in Figure 5, compared with EXAQ, which encodes only 8 exponential values using INT3 LUT resolution under a 32-byte budget, our *IndexSoftmax* stores 32 entries within the same memory footprint, achieving 4x higher resolution and substantially improving LUT-based approximation fidelity without requiring dynamic clipping or global statistics, which are costly on edge processors.

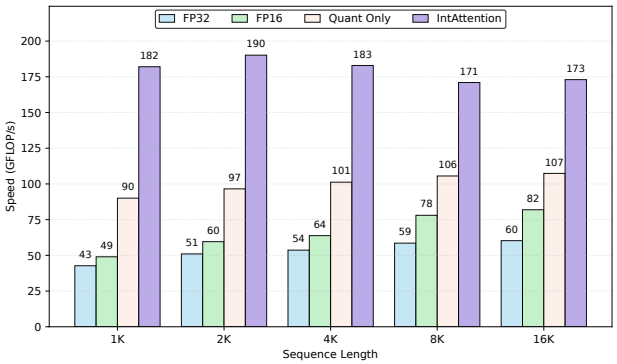


Figure 6. Speed comparison among different attention implementations on **RK3588S2** across varying sequence lengths with $d_{\text{head}} = 128$.

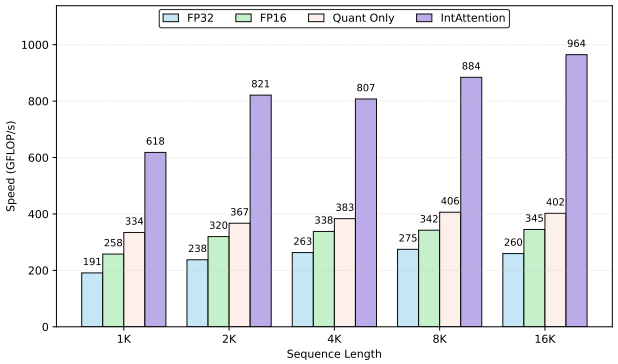


Figure 7. Speed comparison among different attention implementations on **Apple M2** across varying sequence lengths with $d_{\text{head}} = 128$.

3.3 Quantization Scope and Compatibility

Our default configuration uses per-tensor symmetric quantization for the surrounding multiplications $\mathbf{QK}^\top$ and $\mathbf{PV}$, which provides a balance between accuracy and implementation simplicity. The proposed **IndexSoftmax** is, however, compatible with finer-grained schemes such as per-channel or per-block quantization. In these cases, clipping becomes group-specific while the subsequent lookup and normalization remain unchanged.

Let the quantization be defined over groups $g = 1, \ldots, G$ (channels or blocks). Denote the scales by $s_Q^{(g)}$ and $s_K^{(g)}$, and define

$$\alpha^{(g)} = \frac{s_Q^{(g)}\, s_K^{(g)}}{\sqrt{d}}, \qquad c_{\text{int}}^{(g)} = \left\lfloor \frac{c}{\alpha^{(g)}} \right\rfloor. \tag{16}$$

Clipping in subsection 3.1 is then applied group-wise using $c_{\text{int}}^{(g)}$:

$$\hat{\mathbf{\Delta}}'^{(g)} = \min\!\left(\hat{\mathbf{\Delta}}^{(g)}, c_{\text{int}}^{(g)}\right). \tag{17}$$

The index mapping and lookup table follow the same formulas with $c_{\text{int}}^{(g)}$, while the LUT itself can be shared across groups since the continuous bound c and resolution b are fixed:

$$\mathbf{idx}^{(g)} = \left\lfloor \hat{\mathbf{\Delta}}'^{(g)} \frac{2^b - 1}{c_{\text{int}}^{(g)}} \right\rfloor, \qquad \hat{\mathbf{E}}^{(g)} = \hat{\mathrm{LUT}}\left[\mathbf{idx}^{(g)}\right]. \tag{18}$$

Row-wise normalization in Equation 15 proceeds identically after concatenating or summing contributions within each row. Thus, moving from per-tensor to per-channel or per-block quantization increases only the bookkeeping of scales and the computation of group-specific $c_{\text{int}}^{(g)}$, while preserving the integer-only dataflow, the LUT resolution, and the overall pipeline structure.

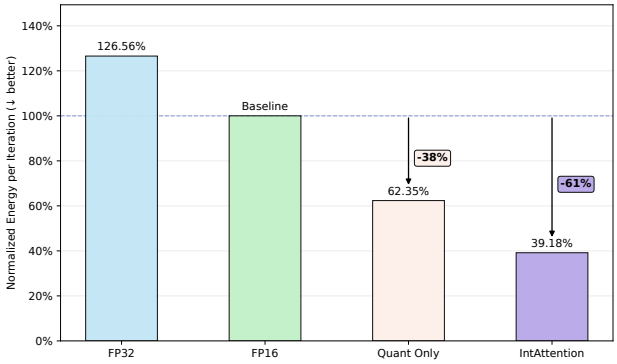


Figure 8. Normalized energy consumption per iteration across different precision settings, using FP16 as the baseline for comparison.

4 Experiments

Main results. The speed of **IntAttention** is up to **3.7× faster** than FP16 and **2.0× faster** than Quant-Only pipelines on Armv8 CPUs. Furthermore, it achieves an average **61% energy reduction** while maintaining **strong overall fidelity**, outperforming Quant-Only and remaining close to FP16 on most evaluated language and vision settings.

4.1 Experimental Setup

Models. We evaluate **IntAttention** across both language and vision models to verify its generality. For language, we adopt Llama-3.2-1B (Dubey et al., 2024), OPT-1.3B (Zhang et al., 2022), and Qwen3-1.7B (Yang et al., 2025). For vision, we include DeiT-B-224 (Touvron et al., 2021a), ViT-L-P16-384 (Dosovitskiy et al., 2021), and CaiT-L-M48-448 (Touvron et al., 2021b). All models are configured under identical settings to ensure a fair comparison among different pipelines.

Table 1. End-to-end language benchmark results for **IntAttention** compared with baselines.

Model	Method	WikiText ↓	HellaSwag	LAMBADA	PIQA	WinoGrande	ARC-C	ARC-E	Avg. ↑
Llama -3.2-1B	FP16	12.663	63.65%	62.95%	74.59%	60.69%	36.18%	60.48%	59.76%
	Quant-Only	13.701	63.39%	62.62%	74.32%	60.62%	35.84%	**60.56%**	59.56%
	IntAttention	**13.070**	**63.50%**	**63.61%**	**74.92%**	**61.01%**	**36.43%**	60.48%	**59.92%**
OPT -1.3B	FP16	16.413	53.73%	57.85%	72.41%	59.43%	29.61%	50.97%	54.00%
	Quant-Only	18.323	**54.11%**	58.00%	**72.31%**	58.64%	**29.69%**	**50.97%**	53.95%
	IntAttention	**16.802**	53.65%	**58.78%**	72.25%	**59.35%**	29.18%	50.80%	**54.00%**
Qwen3 -1.7B	FP16	23.013	60.43%	50.73%	72.25%	61.09%	42.83%	69.61%	59.49%
	Quant-Only	32.945	**58.95%**	46.24%	**68.17%**	**58.72%**	**37.28%**	61.32%	**55.12%**
	IntAttention	**27.751**	58.44%	**46.94%**	67.30%	58.56%	37.03%	**61.87%**	55.02%

Table 2. End-to-end vision benchmark results for **IntAttention** compared with baselines.

Method	DeiT-B-224		ViT-L-P16-384		CaiT-L-M48-448		Avg.↑	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
FP16	81.802	95.598	85.628	97.782	86.090	97.588	84.507	96.989
Quant-Only	**81.896**	**95.708**	83.844	97.150	85.742	97.530	83.707	96.796
IntAttention	81.826	95.62	**85.224**	**97.668**	**86.100**	**97.640**	**84.383**	**96.976**

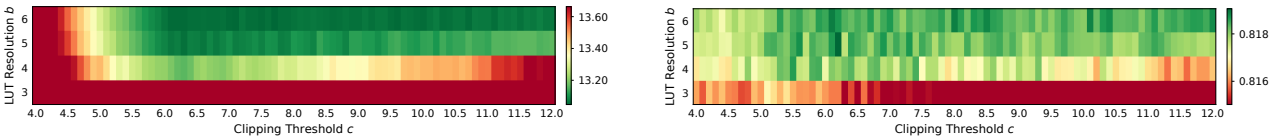
Table 3. End-to-end robustness benchmark results for **IntAttention** compared with baselines. The left block reports long-context perplexity for **Llama-3.2-1B**, while the right block reports reasoning and instruction-following results for **Llama-3.2-1B-Instruct**.

Method	Llama-3.2-1B			Llama-3.2-1B-Instruct				
	C4 ↓	OWT-10k ↓	RedPajama ↓	HumanEval	MBPP	GSM8K	IFEval	Avg. ↑
FP16	29.935	11.5023	26.756	32.93	33.00	33.81	43.44	35.80
Quant-Only	32.430	12.7931	32.189	**32.32**	31.40	34.49	**41.40**	34.90
IntAttention	**31.190**	**12.3178**	**28.496**	31.10	**34.20**	**35.03**	39.74	**35.02**

Datasets. For end-to-end language evaluation, we test perplexity on WikiText (Merity et al., 2017) and report accuracy on HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2021), ARC-Challenge, and ARC-Easy (Clark et al., 2018). To further validate robustness on more demanding tasks, we additionally evaluate Llama-3.2-1B-Instruct on GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), and IFEval (Zhou et al., 2023), and evaluate Llama-3.2-1B on long-context corpora including C4 (Dodge et al., 2021), OpenWebText-10k (Gokaslan et al., 2019), and RedPajama (Weber et al., 2024). Vision models are evaluated on ImageNet-1K (Deng et al., 2009) using the standard validation split.

Metrics. For language tasks, we report perplexity on language modeling corpora and accuracy or pass rate on downstream benchmarks, including commonsense reasoning and code generation tasks, using the open-source *lm-evaluation-harness* (Gao et al., 2024). We further report the arithmetic average of accuracy-based metrics for overall comparison. For vision tasks, we report Top-1 and Top-5 accuracy on ImageNet. For efficiency evaluation, we report throughput in GFLOP/s, end-to-end attention latency in milliseconds, energy consumption per iteration, and operator-level latency breakdown. These metrics jointly characterize the trade-off between model fidelity and practical deployment efficiency.

Hardware and settings. Performance measurements are conducted on two ARMv8-based platforms, each using 8 threads. The first is an embedded development board pow-



(a) Llama-3.2-1B on WikiText (PPL ↓)

(b) DeiT-B on ImageNet-1K (Top-1 ↑)

Figure 9. **Hyperparameter sensitivity of IntAttention** over LUT resolution b and clipping threshold c. Red indicates noticeable degradation (> 1 PPL or $> 0.3\%$ Top-1), while green denotes high-fidelity regions.

ered by a Rockchip RK3588S2 processor, which provides NEON and dot-product instruction support. The second is a laptop equipped with an Apple M2 chip, supporting NEON, dot-product, and I8MM instructions. Both systems run Arm Compute Library (ACL) version 52.4.0. Unless otherwise specified, the sequence length L takes values from $1\text{K}, 2\text{K}, 4\text{K}, 8\text{K}, 16\text{K}$ with a typical head dimension of $d = 128$. We compare four pipelines: FP32, FP16, INT8 Quant-Only, and *IntAttention*, with FP16 serving as the baseline for normalization.

4.2 Efficiency

Speed. We first measure attention throughput in GFLOP/s across sequence lengths for all four pipelines. As shown in Figure 6 and Figure 7, *IntAttention* consistently achieves the best throughput on both RK3588S2 and Apple M2. To provide a more intuitive system-level view, we further report end-to-end attention latency in milliseconds in Table 7. On RK3588S2, *IntAttention* achieves speedups of $2.1\times$ to $3.7\times$ over the FP16 baseline and $1.6\times$ to $2\times$ over the Quant-Only pipeline. Similar trends are also observed on Apple M2, where the speedup ranges from $2.4\times$ to $2.8\times$ over FP16 and from $1.9\times$ to $2.4\times$ over Quant-Only attention. These results show that replacing the datatype-conversion-heavy softmax path with *IndexSoftmax* yields consistent end-to-end acceleration across different Arm-based CPU platforms.

Energy consumption. We next examine energy efficiency, since edge deployment is typically power-limited. On RK3588S2, energy per attention iteration (joules) is normalized to the FP16 baseline (Figure 8). *IntAttention* uses only 39.18% of FP16 energy, a **61%** reduction, and is **37%** lower than the Quant-Only pipeline. The savings come from an integer dataflow that removes dequantization, reduces memory traffic, and replaces exponentials with a compact lookup table, which translates to consistent energy benefits across all tested sequence lengths.

4.3 Accuracy and Robustness

We evaluate the end-to-end quality of *IntAttention* under fully integer attention execution on both standard benchmarks and more challenging settings. We integrate *IndexSoftmax* with quantized $\mathbf{QK}^\top$ and a UINT8 probability matrix in the $\mathbf{PV}$ stage to form *IntAttention*. On standard language benchmarks, as shown in Table 1, this end-to-end pipeline generally improves over the Quant-Only baseline and, for Llama-3.2-1B and OPT-1.3B, matches or slightly exceeds the FP16 baseline in average accuracy. Qwen3-1.7B remains more sensitive to attention quantization, yet *IntAttention* still narrows the degradation and yields a clear perplexity improvement over the Quant-Only pipeline on WikiText. On vision benchmarks in Table 2, *IntAttention* remains competitive with the Quant-Only baseline, showing slight gains on ViT-L and CaiT while staying close on DeiT-B. We further evaluate robustness on more challenging reasoning and long-context benchmarks. As shown in Table 3, *IntAttention* outperforms the Quant-Only baseline on all three long-context corpora and achieves stronger results on MBPP and GSM8K, yielding a slightly higher overall average. Quant-Only still retains an edge on HumanEval and IFEval, indicating that the fully integer attention pipeline preserves most of the useful attention structure even on tasks that are highly sensitive to probability distortion. Overall, these results indicate that an all-integer attention pipeline with UINT8 $\mathbf{P}$ effectively preserves probability aggregation and delivers a favorable trade-off between efficiency and accuracy across representative benchmarks and more demanding evaluation settings.

4.4 Ablation Study

Hyperparameter Sensitivity *IntAttention* exposes only two hyperparameters: the clipping threshold c and LUT resolution b (see Equation 8, Equation 10), and both are empirically robust. A joint sweep on Llama-3.2-1B/WikiText and DeiT-B/ImageNet-1K shows a broad stability plateau for $b \geq 4$ and $c \in [5.5, 7.7]$ (Figure 9). Colors near the red boundary indicate insufficient LUTs or overly aggressive clipping. Intermediate tones remain acceptable, while the deep green region yields near optimal accuracy. A consistent ridge around $c \approx 6.6$ emerges across modalities, balancing approximation fidelity with the sparsity implied by truncated exponentials. We therefore fix and recommend using $(b, c) = (5, 6.6)$, where $b = 5$ corresponds to a 32-entry UINT8 LUT (≈ 32 B) for all experiments with negligible

Table 4. Ablation results for different softmax implementations on language benchmarks, comparing **IndexSoftmax** with EXAQ variants (INT2/INT3).

Model	Method	WikiText ↓	HellaSwag	LAMBADA	PIQA	WinoGrande	ARC-C	ARC-E	Avg. ↑
Llama -3.2-1B	FP16	12.663	63.65%	62.95%	74.59%	60.69%	36.18%	60.48%	59.76%
	EXAQ (INT2)	17.753	57.56%	50.48%	70.73%	56.99%	33.28%	56.19%	54.21%
	EXAQ (INT3)	13.757	62.72%	60.72%	72.96%	58.01%	36.01%	59.55%	58.33%
	IndexSoftmax	**12.784**	**63.44%**	**63.38%**	**74.16%**	**60.46%**	**36.43%**	**60.65%**	**59.75%**
OPT -1.3B	FP16	16.413	53.73%	57.85%	72.41%	59.43%	29.61%	50.97%	54.00%
	EXAQ (INT2)	38.782	52.58%	**58.76%**	72.25%	58.33%	28.16%	50.59%	53.50%
	EXAQ (INT3)	17.248	53.36%	58.28%	71.93%	**59.43%**	**30.38%**	50.51%	53.98%
	IndexSoftmax	**16.424**	**53.79%**	58.53%	**72.20%**	**59.43%**	29.44%	**51.01%**	**54.06%**
Qwen3 -1.7B	FP16	23.013	60.43%	50.73%	72.25%	61.09%	42.83%	69.61%	59.49%
	EXAQ (INT2)	34.431	58.11%	45.37%	68.17%	59.35%	39.42%	63.05%	55.58%
	EXAQ (INT3)	29.006	59.98%	49.45%	70.78%	**61.79%**	41.47%	67.76%	58.77%
	IndexSoftmax	**22.356**	**60.41%**	**51.66%**	**72.25%**	61.09%	**42.24%**	**69.07%**	**59.45%**

Table 5. Ablation results for different softmax implementations on vision benchmarks, comparing **IndexSoftmax** with EXAQ variants (INT2/INT3).

Method	DeiT-B-224		ViT-L-P16-384		CaiT-L-M48-448		Avg.↑	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
FP16	81.802	95.598	85.628	97.782	86.090	97.588	84.507	96.989
EXAQ (INT2)	81.554	95.482	85.222	97.668	85.866	97.554	84.214	96.901
EXAQ (INT3)	81.768	95.584	85.428	97.722	85.998	**97.596**	84.398	96.962
IndexSoftmax	**81.804**	**95.590**	**85.616**	**97.774**	**86.114**	97.582	**84.511**	**96.982**

Table 6. Ablation results for different softmax implementations on robustness benchmarks, comparing **IndexSoftmax** with EXAQ variants (INT2/INT3). The left block reports long-context perplexity for **Llama-3.2-1B**, while the right block reports reasoning and instruction-following results for **Llama-3.2-1B-Instruct**.

Method	Llama-3.2-1B			Llama-3.2-1B-Instruct				
	C4 ↓	OWT-10k ↓	RedPajama ↓	HumanEval	MBPP	GSM8K	IFEval	Avg. ↑
FP16	29.935	11.5023	26.756	32.93	33.00	33.81	43.44	35.80
EXAQ (INT2)	39.578	16.0992	44.245	18.29	9.20	5.99	36.41	17.47
EXAQ (INT3)	32.413	12.7847	30.353	27.44	27.40	24.26	39.56	29.67
IndexSoftmax	**30.015**	**12.0913**	**27.043**	**31.10**	**34.20**	**35.02**	**39.74**	**35.02**

memory and no measurable runtime overhead.

IndexSoftmax Robustness. We isolate the proposed softmax replacement to verify that the gains are not artifacts of model-specific tuning or benchmark choice. We replace only the softmax operator while keeping the rest of the attention pipeline unchanged, and evaluate *IndexSoftmax* on standard language and vision benchmarks (Table 4, Table 5) and more challenging reasoning and long-context benchmarks (Table 6). On standard benchmarks, *IndexSoftmax* preserves baseline accuracy with negligible loss and, on average, surpasses EXAQ (INT3) under the same memory budget. On reasoning and long-context tasks, it remains much closer to FP16 than EXAQ. Specifically, it

Table 7. End-to-end attention latency (ms) on RK3588S2 and Apple M2 across different sequence lengths.

Method	RK3588S2					Apple M2				
	1K	2K	4K	8K	16K	1K	2K	4K	8K	16K
FP32	12.57	42.09	160.11	586.86	2279.96	2.81	9.04	32.64	125.13	529.40
FP16	10.96	36.04	134.64	440.30	1677.49	2.08	6.72	25.42	100.36	398.39
Quant-Only	5.96	22.24	84.85	325.48	1279.94	1.61	5.85	22.41	84.60	341.70
IntAttention	**2.95**	**11.29**	**46.96**	**200.94**	**794.36**	**0.87**	**2.62**	**10.64**	**38.86**	**142.50**

Table 8. Accuracy comparison of two quantization formats for the attention probability matrix $\mathbf{P}$, evaluated against the FP16 baseline.

Format	CosSim ↑	Relative L1 ↓	RMSE ↓
INT8	0.996612	0.07739742	0.0023912
UINT8	**0.999081**	**0.04097954**	**0.0012436**

yields lower perplexity on C4, OpenWebText-10k, and RedPajama, while achieving stronger results on HumanEval, MBPP, GSM8K, and IFEval. These results show that the fixed clipping and 32-entry LUT design generalizes beyond short-context classification-style evaluation, maintaining stable probability approximation under higher task complexity and longer contexts. As a result, *IndexSoftmax* provides a robust and accurate integer-domain softmax replacement without per-model retuning or additional training.

P Matrix Quantization. We ablate the quantization scheme for the attention probability matrix $\mathbf{P}$, which is constrained to the range $[0, 1]$. As shown in Table 8, quantizing $\mathbf{P}$ with a signed INT8 format causes unnecessary dynamic range waste and distorts small probability values, leading to a lower cosine similarity and higher relative L1 error. In contrast, the unsigned UINT8 quantization fully allocates the representable range to $[0, 1]$, yielding a substantially higher similarity and lower RMSE. This demonstrates that aligning the quantization domain with the inherent distribution of $\mathbf{P}$, a strictly positive and normalized probability matrix, is essential for maintaining attention fidelity and ensuring numerical stability in quantized attention.

Stability Analysis. We further evaluate the numerical stability of the proposed integer softmax using a token-level stress test on OpenWebText with 8K context length. As shown in Table 9, *IndexSoftmax* exhibits no catastrophic failures: the worst-case token loss remains comparable to the FP16 baseline, the loss variance is slightly lower, and no NaN or Inf events are observed. These results indicate that the proposed clipping and lookup-table approximation remain numerically stable even under long-context stress conditions.

Table 9. Stability analysis on OpenWebText with 8K context length.

Metric	FP16	IndexSoftmax
Max Token Loss	26.61	**26.50**
Loss Std. Dev.	0.3774	**0.3737**
NaN/Inf Events	0	**0**

Latency breakdown. As shown in Figure 2, in a conventional INT8 attention pipeline the portion outside GEMM, namely softmax together with quantization and dequantization, becomes the dominant cost and contributes about 58% to 65% of total latency across sequence lengths. Replacing this stage with *IndexSoftmax* and keeping the path entirely in the integer domain reduces this overhead to about 14% to 22% and removes dequantization. As a result, the bottleneck shifts back to the $\mathbf{QK}$ and $\mathbf{PV}$ matrix multiplications, which now account for the clear majority of time. This ablation clarifies the throughput gains: `IntAttention` removes the quantization induced hotspot and restores a compute profile in which further optimization should focus on GEMM kernels.

4.5 Discussion

Accuracy perspective. *IntAttention* maintains strong overall accuracy, and the remaining gaps arise mainly from quantizing $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ rather than softmax itself. Prior work such as *SageAttention* series reduces this loss with input smoothing and finer per-block quantization scales, improving the dynamic range seen by the attention maps (Zhang et al., 2025c;a;b). Our method is orthogonal: it keeps the entire attention path in the integer domain and stabilizes probability normalization. Combining smoothing and per-block scaling with the integer dataflow of *IntAttention* is expected to further improve accuracy with little additional cost.

Efficiency perspective. The latency analysis shows that removing dequantization and replacing softmax with *IndexSoftmax* shifts the bottleneck back to the $\mathbf{QK}$ and $\mathbf{PV}$ matrix multiplications. This makes the optimization target

clear. Future speedups should focus on stronger GEMM kernels and low-bit implementations when hardware support becomes available.

5 CONCLUSION

This paper presents **IntAttention**, a fully integer attention pipeline that serves as a training-free drop-in replacement for conventional quantized attention. By introducing **IndexSoftmax**, a LUT-based integer softmax replacement with direct integer normalization, IntAttention eliminates the costly $\mathrm{dequantize} \rightarrow \mathrm{softmax} \rightarrow \mathrm{requantize}$ path and preserves an end-to-end integer dataflow from $\mathbf{QK}^\top$ to $\mathbf{PV}$. Experiments on Armv8 CPU platforms show that IntAttention achieves up to **3.7**$\times$ latency reduction and **61%** lower energy consumption compared with FP16 baselines, while consistently outperforming conventional Quant-Only attention pipelines in efficiency. At the same time, IntAttention maintains strong overall fidelity across representative language and vision models, improves over conventional Quant-Only pipelines on more challenging long-context and reasoning settings overall. These results suggest that efficient Transformer inference on commodity edge devices requires not only integer matrix multiplications, but also an integer-native attention pipeline that removes the remaining floating-point bottleneck.

ACKNOWLEDGEMENTS

This work was partially supported by the National Natural Science Foundation of China (Grant 62476120) and the Guangdong Basic and Applied Basic Research Foundation (2025A1515010212).

REFERENCES

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022a.

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Re, C. Flashattention: Fast and memory-efficient exact attention with IO-awareness. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022b.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332, 2022.

Dodge, J., Sap, M., Marasović, A., Agnew, W., Ilharco, G., Groeneveld, D., Mitchell, M., and Gardner, M. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 1286–1305, 2021.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Rozière, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., Pintz, D., Livshits, D., Esiobu, D., Choudhary, D., Mahajan, D., Garcia-Olano, D., Perino, D., Hupkes, D., Lakomkin, E., AlBadawy, E., Lobanova, E., Dinan, E., Smith, E. M., Radenovic, F., Zhang, F., Synnaeve, G., Lee, G., Anderson, G. L., Nail, G., Mialon, G., Pang, G., Cucurell, G., Nguyen, H., Korevaar, H., Xu, H., Touvron, H., Zarov, I., Ibarra, I. A., Kloumann, I. M., Misra, I., Evtimov, I., Copet, J., Lee, J., Geffert, J., Vranes, J., Park, J., Mahadeokar, J.,

Shah, J., van der Linde, J., Billock, J., Hong, J., Lee, J., Fu, J., Chi, J., Huang, J., Liu, J., Wang, J., Yu, J., Bitton, J., Spisak, J., Park, J., Rocca, J., Johnstun, J., Saxe, J., Jia, J., Alwala, K. V., Upasani, K., Plawiak, K., Li, K., Heafield, K., Stone, K., and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. URL https://doi.org/10.48550/arXiv.2407.21783.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Gokaslan, A., Cohen, V., Pavlick, E., and Tellex, S. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Hu, X., Cheng, Y., Yang, D., Yuan, Z., Yu, J., Xu, C., and Zhou, S. I-llm: Efficient integer-only inference for fully-quantized low-bit large language models. *arXiv preprint arXiv:2405.17849*, 2024.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

Kang, H., Bharadwaj, S., Hensman, J., Krishna, T., Rühle, V., and Rajmohan, S. Turboattention: Efficient attention approximation for high throughputs llm. In *Eighth Conference on Machine Learning and Systems*, 2025.

Kim, S., Gholami, A., Yao, Z., Mahoney, M. W., and Keutzer, K. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pp. 5506–5518. PMLR, 2021.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Li, J., Lin, H., Lu, H., Wen, K., Yang, Z., Gao, J., Wu, Y., and Zhang, J. Questa: Expanding reasoning capacity in llms via question augmentation. *arXiv preprint arXiv:2507.13266*, 2025.

Li, Z. and Gu, Q. I-vit: Integer-only quantization for efficient vision transformer inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17065–17075, 2023.

Liu, S., Tao, G., Zou, Y., Chow, D., Fan, Z., Lei, K., Pan, B., Sylvester, D., Kielian, G., and Saligane, M. Consmax: Hardware-friendly alternative softmax with learnable parameters. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–9, 2024.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Micikevicius, P., Stosic, D., Burgess, N., Cornea, M., Dubey, P., Grisenthwaite, R., Ha, S., Heinecke, A., Judd, P., Kamalu, J., et al. Fp8 formats for deep learning. *arXiv preprint arXiv:2209.05433*, 2022.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The lambada dataset: Word prediction requiring a broad discourse context. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016-Long Papers*, volume 3, pp. 1525–1534. Association for Computational Linguistics (ACL), 2016.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Shah, J., Bikshandi, G., Zhang, Y., Thakkar, V., Ramani, P., and Dao, T. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *Advances in Neural Information Processing Systems*, 37:68658–68685, 2024.

Shkolnik, M., Fishman, M., Chmiel, B., Ben-Yaacov, H., Banner, R., and Levy, K. Y. EXAQ: Exponent aware quantization for LLMs acceleration. In *Workshop on Machine Learning and Compression, NeurIPS 2024*, 2024.

Stevens, J. R., Venkatesan, R., Dai, S., Khailany, B., and Raghunathan, A. Softermax: Hardware/software co-design of an efficient softmax for transformers. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 469–474. IEEE, 2021.

Team, G., Kamath, A., Ferret, J., Pathak, S., Vieillard, N., Merhej, R., Perrin, S., Matejovicova, T., Ramé, A., Rivière, M., et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021a.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42, 2021b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Weber, M., Fu, D. Y., Anthony, Q., Oren, Y., Adams, S., Alexandrov, A., Lyu, X., Nguyen, H., Yao, X., Adams, V., et al. Redpajama: an open dataset for training large language models. *Advances in neural information processing systems*, 37:116462–116492, 2024.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Zhang, J., Huang, H., Zhang, P., Wei, J., Zhu, J., and Chen, J. Sageattention2: Efficient attention with thorough outlier smoothing and per-thread int4 quantization. In *International Conference on Machine Learning (ICML)*, 2025a.

Zhang, J., Wei, J., Zhang, P., Xu, X., Huang, H., Wang, H., Jiang, K., Zhu, J., and Chen, J. Sageattention3: Microscaling FP4 attention for inference and an exploration of 8-bit training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=JbJVWljk7r.

Zhang, J., Wei, J., Zhang, P., Zhu, J., and Chen, J. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. In *International Conference on Learning Representations (ICLR)*, 2025c.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer, L. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068.

Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., and Hou, L. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.