2025 Design, Automation & Test in Europe Conference (DATE 2025)

SoftEx: a Low Power and Flexible Softmax Accelerator with Fast Approximate Exponentiation

Andrea Belano*‡, Yvan Tortorella*, Angelo Garofalo*†, Luca Benini*†, Davide Rossi*, Francesco Conti*
*University of Bologna, Bologna, Italy †ETH Zurich, Zurich, Switzerland ‡University of Pavia, Pavia, Italy

***Abstract*—Transformer-based models excel in NLP, vision, and audio processing, but the softmax operator can be a performance bottleneck, especially with optimized matrix-multiplication hardware. We introduce SoftEx, a parametric accelerator for BF16 softmax, using approximate exponentiation (<0.14% relative error) to boost softmax calculation. Integrated into a 12nm octa-core RISC-V cluster together with a matrix-multiplication systolic array, SoftEx reduces time and energy for attention probability computation by up to 10.8x and 26.8x, boosting MobileBERT throughput by 2.17x to 324 GOPS or 1.30 TOPS/W.**

***Index Terms*—neural network acceleration, softmax, transformers, approximate computing, exponential function**

I. Introduction

Transformers [1] have gained significant attention due to their groundbreaking performance in fields like natural language processing [2], computer vision [3], and audio processing [4]. The attention mechanism, a key component of Transformers, involves multiple matrix-matrix multiplications and softmax operations. Softmax introduces two main challenges: it is not an element-wise operation and relies on the exponential function. These issues make softmax a major bottleneck, especially when linear operations are highly accelerated. Approximations like Schraudolph's method [5] mitigate the bottleneck but often sacrifice accuracy.

This work contributes to enabling floating-point Transformers on embedded devices without requiring post-training modifications such as quantization or fine-tuning by introducing: *1)* An Hardware-friendly exp approximation based on Schraudolph's method, reducing mean relative error by 13× and maximum relative error by 3.7×; *2)* A parametric hardware accelerator for BF16 softmax, achieving up to 10.8× speedup and 26.8× energy reduction compared to 8 RISC-V cores using Schraudolph's method; *3)* A fully placed-and-routed cluster optimized for BF16 Transformers, delivering 324 GOPS peak performance or 1.30 TOPS/W efficiency in GlobalFoundries 12nm technology.

II. Exponential Function Approximation

Schraudolph's method, which we will refer to as $\mathrm{exps}(\cdot)$, is based on the following approximation: we move to base 2 to simplify the exponentiation, defining $x' = x \cdot \frac{1}{\log 2}$; then,

$$\exp(x) = 2^{x'} \approx 2^{\mathrm{int}(x')} \cdot \left(1 + \mathrm{frac}(x')\right) \doteq \mathrm{exps}(x) \qquad (1)$$

For details and proof, we defer to the original paper [5] or our open-source implementation[1].

We propose to replace $\mathrm{frac}(x')$ with a piecewise second-order polynomial, $P(\mathrm{frac}(x'))$. For $x \in [0, 0.5)$, we approximate $2^x - 1$ using $P(x) = \alpha x(x + \gamma_1)$, derived from a tangent line at $x = 0$ and a parabola centered at $x = 0$. For $x \in [0.5, 1)$, we use $P(x) = \mathrm{not}(\beta, \mathrm{not}(x) \cdot (x + \gamma_2))$, derived similarly but with a tangent line at $x = 1$ and a parabola centered at $x = 1$. The optimal paramteres, $\alpha = 0.21875$, $\beta = 0.4375$, $\gamma_1 = 3.296875$, and $\gamma_2 = 2.171875$, are calculated with a Montecarlo procedure.

This work was financially supported in part by the Chips-IT Foundation, in part by the ChipsJU project ISOLDE (g.a. 101112274), and in part by the Spoke 1 on Future HPC of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Mission 4 - Next Generation EU.

[1]https://github.com/belanoa/softex/tree/master/rtl

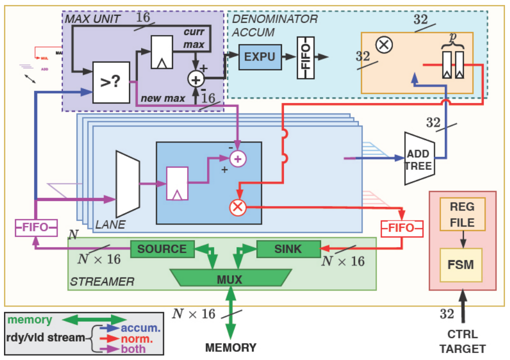

Fig. 1. A detailed view of SoftEx. Paths used in the accumulation step are highlighted in blue, those used in the normalization step are highlighted in red, and the ones used in both steps are highlighted in purple.

III. Architecture

A. SoftEx softmax accelerator

We introduce SoftEx, a parametric hardware accelerator for the softmax function targeting BF16 vectors. The high-level architecture of SoftEx is shown in Fig. 1. It comprises a programming interface with an internal control finite-state machine (*controller*), a set of specialized data movers to move data to/from memory (*streamer*), and a specialized datapath.

The datapath features a configurable number of lanes (N), each comprising a Multiplication and Addition Unit (MAU) and an Exponential Unit (EXPU) implementing the algorithm described in Section II. SoftEx also contains an Accumulator module containing a single pipelined FP32 Fused Multiply-Add (FMA) unit, which calculates first the value of the denominator and then its reciprocal. Softmax computation is divided into: *accumulation* to find the maximum and denominator, *inversion* to compute the reciprocal of the denominator, and *normalization* to produce the output probabilities.

B. Integration in a Transformer acceleration cluster

We integrate SoftEx into a compute cluster based on the PULP template[2], featuring 8 RISC-V `RV32IMFCXpulpnn` cores, a shared 32KiB instruction cache, and 256KiB of shared scratchpad memory. To accelerate the typical workloads in Transformers, we integrate a large Tensor Processing Engine

[2]https://pulp-platform.org

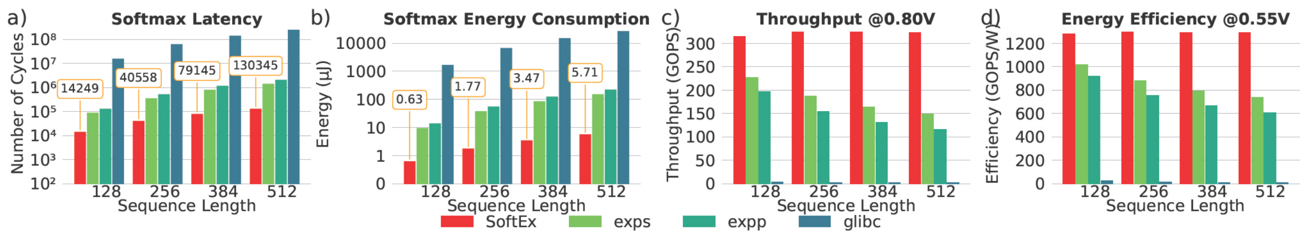

Fig. 2. Comparison of SoftEx's latency (a) and energy consumption (b) at 0.80V with different software implementations; System throughput @0.80V (c) and energy efficiency @0.55V (d) on MobileBERT's attention layer using SoftEx and different software implementations.

with 24×8 BF16 FMAs based on the RedMulE architecture [6], together with an instance of SoftEx employing $N = 16$ lanes.

IV. Experimental Results

A. Experimental Setup

The SoftEx-augmented cluster has been implemented in GlobalFoundries 12LP+ technology, targeting 700MHz frequency in worst-case conditions (SS process, 0.72V and 125°C) using Synopsys Design Compiler for synthesis and Cadence Innovus for placement, clock tree synthesis, and routing. The system's power consumption has been extracted under typical conditions (TT process and 25°C) using Synopsys Primetime with 100% annotated switching activity from a post-layout simulation at 0.80V and 1.12 GHz to maximize the throughput and one at 0.55V and 460 MHz to maximize the energy efficiency. We consider 1 MAC = 2 OPs.

B. Exponentiation Algorithm

We validated our expp approximation of the exponential function on 10^8 samples from a uniform distribution in the $[-88.7, 88.7]$. The results were compared with those obtained using Schraudolph's method (exps) and with glibc's implementation as the baseline. With respect to glibc's implementation, the proposed exponentiation algorithm achieves a mean relative error of 0.14% and a maximum relative error of 0.78%, 13× and 3.7× lower compared to Schraudolph's method.

C. SoftEx Area, Power and Performance

a) Area breakdown: SoftEx measures 0.033 mm^2, 2.75% of the cluster, which occupies 1.21 mm^2. The adder tree is the largest contributor to SoftEx's area, accounting for 30.1% of the total, followed by the MAUs (22.4%). With an overall contribution of 9.9%, the exponential units are the fourth largest contributor.

b) Power breakdown: Under full load, the average power consumption of the cluster is 276 mW (49.3 mW for SoftEx, with the rest dominated by the SRAM banks) at 0.80V or 54.4 mW at 0.55V (8.19 mW for SoftEx). Inside SoftEx, the MAUs dominate the power consumption contributing 24.9%, while the adder tree accounts for 23.9% of the total. The exponential units contribute 8.4% of SoftEx's total power.

c) Performance analysis: We benchmarked SoftEx using activations from MobileBERT's attention layer with different sequence lengths, comparing it to three parallel software implementations using different exponentiation algorithms: glibc's implementation, Schraudolph's method (exps), and the algorithm described in Section II (expp). As shown in Fig. 2, with a sequence length of 128, SoftEx computes attention probabilities 6.2× faster and uses 15.3× less energy than the second best method (exps). At a sequence length of 512, the gap widens to 10.8× speedup and 26.8× less energy.

D. Cluster performance on MobileBERT

We tested the performance of the system on MobileBERT's attention layer and show the results of this experiment in Fig 2.

When performing the attention layer with SoftEx, at 0.8V the cluster achieves a compound throughput of up to 324 GOPS (87% of the theoretical peak of purely linear operations). All software implementations result in a substantial throughput loss (>2.17× slowdown for larger sequence sizes), even when targeting the fastest and least precise exps.In terms of energy efficiency, at 0.55V the system achieves up to 1.30 TOPS/W, improving this value by 20.5-75.4% with respect to the eight RISC-V cores running Schraudolph exps.

Moreover, we tested the proposed cluster on the 24 encoder layers of MobileBERT with a sequence length of 512. Under the assumption of sufficient memory bandwidth on the external memory, the cluster achieves an average throughput of up to 297 GOPS (80% of the theoretical peak), with a total latency of 152 ms.

V. Conclusion

We presented SoftEx, an accelerator of the softmax function of BFloat16 vectors, achieving up to 10.8× speedup and 26.8× less energy compared to a software implementation running on 8 RISC-V cores. Integrated into a PULP cluster and coupled with a 24×8 Tensor Processing Engine, SoftEx makes the system achieve up to 324 GOPS at 0.80V or an energy efficiency of up to 1.30 TOPS/W at 0.55V on MobileBERT.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
[2] H. Touvron, T. Lavril *et al.*, "LLaMA: Open and Efficient Foundation Language Models," no. arXiv:2302.13971, Feb. 2023.
[3] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.
[4] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech 2021*, 2021, pp. 571–575.
[5] N. N. Schraudolph, "A Fast, Compact Approximation of the Exponential Function," *Neural Computation*, vol. 11, no. 4, pp. 853–862, May 1999.
[6] Y. Tortorella, L. Bertaccini, L. Benini, D. Rossi, and F. Conti, "RedMule: A mixed-precision matrix–matrix operation engine for flexible and energy-efficient on-chip linear algebra and TinyML training acceleration," *Future Generation Computer Systems*, vol. 149, pp. 122–135, Dec. 2023.