EXP-REDUCED TWO-LEVEL FP8 ATTENTION FOR VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

As GPU compute throughput increases, softmax becomes an increasingly significant bottleneck in attention on modern GPUs, particularly those based on NVIDIA's Blackwell architecture. Reducing normalization cost while retaining pretrained accuracy remains a challenge. This work proposes the Exp-Reduced Two-Level (ERTL) function, a precision-agnostic alternative to softmax for vision transformers (ViTs). The ERTL function uses block statistics and shifted-ReLU weights for intra-block normalization while retaining exponential normalization across blocks. This design substantially reduces the number of exponential evaluations and maps naturally to tiled attention kernels with an appropriate block size. We implement ERTL in a fused FP8 attention kernel with lazy output rescaling. Experiments on classification, detection, and segmentation show that models using ERTL recover near-baseline accuracy after adaptation, with attention-focused fine-tuning in most cases. On B200, FP8 ERTL achieves up to 14.5% higher throughput than FP8 FlashAttention-4, already optimized for Blackwell, and up to 1.55× BF16 exact-softmax throughput in long-sequence benchmarks.

1 INTRODUCTION

Attention efficiency directly affects the performance and scalability of Transformer models (Vaswani et al., 2017; Dosovitskiy et al., 2021; Dao et al., 2022; Dao, 2024). As Tensor Core throughput increases, attention bottlenecks shift toward non-matrix operations, notably softmax (Zadouri et al., 2026). Figure 1 shows that softmax already accounts for a large fraction of the runtime of decomposed BF16 attention on B200 at only 512 tokens.

Quantization further exposes this imbalance. The softmax bottleneck persists in quantized variants of highly optimized kernels such as FlashAttention-4 (Zadouri et al., 2026). As shown in Fig. 2, FP8 quantization increases the softmax time fraction to approximately 48% for both standard attention and grouped-query attention (GQA). The peak-rate gap grows further at 4-bit precision.

Two directions address this imbalance. At the algorithmic level, sigmoid (Ramapuram et al., 2025) and ReLU (Wortsman et al., 2023; Shen et al., 2023) replace softmax, but require suitable normalization and adaptation to recover pretrained accuracy. At the kernel level, FlashAttention-4 approximates exponential operations, the most expensive part of softmax, with a polynomial evaluated using Horner's method (Zadouri et al., 2026). This optimization reduces exponential latency but is not sufficient to remove the softmax bottleneck from quantized attention.

We make three contributions:

- The softmax bottleneck in modern attention kernels is analyzed, and criteria for a suitable softmax alternative are established.
- A new function, Exp-Reduced Two-Level (ERTL), is proposed. The approximation error of the ERTL function is analyzed, and an FP8 attention kernel is designed to implement it efficiently.
- We evaluate ERTL on classification, detection, and segmentation. On B200, FP8 ERTL achieves up to 14.5% more throughput than FP8 FlashAttention-4, already optimized for Blackwell, and up to 1.55× BF16 exact-softmax throughput in long-sequence benchmarks.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

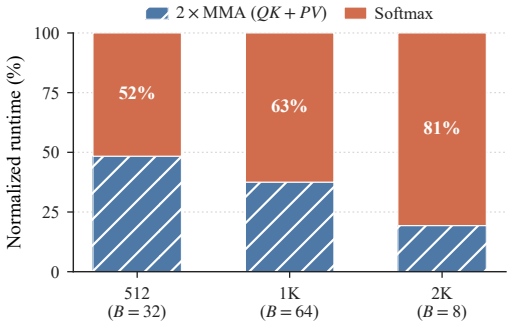


Figure 1: Runtime breakdown for decomposed BF16 attention (H=16, D=128). Batch size B varies as labeled; softmax accounts for 52–81% across these configurations.

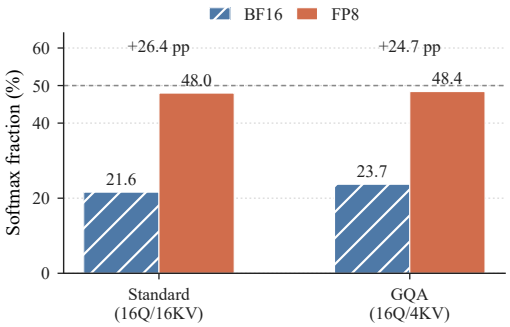


Figure 2: Softmax time fraction in BF16 and FP8 FlashAttention-4 at sequence length N=32K and head dimension D=128.

2 Softmax Bottleneck Analysis

FlashAttention-style kernels pipeline the $QK^\top$ and PV matrix multiplications with softmax normalization. The matrix multiplications execute on Tensor Cores, whereas softmax uses reductions, scalar arithmetic, and SFU exponentials. Consequently, increasing matrix-multiplication throughput can expose normalization as the limiting stage. Figures 1 and 2 illustrate this imbalance on B200 in BF16 and FP8 attention.

A heuristic per-tile model gives $T_{\mathrm{sm}}/T_{QK} \approx [\kappa_{\mathrm{sm}}/(2D)]R_{\mathrm{eff}}$, where D is the head dimension, $\kappa_{\mathrm{sm}} \approx 5$ counts scalar-equivalent normalization operations, and $R_{\mathrm{eff}} = P_{\mathrm{TC}}/P_{\mathrm{SFU}}$ is the Tensor-Core–SFU throughput ratio. At $D = 128$, its crossover is $R_{\mathrm{eff}} > 51.2$. The B200 peak-rate ratios are 512, 1024, and 2048 for BF16, FP8, and NVFP4 (Zadouri et al., 2026; NVIDIA, 2026). These ratios illustrate the hardware imbalance. Reductions and other scalar operations do not all execute on the SFU, so this calculation does not predict fused-kernel latency or its actual bottleneck; Appendix D details the assumptions.

These observations motivate three requirements for a practical softmax alternative: **sufficient exponential reduction** to relieve the normalization bottleneck, **accuracy parity** after adaptation, and **low adaptation cost** when starting from pretrained softmax weights. The ERTL function in Section 3.1 is designed around these requirements.

3 Method

Generally, our ERTL function is designed through a two-level decomposition: exponential competition is retained across blocks, whereas intra-block exponentials are replaced by shifted-ReLU weighting. The following subsections define the ERTL function in detail and describe its adaptation scheme, analyze the resulting approximation error, and present a fused FP8 kernel with lazy-O accumulation.

3.1 ERTL Function

Let $[n] = \{1, \ldots, n\}$. Consider one query $q \in \mathbb{R}^D$ attending to N=BM key–value pairs, partitioned into B contiguous blocks of M tokens. When N is not divisible by M in implementation, every remaining token is treated as a singleton block; its intra-block weight is therefore one, and it participates in the same outer softmax as the M-token blocks. Block $b \in [B]$ contains positions $j \in [M]$, with key $k_{b,j} \in \mathbb{R}^D$ and value $v_{b,j} \in \mathbb{R}^{D_v}$; D and D_v denote the query/key and value dimensions, respectively. The scaled attention score, block maximum, and nonnegative score gap are

$$z_{b,j} = \frac{q^\top k_{b,j}}{\sqrt{D}}, \qquad m_b = \max_j z_{b,j}, \qquad d_{b,j} = m_b - z_{b,j}. \tag{1}$$

Exact softmax defines

$$a^S_{b,j} = \frac{\exp(z_{b,j})}{\sum_{b'=1}^{B}\sum_{j'=1}^{M}\exp(z_{b',j'})}, \qquad o^S = \sum_{b=1}^{B}\sum_{j=1}^{M} a^S_{b,j} v_{b,j}. \tag{2}$$

The superscripts S and E denote exact softmax and ERTL, respectively. This convention is used throughout the analysis.

The exact distribution factors into a block mass and a conditional distribution within each block. Defining the effective block mass

$$R_b = \sum_{j=1}^{M} \exp(-d_{b,j}), \tag{3}$$

gives

$$a^S_{b,j} = \alpha^S_b \beta^S_{j|b}, \quad \alpha^S_b = \frac{\exp(m_b)R_b}{\sum_{b'=1}^{B}\exp(m_{b'})R_{b'}}, \qquad \beta^S_{j|b} = \frac{\exp(-d_{b,j})}{R_b}. \tag{4}$$

Our ERTL function approximates the two factors separately. At the inter-block level, it retains a softmax over all B block maxima while omitting the effective-mass factor R_b:

$$\alpha^E_b = \frac{\exp(m_b)}{\sum_{b'=1}^{B}\exp(m_{b'})}. \tag{5}$$

At the intra-block level, the exponential is replaced by a shifted ReLU. Each block b has a trainable threshold δ_b. The thresholds are initialized to 3.0 and parameterized directly, with clipping applied after each optimizer update to keep them positive. Let $x_+ = \max(x, 0)$ denote the positive part and define the local normalizer

$$L_b(\delta_b) = \sum_{j=1}^{M} (\delta_b - d_{b,j})_+, \qquad \beta^E_{j|b} = \frac{(\delta_b - d_{b,j})_+}{L_b(\delta_b)}. \tag{6}$$

The clipping step and the zero-gap maximizer in every block keep the local normalizer nonzero. The ERTL distribution and output are

$$a^E_{b,j} = \alpha^E_b \beta^E_{j|b}, \qquad o^E = \sum_{b=1}^{B}\sum_{j=1}^{M} a^E_{b,j} v_{b,j}. \tag{7}$$

Within block b, a position is active exactly when $d_{b,j} < \delta_b$; hence, δ_b controls the number of active weights. The thresholds are trained jointly with the other network parameters. For $N = BM$, **ERTL reduces the exponential count from $\mathcal{O}(N)$ to $\mathcal{O}(N/M)$ per query by applying exponentials only across block maxima**, while retaining nonzero mass in every block. With singleton remainders, the block count is $\lfloor N/M \rfloor + (N \bmod M)$; Appendix C.2 explains this case.

Block size and threshold. The block size controls where the approximation is introduced. At $M = 1$, every block is a singleton and ERTL equals softmax. At $M = N$, there is only one block and ERTL becomes globally normalized shifted ReLU. Larger full blocks save more exponentials, but allow greater variation in the effective masses R_b. We use $M = 128$ to match the kernel tile. The threshold controls both the discarded mass and the relative weights of surviving tokens; increasing it need not reduce the total approximation error. Section 3.2 separates these effects.

Preserving softmax structure. Related approaches, such as sigmoid- and ReLU-based alternatives, replace softmax's coupled exponential normalization with different pointwise mappings and normalization rules (Ramapuram et al., 2025; Wortsman et al., 2023; Shen et al., 2023). Instead, our ERTL function starts from the exact factorization in Eq. (4). Its inter-block factor remains an exponentially normalized distribution, while its intra-block factor is nonnegative, normalized, and monotone in the original scores among surviving tokens. It therefore preserves row stochasticity, cross-block competition, and within-block score ordering. Its departure from softmax is confined to two explicit changes: omitting R_b from the block mass and replacing the local exponential shape with a shifted linear one. These controlled changes make pattern drift analyzable and provide a structured initialization for low-cost adaptation.

Adapting pretrained softmax models. Adaptation adjusts the attention projections and block-wise thresholds to compensate for the change in normalization. The projection updates adjust score gaps and value representations, while the thresholds adjust the local weights. These parameters are optimized jointly under the task loss, without a separate learning stage. Most experiments use LoRA (Hu et al., 2022) and train the task-specific head while freezing the remaining backbone; SegFormer uses full-model fine-tuning. Section 4.1 gives the evaluation protocol.

3.2 Approximation Error

For $X \in \{S, E\}$, write $a^X = (a^X_{b,j})_{b,j}$, $\alpha^X = (\alpha^X_b)_b$, and $\beta^X_{\cdot|b} = (\beta^X_{j|b})_j$ for the full, inter-block, and block-b conditional distributions, respectively. For distributions x and y on the same support, let $\mathrm{TV}(x, y) = \frac{1}{2}\|x - y\|_1$. The elementary likelihood-ratio bound $\mathrm{TV}(p, q) \le \tanh(\Omega/4)$ holds whenever $\max_i \log(p_i/q_i) - \min_i \log(p_i/q_i) \le \Omega$. Adding and subtracting $\alpha^E_b \beta^S_{j|b}$ and applying the triangle inequality separates the total error into block selection and within-block approximation:

$$\mathrm{TV}(a^S, a^E) \le \mathrm{TV}(\alpha^S, \alpha^E) + \sum_b \alpha^E_b \, \mathrm{TV}(\beta^S_{\cdot|b}, \beta^E_{\cdot|b}). \tag{8}$$

The following results characterize the two terms and translate distribution error into output error. They concern the normalization functions in exact arithmetic; FP8 quantization and rounding are separate sources of error. Complete proofs appear in Appendix B, using the preliminaries in Appendix A.

Theorem 1 (Inter-block approximation error)**.** *Let $\overline{R} = \sum_b \alpha^E_b R_b$. Then*

$$\mathrm{TV}(\alpha^S, \alpha^E) = \frac{1}{2\overline{R}} \sum_b \alpha^E_b |R_b - \overline{R}|. \tag{9}$$

If $\Omega_R = \max_b \log R_b - \min_b \log R_b$, then

$$\mathrm{TV}(\alpha^S, \alpha^E) \le \tanh\left(\frac{\Omega_R}{4}\right) \le \frac{\sqrt{M} - 1}{\sqrt{M} + 1}. \tag{10}$$

In particular, the inter-block error is exactly zero when all R_b are equal.

Theorem 2 (Intra-block truncation and shape error)**.** *For block b, define the truncated set $\mathcal{T}_b = \{j : d_{b,j} \ge \delta_b\}$, its exact-softmax mass $\tau_b = \sum_{j \in \mathcal{T}_b} \beta^S_{j|b}$, and the surviving set $\mathcal{S}_b = [M] \setminus \mathcal{T}_b$. For $0 \le d < \delta_b$, let*

$$h_{\delta_b}(d) = -d - \log\left(1 - \frac{d}{\delta_b}\right),$$
$$\Omega_b = \max_{j \in \mathcal{S}_b} h_{\delta_b}(d_{b,j}) - \min_{j \in \mathcal{S}_b} h_{\delta_b}(d_{b,j}). \tag{11}$$

Then

$$\mathrm{TV}(\beta^S_{\cdot|b}, \beta^E_{\cdot|b}) \le \tau_b + (1 - \tau_b) \tanh\left(\frac{\Omega_b}{4}\right). \tag{12}$$

If $n_b = |\mathcal{T}_b|$, the discarded mass satisfies

$$\tau_b \le \frac{n_b e^{-\delta_b}}{1 + n_b e^{-\delta_b}} \le \frac{(M-1)e^{-\delta_b}}{1 + (M-1)e^{-\delta_b}}. \tag{13}$$

Theorem 3 (Attention-output error and tightness)**.** *Index tokens by $i = (b, j)$, with $a^X_i = a^X_{b,j}$ and $v_i = v_{b,j}$, and let $t = \mathrm{TV}(a^S, a^E)$. For $t > 0$, define $\Delta_i = a^S_i - a^E_i$ and*

$$\mu^+ = \frac{1}{t} \sum_{\Delta_i > 0} \Delta_i v_i, \qquad \mu^- = \frac{1}{t} \sum_{\Delta_i < 0} (-\Delta_i) v_i. \tag{14}$$

Then

$$o^S - o^E = t(\mu^+ - \mu^-),$$
$$\|o^S - o^E\|_2 = \mathrm{TV}(a^S, a^E) \|\mu^+ - \mu^-\|_2. \tag{15}$$

If $D_V = \max_{i,i'} \|v_i - v_{i'}\|_2$, then

$$\|o^S - o^E\|_2 \le D_V \, \mathrm{TV}(a^S, a^E) \le 2V_{\max} \, \mathrm{TV}(a^S, a^E), \tag{16}$$

where the last inequality assumes $\|v_i\|_2 \le V_{\max}$. The final bound is attainable.

Implications of the theorems. The inter-block error depends on variation in R_b, rather than on the block maxima alone: identical effective masses make the block allocation exact even when the maxima differ. Within each block, τ_b measures the mass removed by truncation, while Ω_b measures the distortion among surviving weights. The output identity further shows why a change in attention weights need not cause an equally large change in representation: the error also depends on the separation between μ^+ and μ^-. These quantities distinguish where approximation enters the computation and motivate the output-similarity measurements in Sec. 4.3.

3.3 Fused FP8 Kernel

Our ERTL function allows a streaming implementation because each block produces only a local output and one scalar logit. Specifically, after the first matrix multiplication computes $z_{b,j}$, the kernel obtains m_b and $d_{b,j}$, applies the blockwise threshold δ_b in Eq. (6), and forms

$$u_b = \sum_{j=1}^{M} \beta^E_{j|b} v_{b,j}. \tag{17}$$

Thus, the intra-block weights need not leave the tile, and neither the score matrix nor the attention matrix is materialized in HBM. We use $M = 128$, matching the key–value tile size so that local normalization stays within a tile.

Online inter-block aggregation. The remaining computation uses online softmax normalization (Milakov & Gimelshein, 2018) over the block maxima $\{m_b\}_{b=1}^{B}$. Let μ_b, ℓ_b, and O_b denote the running maximum, normalizer, and unnormalized output after block b. Starting from $\mu_1 = m_1$, $\ell_1 = 1$, and $O_1 = u_1$, the exact online recurrence for the ERTL function, for $b = 2, \ldots, B$, is

$$\begin{aligned} &\mu_b = \max(\mu_{b-1}, m_b), \\ &r_b = \exp(\mu_{b-1} - \mu_b), \qquad w_b = \exp(m_b - \mu_b), \\ &\ell_b = r_b \ell_{b-1} + w_b, \qquad O_b = r_b O_{b-1} + w_b u_b. \end{aligned} \tag{18}$$

Here r_b rescales the accumulated state from preceding blocks, whereas w_b is the unnormalized weight of the incoming block. The final result is $o^E = O_B/\ell_B$. Equation (18) evaluates only $\mathcal{O}(B)$ exponentials, stores $\mathcal{O}(D_v)$ running state per query, and is algebraically identical to Eqs. (5) and (7).

Lazy-O schedule. The vector rescaling $r_b O_{b-1}$ lies on the dependency path between consecutive tiles. FlashAttention-4 also avoids per-tile rescaling through a conditional update (Zadouri et al., 2026). Here, lazy-O represents the accumulated output as $O_b = g_b \widetilde{O}_b$, with a scalar pending scale g_b. With $g_1 = 1$ and $\widetilde{O}_1 = u_1$, Eq. (18) becomes

$$g_b = r_b g_{b-1}, \qquad \widetilde{O}_b = \widetilde{O}_{b-1} + \frac{w_b}{g_b} u_b. \tag{19}$$

The accumulated scale is materialized when numerical range requires rebasing and at finalization, where $o^E = g_B \widetilde{O}_B/\ell_B$. In the FP8 path, w_b/g_b is folded into the local value-product accumulation scale. Lazy-O preserves o^E in exact arithmetic while deferring vector-wide rescaling. This change of variables also applies to exact-softmax online accumulation; our ablation measures its contribution within the ERTL kernel.

Fused dataflow. Each streamed tile follows the same four stages: FP8 score MMA; tile-local reduction and shifted-ReLU normalization using δ_b; FP8 value MMA producing u_b; and the scalar online update with lazy-O. The score and weight tiles remain on chip across the two MMAs, and only $(\mu_b, \ell_b, g_b, \widetilde{O}_b)$ persist between tiles. This structure preserves dense attention and introduces no sequence-length-dependent global synchronization.

4 Experiments

The ERTL function is evaluated on DeiT classification models (Touvron et al., 2021), PVTv2 detection (Wang et al., 2022), and SegFormer segmentation (Xie et al., 2021). Separate kernel benchmarks measure FP8 throughput on an NVIDIA B200 (NVIDIA, 2026). Ablations examine the two-level formulation and lazy-O scheduling. All ERTL experiments use $M = 128$; model thresholds are initialized to 3.0, with one scalar per layer, head, and block (Appendix C.2).

Table 1: Accuracy before and after replacing softmax with the ERTL function. DeiT models perform ImageNet-1K classification, PVTv2-B0 performs COCO detection, and SegFormer-B0 performs ADE20K segmentation. DeiT and PVTv2-B0 use LoRA on the four attention projections; SegFormer-B0 is fully fine-tuned.

Model	Metric	Softmax	ERTL (0 ep.)	ERTL adapted	Epochs
DeiT-Tiny/16	Top-1↑	72.1	36.5	**72.2**	30
DeiT-Small/16	Top-1↑	**80.1**	69.6	79.7	30
DeiT-Base/16	Top-1↑	**81.4**	76.7	80.9	30
PVTv2-B0	AP^{box}↑	38.2	36.9	**38.7**	12
SegFormer-B0	mIoU↑	37.4	35.6	**37.9**	40

Table 2: DeiT-Tiny/16 ImageNet Top-1 accuracy (%) after replacing softmax in the same pretrained checkpoint and applying an identical LoRA budget. Gap@30 equals the 30-epoch accuracy minus the frozen softmax reference.

Method	0 ep.	10 ep.	20 ep.	30 ep.	Gap@30↑
Softmax reference	72.1	–	–	–	0.0
ERTL	36.5	**71.5**	**71.9**	**72.2**	**0.1**
Sigmoid (Ramapuram et al., 2025)	0.1	8.0	10.6	12.0	-60.1
ReLU (Wortsman et al., 2023)	0.1	68.0	68.8	69.2	-2.9

4.1 Model Accuracy and Adaptation

Evaluation protocol. Each alternative function replaces softmax in the same pretrained checkpoint and is evaluated before and after adaptation. DeiT and PVTv2-B0 use LoRA on the four attention projections, jointly training the function parameters and task-specific heads for 30 and 12 epochs while freezing the remaining backbone. SegFormer-B0 uses full-model fine-tuning for 40 epochs. Training uses AMP for DeiT and FP32 for PVTv2-B0 and SegFormer-B0 (Table 8). Within each comparison, the optimizer, learning rate, data order, augmentation, and budget are fixed. Inference for the reported ERTL model accuracies uses our fused FP8 attention kernel (Sec. 3.3); throughput is evaluated in Sec. 4.2.

Transfer across models and tasks. Table 1 evaluates the ERTL function on three DeiT classification backbones and two hierarchical ViTs for dense prediction: PVTv2-B0 on COCO detection and SegFormer-B0 on ADE20K segmentation. After adaptation, models using the ERTL function are within 0.5 percentage points of their exact-softmax counterparts on all three classification backbones: $+0.1$ on DeiT-Tiny, -0.4 on DeiT-Small, and -0.5 on DeiT-Base. The adapted ERTL models also exceed the corresponding softmax baselines by 0.5 AP on COCO and 0.5 mIoU on ADE20K. The recovery therefore extends across model sizes and beyond image classification.

Comparison with alternative functions. Table 2 compares how readily the ERTL function, sigmoid, and ReLU can be adapted from the same pretrained softmax checkpoint on DeiT-Tiny/16. In each case, only the attention normalization is replaced before applying the same LoRA configuration and training schedule; the comparison therefore measures recovery under a matched lightweight fine-tuning budget rather than the best achievable accuracy of separately optimized recipes. With the ERTL function, accuracy reaches 71.5% after 10 epochs and 72.2% after 30 epochs, recovering the 72.1% softmax reference. Under the same budget, ReLU reaches 69.2% and sigmoid reaches 12.0%. These results show that the softmax-derived two-level structure enables faster recovery under lightweight fine-tuning than sigmoid or ReLU.

Attention-pattern preservation. Table 3 measures how closely each adapted function follows the attention pattern of the original softmax model after the same 30-epoch schedule. The comparison reports Top-16 overlap between the highest-weight tokens and cosine similarity between attention outputs. The ERTL function achieves the highest value on both metrics, indicating closer preservation of the selected tokens and the resulting attention output.

Table 3: Attention-pattern preservation after 30 epochs on DeiT-Tiny/16. Metrics average over 1,000 ImageNet-1K images, layers, heads, and queries.

Method	Top-16 overlap↑	Output cosine↑
ERTL	**0.60**	**0.87**
ReLU	0.58	0.70
Sigmoid	0.12	0.01

Table 4: Resolution scalability on DeiT-Tiny/16. Columns report ImageNet Top-1 accuracy (%); retention is $100 \times \mathrm{ERTL}/\mathrm{Softmax}$.

Resolution	N	Softmax	ERTL	Retention↑
224	197	72.1	72.2	**100.1%**
640	1601	78.1	77.7	**99.5%**
1024	4097	78.6	78.2	**99.5%**

Resolution scalability. Table 4 evaluates the ERTL function on DeiT-Tiny/16 at resolutions 224, 640, and 1024. For a square input of resolution $R \times R$, the class token gives $N = (R/16)^2 + 1$, corresponding to 197, 1601, and 4097 tokens. Threshold initialization and block-index mapping are identical across resolutions; a higher resolution simply instantiates the corresponding additional block thresholds. With identical positional-embedding interpolation and adaptation schedules, models using the ERTL function retain at least 99.5% of the corresponding softmax accuracy as N increases by 20.8×; the absolute difference remains within 0.4 percentage points. These results indicate that the ERTL function maintains stable accuracy as input resolution increases.

4.2 Kernel Efficiency

Setup. An isolated non-causal attention kernel is benchmarked on an NVIDIA B200 (SM100) with CUDA 13.3, batch size 1, 16 heads, and head dimension 128. This fixed configuration differs from the model evaluation settings; reported speedups refer to standalone attention kernels. The comparison includes BF16 and FP8 exact-softmax FlashAttention-4 baselines and two FP8 ERTL kernel configurations: standard online output accumulation and lazy-O scheduling. The lazy-O configuration is used for the headline comparison, and per-head FP8 quantization is included in the timed region. TFLOPS counts both the $QK^\top$ and PV matrix multiplications, with each fused multiply-add counted as two operations.

Long-sequence kernel throughput. Table 5 reports the longest-sequence point in the kernel sweep. At N=1M, the FP8 ERTL kernel reaches 1831.8 TFLOPS, exceeding FP8 exact softmax by 162.3 TFLOPS (9.7%) and reaching 1.51× the BF16 throughput. The same-precision comparison measures the combined effect of ERTL normalization and its schedule; Table 7 separately isolates lazy-O within ERTL.

Table 5: Kernel throughput at N=1M, batch size 1, and head dimension 128 on B200. Speedup is relative to BF16 exact softmax.

Variant	Latency (ms)	Throughput (TFLOPS)	Speedup
BF16 exact softmax	7424.69	1213.1	1.00×
FP8 exact softmax	5395.22	1669.5	1.38×
FP8 ERTL + lazy-O	**4917.10**	**1831.8**	**1.51×**

Throughput across sequence lengths. At N=262,144 in Fig. 3, FP8 FlashAttention-4 already reaches 1689.2 TFLOPS on B200. ERTL increases this to 1934.7 TFLOPS, a 245.5 TFLOPS (14.5%) gain over a baseline optimized for Blackwell. ERTL is near parity with FP8 at 1K and faster from 2K onward. Its peak speedup over BF16 is 1.55× at N=524,288; BF16 remains faster below 16K.

4.3 Ablation Study

Two complementary ablations examine the ERTL function and its kernel implementation. On DeiT-Tiny/16, the function ablation separates the contributions of the inter- and intra-block components under the classification adaptation protocol in Sec. 4.1, while the kernel ablation quantifies the throughput gained from lazy-O using the setup in Sec. 4.2.

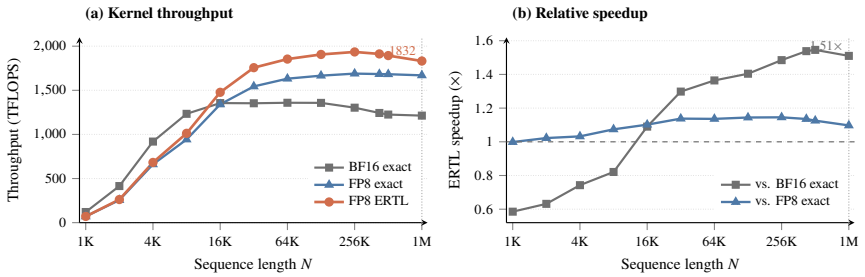


Figure 3: Kernel scaling on B200 with batch size 1. (a) Achieved throughput for BF16 exact softmax, FP8 exact softmax, and FP8 ERTL. (b) FP8 ERTL speedup relative to the two exact-softmax baselines.

Table 6: ERTL function ablation on DeiT-Tiny/16 (ImageNet Top-1, %). All ERTL variants use the same LoRA budget; exact softmax is the frozen reference.

Variant	0 ep.	Adapted
Exact softmax	**72.1**	72.1
Inter only	4.9	51.5
Intra only	17.8	69.0
Full ERTL	36.5	**72.2**

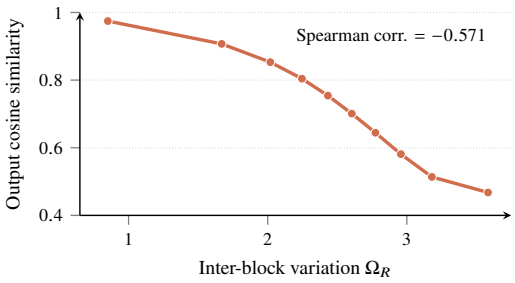


Figure 4: Theory–experiment consistency on frozen DeiT-Tiny/16: mean output cosine similarity $\cos(o^S, o^E)$ in ten equal-count Ω_R bins across layers, heads, and queries.

Two-level normalization. Table 6 evaluates the two components of the ERTL function under a matched adaptation budget. In the inter-only variant, a softmax is applied to the maximum score of each block, and the resulting block weight is broadcast to all positions within that block. The intra-only variant applies the ERTL shifted-ReLU function independently within each block and removes the outer softmax reweighting across blocks. Without adaptation, the intra-only variant retains more accuracy than the inter-only variant (17.8% vs. 4.9%). After adaptation, the inter-only and intra-only variants reach 51.5% and 69.0%, respectively, while the full ERTL function recovers 72.2%. These results show that intra-block weighting accounts for most of the accuracy recovery, while the outer softmax provides a complementary benefit when the two components are combined.

Consistency with the error analysis. Figure 4 examines the inter-block statistic Ω_R from Theorem 1. A frozen DeiT-Tiny/16 softmax checkpoint and its ERTL replacement (M=128, δ_b=3) are evaluated without adaptation on 1,000 fixed ImageNet-1K validation images. Output cosine similarity decreases as Ω_R grows: Spearman correlation is -0.571 over all 7,092,000 layer–head–query instances; the plot shows ten bin means. This trend agrees with the error analysis: variation in effective block mass helps diagnose output drift, although output error also depends on intra-block approximation and the value vectors.

Output-rescaling schedule. Table 7 compares kernels with and without lazy-O scheduling while holding the ERTL function and FP8 tile shapes fixed. Lazy-O adds 133.4–187.8 TFLOPS across the three configurations, with the largest gain at N=32K and batch size 1. The relative gains are 8.3%, 12.0%, and 7.5% for the three rows, respectively. The benefit appears at both tested batch sizes at 32K and persists at 436K. These results show that deferred output rescaling complements the removal of within-block exponentials: it reduces vector-wide updates along the tile dependency path while retaining the same ERTL normalization. Together, the local normalization and output-accumulation schedule form the fused kernel evaluated in Fig. 3.

Table 7: Lazy-O ablation on B200. Both kernels implement the same ERTL function and use identical FP8 tile shapes; Gain is the absolute throughput increase from lazy-O.

N	Batch	Throughput (TFLOPS)		
		w/o lazy-O	+ lazy-O	Gain
32K	8	1758.8	1904.9	**+146.1**
32K	1	1567.9	1755.7	**+187.8**
436K	1	1778.8	1912.2	**+133.4**

5 Related Work

Softmax alternatives. Sigmoid attention can match softmax with suitable bias and initialization (Ramapuram et al., 2025). ReLU alternatives require normalization or training adjustments (Wortsman et al., 2023; Shen et al., 2023). MixA combines quadratic ReLU attention with stabilized linear attention in selected ViT layers (Ahmed et al., 2025). ERTL retains exponential competition between block maxima and approximates within-block weights. This separates the allocation of attention across blocks from its distribution within a block. The two factors remain normalized, allowing their approximation errors to be analyzed separately and then combined into an output-error bound.

Low-precision GPU attention. FlashAttention-3 overlaps matrix multiplication, softmax, and data movement, and uses block quantization for FP8 attention (Shah et al., 2024). SageAttention introduces 8-bit attention quantization (Zhang et al., 2024). SageAttention2 combines INT4 score products and FP8 value products, using outlier smoothing and a two-level accumulation strategy to control quantization error (Zhang et al., 2025a). SageAttention3 extends this direction to microscaling FP4 on Blackwell and studies 8-bit attention for both forward and backward passes (Zhang et al., 2025b). FlashAttention-4 uses polynomial exponential evaluation and conditional output rescaling on Blackwell (Zadouri et al., 2026). These kernels improve the numerical formats and scheduling of softmax attention. ERTL changes the normalization function to remove within-block exponentials, and combines this reduction with tiled FP8 computation and deferred output rescaling.

Hardware softmax acceleration. Softermax uses base-2 arithmetic in a custom accelerator (Stevens et al., 2021), while FLASH-D hides online softmax division in dedicated hardware (Alexandridis et al., 2025). These designs target specialized microarchitectures. ERTL changes the normalization function and maps it to existing Tensor Core GPUs: within-block weights are formed using reductions and shifted-ReLU operations, while the retained inter-block softmax is evaluated online.

Non-standard attention. Linear attention uses feature maps to reduce asymptotic cost (Katharopoulos et al., 2020; Choromanski et al., 2021), and sparse attention restricts token interactions (Child et al., 2019; Zaheer et al., 2020). MonarchAttention projects softmax onto structured matrices for subquadratic, hardware-aware computation without additional training (Yaras et al., 2025). ERTL retains dense, quadratic score computation and reduces normalization work within streamed tiles, with adaptation used to recover pretrained accuracy. Its intra-block zero weights arise after score computation; they do not remove token pairs from the matrix multiplications. The computational saving is therefore in normalization rather than in the number of score products.

6 Conclusion

This work proposes the ERTL function, an exp-reduced two-level alternative to softmax that addresses the growing normalization bottleneck in attention. Notably, ERTL is agnostic to the numerical precision used for attention and can be applied to BF16, FP8, or even lower-precision attention. Our analysis characterizes the approximation error introduced by its two-level design, while an FP8 attention kernel that replaces softmax with ERTL demonstrates its efficiency on Blackwell GPUs. Experiments show that ERTL-based ViTs can recover the accuracy of their pretrained softmax counterparts through lightweight fine-tuning, and that replacing softmax with ERTL significantly improves the throughput of the FP8 attention kernel. These results establish structure-preserving approximation as a practical approach to alleviating the softmax bottleneck in ViTs.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript writing, sentence-level polishing, and LaTeX formatting. The authors are responsible for the final content of the paper.

REPRODUCIBILITY STATEMENT

Section 3.1 defines the ERTL function, and Section 3.3 specifies its streaming recurrence and lazy-O schedule. Appendices A and B provide the preliminary results and complete proofs. Appendix C records the model-adaptation protocols, training hyperparameters, evaluation subsets, and kernel benchmark settings used in the experiments.

REFERENCES

Sabbir Ahmed, Jingtao Li, Weiming Zhuang, Chen Chen, and Lingjuan Lyu. MixA: A mixed attention approach with stable lightweight linear attention to enhance efficiency of vision transformers at the edge. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21187–21196, 2025.

Kosmas Alexandridis, Vasileios Titopoulos, and Giorgos Dimitrakopoulos. FLASH-D: Flashattention with hidden softmax division. In *IEEE/ACM International Symposium on Low Power Electronics and Design*, 2025. URL https://arxiv.org/abs/2505.14201.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *Advances in Neural Information Processing Systems*, 35, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, volume 119, pp. 5156–5165, 2020.

Maxim Milakov and Natalia Gimelshein. Online normalizer calculation for softmax. *arXiv preprint arXiv:1805.02867*, 2018. URL https://arxiv.org/abs/1805.02867.

NVIDIA. NVIDIA HGX Platform. https://www.nvidia.com/en-us/data-center/hgx/, 2026. Accessed: 2026-07-26.

Jason Ramapuram, Federico Danieli, Eeshan Gunesh Dhekane, Floris Weers, Dan Busbridge, Pierre Ablin, Tatiana Likhomanenko, Jagrit Digani, Zijin Gu, Amitis Shidani, and Russell Webb. Theory, analysis, and best practices for sigmoid self-attention. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Zhdhg6n2OG.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. *arXiv preprint arXiv:2407.08608*, 2024.

Kai Shen, Junliang Guo, Xu Tan, Siliang Tang, Rui Wang, and Jiang Bian. A study on ReLU and softmax in transformer. *arXiv preprint arXiv:2302.06461*, 2023. doi: 10.48550/arXiv.2302.06461. URL https://arxiv.org/abs/2302.06461.

Jacob R. Stevens, Rangharajan Venkatesan, Steve Dai, Brucek Khailany, and Anand Raghunathan. Softermax: Hardware/software co-design of an efficient softmax for transformers. In *58th ACM/IEEE Design Automation Conference*, pp. 469–474, 2021. doi: 10.1109/DAC18074.2021.9586134.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357. PMLR, 2021. URL https://proceedings.mlr.press/v139/touvron21a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. PVT v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022. doi: 10.1007/s41095-022-0274-8.

Mitchell Wortsman, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Replacing softmax with ReLU in vision transformers. *arXiv preprint arXiv:2309.08586*, 2023. doi: 10.48550/arXiv.2309.08586. URL https://arxiv.org/abs/2309.08586.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, José M. Álvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://papers.neurips.cc/paper/2021/hash/64f1f27bf1b4ec22924fd0acb550c235-Abstract.html.

Can Yaras, Alec S. Xu, Pierre Abillama, Changwoo Lee, and Laura Balzano. MonarchAttention: Zero-shot conversion to fast, hardware-aware structured attention. In *Advances in Neural Information Processing Systems*, 2025.

Ted Zadouri, Markus Hoehnerbach, Jay Shah, Timmy Liu, Vijay Thakkar, and Tri Dao. FlashAttention-4: Algorithm and kernel pipelining co-design for asymmetric hardware scaling. *arXiv preprint arXiv:2603.05451*, 2026.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big Bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pp. 17283–17297, 2020.

Jintao Zhang, Jia Wei, Haofeng Huang, Pengle Zhang, Jun Zhu, and Jianfei Chen. SageAttention: Accurate 8-bit attention for plug-and-play inference acceleration. *arXiv preprint arXiv:2410.02367*, 2024. URL https://arxiv.org/abs/2410.02367.

Jintao Zhang, Haofeng Huang, Pengle Zhang, Jia Wei, Jun Zhu, and Jianfei Chen. SageAttention2: Efficient attention with thorough outlier smoothing and per-thread INT4 quantization. In *International Conference on Machine Learning*, 2025a.

Jintao Zhang, Jia Wei, Pengle Zhang, Xiaoming Xu, Haofeng Huang, Haoxu Wang, Kai Jiang, Jianfei Chen, and Jun Zhu. SageAttention3: Microscaling FP4 attention for inference and an exploration of 8-bit training. *arXiv preprint arXiv:2505.11594*, 2025b. URL https://arxiv.org/abs/2505.11594.

A Notation and Preliminary Results

For one query, partition the N attention scores into $B = N/M$ blocks, each containing M scores. When N is not divisible by M in implementation, every remaining token is treated as a singleton block; its intra-block weight is one, and it participates in the same outer softmax as the M-token blocks. For block b and position j, let $z_{b,j}$ denote the score, $m_b = \max_j z_{b,j}$ the block maximum, and $d_{b,j} = m_b - z_{b,j} \geq 0$ the gap from that maximum. Define the exact within-block exponential mass

$$R_b = \sum_{j=1}^{M} \exp(-d_{b,j}). \tag{A.1}$$

The exact-softmax and ERTL inter-block distributions are

$$\alpha_b^S = \frac{\exp(m_b)R_b}{\sum_{r=1}^{B} \exp(m_r)R_r}, \qquad \alpha_b^E = \frac{\exp(m_b)}{\sum_{r=1}^{B} \exp(m_r)}. \tag{A.2}$$

Their corresponding conditional distributions within block b are

$$\beta_{j|b}^S = \frac{\exp(-d_{b,j})}{R_b}, \qquad \beta_{j|b}^E = \frac{(\delta_b - d_{b,j})_+}{\sum_{r=1}^{M}(\delta_b - d_{b,r})_+}, \tag{A.3}$$

where $(x)_+ = \max(x, 0)$ and the trainable threshold δ_b is specific to block b. In implementation, each threshold is clipped after an optimizer update to keep it positive. The full attention distributions are $a_{b,j}^X = \alpha_b^X \beta_{j|b}^X$ for $X \in \{S, E\}$. For two distributions p and q on the same finite support, let

$$\mathrm{TV}(p, q) = \frac{1}{2} \sum_i |p_i - q_i|. \tag{A.4}$$

Lemma A.1 (Likelihood-ratio range)**.** *Suppose* $p_i, q_i > 0$ *and*

$$\max_i \log \frac{p_i}{q_i} - \min_i \log \frac{p_i}{q_i} \leq \Omega. \tag{A.5}$$

Then $\mathrm{TV}(p, q) \leq \tanh(\Omega/4)$.

Proof. Let $r_i = p_i/q_i$, $a = \min_i r_i$, and $b = \max_i r_i$. Since $\sum_i q_i r_i = 1$, either $p = q$ or $0 < a < 1 < b$. Convexity of $|r - 1|$ on $[a, b]$ gives

$$\mathrm{TV}(p, q) = \frac{1}{2}\mathbb{E}_q|r - 1| \leq \frac{(1 - a)(b - 1)}{b - a}. \tag{A.6}$$

Write $k = b/a$. For fixed k, the right-hand side equals

$$\frac{(1 - a)(ka - 1)}{a(k - 1)} \tag{A.7}$$

and is maximized at $a = k^{-1/2}$. Its maximum is

$$\frac{\sqrt{k} - 1}{\sqrt{k} + 1} = \tanh\left(\frac{\log k}{4}\right). \tag{A.8}$$

Because $\log k \leq \Omega$ and $\tanh$ is increasing, the result follows. □

Lemma A.2 (Hierarchical total-variation decomposition)**.** *For* $a_{b,j}^X = \alpha_b^X \beta_{j|b}^X$,

$$\mathrm{TV}(a^S, a^E) \leq \mathrm{TV}(\alpha^S, \alpha^E) + \sum_b \alpha_b^E \, \mathrm{TV}(\beta_{\cdot|b}^S, \beta_{\cdot|b}^E). \tag{A.9}$$

Proof. Adding and subtracting $\alpha_b^E \beta_{j|b}^S$ inside each absolute value and applying the triangle inequality yields

$$2\,\mathrm{TV}(a^S, a^E) = \sum_{b,j} \left| \alpha_b^S \beta_{j|b}^S - \alpha_b^E \beta_{j|b}^E \right| \tag{A.10}$$

$$\leq \sum_{b,j} |\alpha_b^S - \alpha_b^E| \beta_{j|b}^S + \sum_{b,j} \alpha_b^E |\beta_{j|b}^S - \beta_{j|b}^E|. \tag{A.11}$$

Summing each conditional distribution over j gives Eq. (A.9). □

B Proofs of the Main Theorems

B.1 Inter-Block Approximation Error

Theorem B.1 (Restatement of Theorem 1)**.** *Let $\overline{R} = \sum_b \alpha_b^E R_b$. Then*

$$\mathrm{TV}(\alpha^S, \alpha^E) = \frac{1}{2\overline{R}} \sum_b \alpha_b^E |R_b - \overline{R}|. \tag{B.1}$$

If $\Omega_R = \max_b \log R_b - \min_b \log R_b$, then

$$\mathrm{TV}(\alpha^S, \alpha^E) \le \tanh\left(\frac{\Omega_R}{4}\right) \le \frac{\sqrt{M}-1}{\sqrt{M}+1}. \tag{B.2}$$

In particular, the inter-block error is zero when all R_b are equal.

Proof. Let $Z_E = \sum_r \exp(m_r)$. The exact-softmax normalizer can be written as

$$Z_S = \sum_r \exp(m_r) R_r = Z_E \sum_r \alpha_r^E R_r = Z_E \overline{R}. \tag{B.3}$$

Consequently,

$$\alpha_b^S = \frac{\exp(m_b) R_b}{Z_E \overline{R}} = \alpha_b^E \frac{R_b}{\overline{R}}. \tag{B.4}$$

Substitution into the definition of total variation gives the claimed exact identity. Moreover,

$$\log \frac{\alpha_b^S}{\alpha_b^E} = \log R_b - \log \overline{R}, \tag{B.5}$$

so the range of the log likelihood ratio is exactly Ω_R. Lemma A.1 proves the first upper bound. Since every block contains at least one maximizer with $d_{b,j} = 0$,

$$1 \le R_b = \sum_j \exp(-d_{b,j}) \le M. \tag{B.6}$$

Thus $\Omega_R \le \log M$, and

$$\tanh\left(\frac{\log M}{4}\right) = \frac{\sqrt{M}-1}{\sqrt{M}+1}. \tag{B.7}$$

Finally, if R_b is constant across blocks, then $R_b = \overline{R}$ and the exact identity is zero. □

B.2 Intra-Block Truncation and Shape Error

Theorem B.2 (Restatement of Theorem 2)**.** *For block b, define the truncated set $\mathcal{T}_b = \{j : d_{b,j} \ge \delta_b\}$, its exact-softmax mass $\tau_b = \sum_{j \in \mathcal{T}_b} \beta_{j|b}^S$, and the surviving set $\mathcal{S}_b = [M] \setminus \mathcal{T}_b$. For $0 \le d < \delta_b$, let*

$$h_{\delta_b}(d) = -d - \log\left(1 - \frac{d}{\delta_b}\right), \tag{B.8}$$

$$\Omega_b = \max_{j \in \mathcal{S}_b} h_{\delta_b}(d_{b,j}) - \min_{j \in \mathcal{S}_b} h_{\delta_b}(d_{b,j}). \tag{B.9}$$

Then

$$\mathrm{TV}(\beta_{\cdot|b}^S, \beta_{\cdot|b}^E) \le \tau_b + (1 - \tau_b) \tanh\left(\frac{\Omega_b}{4}\right). \tag{B.10}$$

If $n_b = |\mathcal{T}_b|$, then

$$\tau_b \le \frac{n_b e^{-\delta_b}}{1 + n_b e^{-\delta_b}} \le \frac{(M-1)e^{-\delta_b}}{1 + (M-1)e^{-\delta_b}}. \tag{B.11}$$

Proof. The maximizer of each block has gap zero and therefore belongs to $\mathcal{S}_b$ because the implemented clipping keeps the threshold positive; hence the surviving set and the ERTL normalizer are nonempty. Condition exact softmax on this set:

$$\widetilde{\beta}^S_{j|b} = \frac{\beta^S_{j|b}}{1-\tau_b}, \qquad j \in \mathcal{S}_b. \tag{B.12}$$

For surviving positions, constants independent of j cancel from the range of the log likelihood ratio, giving

$$\log \frac{\widetilde{\beta}^S_{j|b}}{\beta^E_{j|b}} = -d_{b,j} - \log(\delta_b - d_{b,j}) + C_b \tag{B.13}$$

$$= h_{\delta_b}(d_{b,j}) + C'_b, \tag{B.14}$$

where C_b and C'_b depend only on the block. The range of this ratio is therefore Ω_b, and Lemma A.1 yields

$$\mathrm{TV}(\widetilde{\beta}^S_{\cdot|b}, \beta^E_{\cdot|b}) \le \tanh\left(\frac{\Omega_b}{4}\right). \tag{B.15}$$

The ERTL distribution assigns zero mass to $\mathcal{T}_b$. On $\mathcal{S}_b$, add and subtract $(1-\tau_b)\beta^E_{j|b}$. The triangle inequality then gives

$$2\,\mathrm{TV}(\beta^S_{\cdot|b}, \beta^E_{\cdot|b}) = \tau_b + \sum_{j\in\mathcal{S}_b} \left|(1-\tau_b)\widetilde{\beta}^S_{j|b} - \beta^E_{j|b}\right| \tag{B.16}$$

$$\le 2\tau_b + 2(1-\tau_b)\mathrm{TV}(\widetilde{\beta}^S_{\cdot|b}, \beta^E_{\cdot|b}), \tag{B.17}$$

which proves the first result.

For the discarded-mass bound, define $A_b = \sum_{j\in\mathcal{T}_b} \exp(-d_{b,j})$. Every truncated position satisfies $\exp(-d_{b,j}) \le e^{-\delta_b}$, so $A_b \le n_b e^{-\delta_b}$. The surviving maximum contributes one to R_b, and hence

$$\tau_b = \frac{A_b}{R_b} \le \frac{A_b}{1+A_b} \le \frac{n_b e^{-\delta_b}}{1+n_b e^{-\delta_b}}. \tag{B.18}$$

Because the maximum is never truncated, $n_b \le M-1$, completing the proof. □

B.3 Attention-Output Error and Tightness

Theorem B.3 (Restatement of Theorem 3)**.** *Index tokens by $i=(b,j)$ and let $t = \mathrm{TV}(a^S, a^E)$. For $t>0$, define $\Delta_i = a^S_i - a^E_i$ and*

$$\mu^+ = \frac{1}{t}\sum_{\Delta_i>0} \Delta_i v_i, \qquad \mu^- = \frac{1}{t}\sum_{\Delta_i<0} (-\Delta_i) v_i. \tag{B.19}$$

Then

$$o^S - o^E = t(\mu^+ - \mu^-) \tag{B.20}$$

and therefore

$$\|o^S - o^E\|_2 = \mathrm{TV}(a^S, a^E)\|\mu^+ - \mu^-\|_2. \tag{B.21}$$

If $D_V = \max_{i,i'} \|v_i - v_{i'}\|_2$, then

$$\|o^S - o^E\|_2 \le D_V\,\mathrm{TV}(a^S, a^E) \le 2V_{\max}\,\mathrm{TV}(a^S, a^E), \tag{B.22}$$

where the last inequality assumes $\|v_i\|_2 \le V_{\max}$. The final bound is attainable.

Proof. Because both attention vectors are distributions, $\sum_i \Delta_i = 0$. Their positive and negative parts therefore have equal mass, and the definition of total variation gives

$$\sum_{\Delta_i>0} \Delta_i = \sum_{\Delta_i<0} (-\Delta_i) = t. \tag{B.23}$$

It follows directly that

$$o^S - o^E = \sum_i \Delta_i v_i \tag{B.24}$$

$$= \sum_{\Delta_i>0} \Delta_i v_i - \sum_{\Delta_i<0} (-\Delta_i) v_i \tag{B.25}$$

$$= t(\mu^+ - \mu^-), \tag{B.26}$$

which also proves the norm identity.

Both μ^+ and μ^- are convex combinations of value vectors. If $p_i = \Delta_i/t$ on the positive support and $q_j = (-\Delta_j)/t$ on the negative support, then

$$\mu^+ - \mu^- = \sum_{i,j} p_i q_j (v_i - v_j). \tag{B.27}$$

The triangle inequality yields

$$\|\mu^+ - \mu^-\|_2 \le \sum_{i,j} p_i q_j \|v_i - v_j\|_2 \le D_V. \tag{B.28}$$

If every $\|v_i\|_2 \le V_{\max}$, then $D_V \le 2V_{\max}$, proving both inequalities. To attain the final bound, choose a unit vector u, set $v_i = V_{\max} u$ wherever $\Delta_i > 0$, and set $v_i = -V_{\max} u$ wherever $\Delta_i < 0$. Then $\mu^+ - \mu^- = 2V_{\max} u$, so equality holds. □

C Experimental and Implementation Details

C.1 Model Adaptation

Each experiment begins from a pretrained exact-softmax checkpoint. In most cases, adaptation is attention-focused: the DeiT and PVTv2 experiments apply LoRA to the attention projections and also update their task-specific heads, while the remaining pretrained backbone parameters are frozen. The SegFormer experiment instead fine-tunes the full model. For the ERTL function, the blockwise thresholds $\{\delta_b\}_{b=1}^{B}$ are optimized jointly with these parameters without a separate objective or learning stage. The reported ERTL model accuracies are evaluated with the fused FP8 attention kernel; Table 8 lists the precisions used during training.

For all LoRA runs, the attention targets are W_Q, W_K, W_V, W_O (implemented as `qkv` and `proj`); PVTv2 applies them to backbone attention. The low-rank factors use rank $r = 8$, scaling $\alpha = 16$, and dropout 0.05. Matrix A is initialized with Kaiming uniform initialization and matrix B with zeros. Base biases remain frozen. The classification or detection head is trained in addition to LoRA and δ_b. ERTL, ReLU, and Sigmoid use the same LoRA configuration in matched comparisons.

The comparison among ERTL, ReLU, and Sigmoid on DeiT-Tiny/16 uses the same pretrained checkpoint, LoRA budget, optimizer, data order, augmentation, and 30-epoch schedule for every replacement. Accuracy is recorded before adaptation and after 10, 20, and 30 epochs. Attention-pattern preservation is evaluated after epoch 30 on a fixed set of 1,000 ImageNet-1K validation images, with metrics averaged across images, layers, heads, and queries. The resolution study uses inputs of 224^2, 640^2, and 1024^2 pixels, corresponding to 197, 1,601, and 4,097 tokens for a patch size of 16, and applies the same positional-embedding interpolation and adaptation protocol at every resolution.

For the theory-validation experiment, the frozen DeiT-Tiny/16 exact-softmax checkpoint is evaluated without adaptation together with its ERTL replacement. Both functions use the same query, key, and value tensors; ERTL uses $M = 128$ and $\delta_b = 3$ for every block. The fixed evaluation subset contains 1,000 ImageNet-1K validation images, one per class, selected with seed 42. The resulting 7,092,000 layer–head–query instances are divided into ten equal-count bins by Ω_R. The theory-validation figure in the main paper reports the mean output cosine similarity in each bin; the Pearson and Spearman correlations between Ω_R and $1 - \cos(o^S, o^E)$ over all instances are 0.504 and 0.571, respectively.

Table 8: Training configurations. B_g denotes global batch size; LS denotes label smoothing. A dash indicates that no separate value was configured.

Setting	DeiT-Tiny	DeiT-Small	DeiT-Base	PVTv2-B0	SegFormer-B0
Dataset	ImageNet-1K	ImageNet-1K	ImageNet-1K	COCO	ADE20K
Adaptation	LoRA	LoRA	LoRA	LoRA	Full FT
Additional head	Classifier	Classifier	Classifier	RPN/RoI	Decode
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Configured LR	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	1.5×10^{-5}
LR scaling	$B_g/512$	$B_g/512$	$B_g/512$	–	$B_g/16$
Weight decay	0.05	0.05	0.05	0.05	0.01
AdamW β	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.999)$
Schedule	Cosine	Cosine	Cosine	Step (8, 11)	Polynomial
Warmup	3 ep.; 10^{-6}	3 ep.; 10^{-6}	3 ep.; 10^{-6}	500 it.; 5×10^{-7}	5 ep.; 1.2×10^{-5}
Minimum LR	10^{-6}	10^{-6}	10^{-6}	–	0
Batch (per GPU/global)	128/1024	128/1024	128/1024	8/64	8/64
Number of GPUs	8	8	8	8	8
Epochs	30	30	30	12	40
Gradient clipping	1.0	1.0	1.0	1.0	1.0
Loss	CE+LS (0.1)	CE+LS (0.1)	CE+LS (0.1)	CE+L1+mask	CE
Input resolution	224^2	224^2	224^2	1333×800	512^2
Precision	AMP	AMP	AMP	FP32	FP32
Random seed	42	42	42	42	42

C.2 ERTL and Kernel Configuration

The mathematical block size is fixed to $M = 128$. The thresholds are initialized to 3.0 and parameterized directly, with one scalar per layer, head, and block. They are updated by the same optimizer and learning rate as the other trainable parameters, without weight decay; after each optimizer update, they are clipped to remain positive. No softplus reparameterization is used. Threshold initialization and block-index mapping are identical across resolutions; a higher resolution simply instantiates the corresponding additional block thresholds. The kernel uses blockwise exponential allocation, shifted-ReLU intra-block normalization, and lazy-O aggregation; lazy-O leaves the ERTL output unchanged in exact arithmetic.

Choosing $M = 128$ aligns each full block with a key–value tile. For divisible sequence lengths, increasing M reduces the number of inter-block exponentials but loosens the worst-case bound in Theorem 1. At $M = 1$, ERTL equals softmax; at $M = N$, it becomes globally normalized shifted ReLU. Increasing δ_b reduces truncation but also changes the weights of surviving tokens, so the total error need not decrease monotonically. These observations explain the tradeoff; the reported experiments use one fixed block size and threshold initialization.

Remainder blocks. Under the singleton convention in Sec. 3.1, write $N = qM + r$ with $0 \le r < M$. The partition contains $q + r$ blocks, rather than $\lceil N/M \rceil$. For unequal block sizes, each intra-block sum runs over that block's tokens. Singleton blocks have $R_b = 1$ and $\beta^S_{1|b} = \beta^E_{1|b} = 1$, so they introduce no intra-block error. The bound $1 \le R_b \le M$ remains valid. All sequence lengths in the kernel sweep are divisible by 128; the model evaluations use the stated singleton convention.

Deferred rescaling beyond ERTL. The change of variables in Eq. (19) applies to an update $O_b = r_b O_{b-1} + c_b$ with $r_b > 0$: setting $g_b = r_b g_{b-1}$ gives $\widetilde{O}_b = \widetilde{O}_{b-1} + c_b/g_b$. For exact online softmax, $c_b = \sum_j \exp(z_{b,j} - \mu_b) v_{b,j}$ and the normalizer receives $\sum_j \exp(z_{b,j} - \mu_b)$. Thus, deferred rescaling can also be used with exact softmax, while ERTL separately removes within-block exponentials. A rebase materializes $g_b \widetilde{O}_b$ and resets g_b to one, preserving the represented output in exact arithmetic. This algebra does not specify a floating-point rebase threshold or establish a speedup for a modified exact-softmax kernel.

Table 9: Recorded kernel benchmark configuration.

Setting	Value
GPU	NVIDIA B200 (SM100)
CUDA	13.3
Kernel environment	`fp8attn_latest`
Attention mode	Non-causal
Batch sizes in sweep	1 and 8
Number of heads	16
Head dimension	128
ERTL block size M	128
Quantization	Per-head FP8 QKV, timed
FP8 format/scaling	Dynamic E4M3, per-head absmax
Score tile	128×128
SM100 contract	2-CTA, MMA 256×128, `q_stage=2`
Warmup/measured runs	20/100
Safe reciprocal path	$N \geq 131{,}072$

The sequence-length sweep evaluates

$$N \in \{1{,}024, 2{,}048, 4{,}096, 8{,}192, 16{,}384, 32{,}768, \\ 65{,}536, 131{,}072, 262{,}144, \\ 436{,}480, 524{,}288, 1{,}048{,}576\}. \tag{C.1}$$

The comparison includes BF16 exact-softmax FlashAttention-4, FP8 exact-softmax FlashAttention-4, FP8 ERTL with standard online output accumulation, and FP8 ERTL with lazy-O. Per-head FP8 quantization is included in the timed region. Reported TFLOPS count both $QK^\top$ and PV, with a fused multiply–add counted as two floating-point operations. The lazy-O ablation holds the ERTL function, FP8 quantization path, and tile shapes fixed so that the measured difference isolates deferred output rescaling.

D Detailed Softmax Bottleneck Analysis

A throughput-imbalance view. Let Q, K, and V denote the query, key, and value matrices, and let P denote the normalized attention-weight matrix. FlashAttention-style kernels process attention tile by tile. For a score tile containing B_r query rows and B_c key columns, with head dimension D, the $QK^\top$ and PV stages run on Tensor Cores, whereas softmax relies on reductions, scalar arithmetic, and SFU exponentials. Let T_{QK}, T_{sm}, and T_{PV} denote their respective per-tile latencies, and let T_{tile} denote steady-state tile latency. An idealized model with overlapping stages gives

$$T_{\mathrm{tile}} \approx \max(T_{QK}, T_{\mathrm{sm}}, T_{PV})\,. \tag{D.1}$$

The two MMA stages have comparable cost. This model treats normalization as the limiting stage when $T_{\mathrm{sm}} > T_{QK}$; it omits contention between the two MMAs for Tensor Core resources and other pipeline dependencies.

For each tile, $QK^\top$ performs approximately $2B_rB_cD$ floating-point operations, while online softmax performs $\kappa_{\mathrm{sm}} B_r B_c$ scalar-equivalent operations, where $\kappa_{\mathrm{sm}} \approx 5$ accounts for maximum, subtraction, exponential, accumulation, and normalization. Softmax therefore represents only $\kappa_{\mathrm{sm}}/(2D)$ of the arithmetic work of $QK^\top$. This small operation count, however, does not imply a small runtime. Let P_{TC} denote the effective Tensor-Core throughput and P_{SFU} the effective exponential-equivalent throughput of the softmax path. A roofline estimate gives

$$\frac{T_{\mathrm{sm}}}{T_{QK}} \approx \frac{\kappa_{\mathrm{sm}}}{2D} \frac{P_{\mathrm{TC}}}{P_{\mathrm{SFU}}} = \frac{\kappa_{\mathrm{sm}}}{2D} R_{\mathrm{eff}}, \tag{D.2}$$

where $R_{\mathrm{eff}} = P_{\mathrm{TC}}/P_{\mathrm{SFU}}$ measures the hardware imbalance. Softmax becomes slower than one MMA stage when

$$R_{\mathrm{eff}} > \frac{2D}{\kappa_{\mathrm{sm}}}. \tag{D.3}$$

For D=128 and κ_{sm}=5, the model's crossover is 51.2. The per-SM peak-rate calculation in FlashAttention-4 (Zadouri et al., 2026) gives 8,192 BF16 Tensor-Core operations and 16 MUFU exponentials per clock on B200. NVIDIA's dense Tensor-Core specifications give $2\times$ and $4\times$ the BF16 rate for FP8 and NVFP4, respectively (NVIDIA, 2026). The resulting hardware ratios are

$$R_{\mathrm{eff}}^{\mathrm{BF16}} = 512, \qquad R_{\mathrm{eff}}^{\mathrm{FP8}} = 1024, \qquad R_{\mathrm{eff}}^{\mathrm{NVFP4}} = 2048. \tag{D.4}$$

These peak-rate ratios are $10\times$, $20\times$, and $40\times$ the model's crossover. They show the widening gap between matrix and exponential throughput. Substituting peak SFU throughput for the mixed normalization path is only a heuristic: maxima, sums, subtraction, and division use other instructions and resources. Thus, the calculation motivates reducing normalization work but does not establish the bottleneck of a particular fused kernel. The measured runtime fractions in Figs. 1 and 2 provide evidence for the configurations shown.

Requirements for a practical softmax alternative. Reducing the arithmetic cost alone is insufficient. A replacement suitable for low-precision fused attention should satisfy three requirements:

1. **Sufficient exponential reduction.** Exponential evaluation is a major source of softmax latency and is inefficient on modern GPUs. A practical alternative need only reduce—not eliminate—exponentials, since their sharpening effect benefits model accuracy. As indicated by Eq. (D.1), reducing their cost until normalization is no longer the attention bottleneck is sufficient to achieve meaningful end-to-end attention speedup.
2. **Accuracy parity.** After replacement, the network should attain task accuracy comparable to its softmax counterpart; acceleration is not useful if it comes with a material loss in model quality.
3. **Low adaptation cost.** The training required to recover that accuracy should be as small as possible. A substitute that requires pretraining from scratch or extensive retraining is costly to adopt across existing models; recovery through brief fine-tuning makes the replacement practical and enables broad deployment.

These requirements motivate our ERTL function developed in Sec. 3.1.

D.1 How ERTL Meets the Design Requirements

ERTL addresses the criteria in Sec. 2 at the function and kernel levels. For $N = BM$, it reduces exponential work from $\mathcal{O}(N)$ to $\mathcal{O}(N/M)$ per query; Sec. 4.2 measures the resulting implementation's long-sequence throughput. The analysis in Sec. 3.2 separates the sources of approximation error and bounds their effect on attention outputs. The model experiments in Sec. 4.1 show near-baseline accuracy after adaptation, using LoRA for DeiT and PVTv2 and full-model fine-tuning for SegFormer. This empirical recovery supports the design; it is not implied by the error bounds alone.