Genetic Quantization-Aware Approximation for Non-Linear Operations in Transformers

Pingcheng Dong[1,2,*], Yonghao Tan[1,2,*], Dong Zhang[1,2], Tianwei Ni[3], Xuejiao Liu[2], Yu Liu[2], Peng Luo[2], Luhong Liang[2], Shih-Yang Liu[1], Xijie Huang[1], Huaiyu Zhu[3], Yun Pan[3], Fengwei An[4], Kwang-Ting Cheng[1,2†]

[1]The Hong Kong University of Science and Technology, Hong Kong SAR. [2]AI Chip Center for Emerging Smart Systems (ACCESS), Hong Kong SAR. [3]Zhejiang University, China. [4]Southern University of Science and Technology, China.

{pdongaa, ytanaz}@connect.ust.hk, dongz@ust.hk, nitianwei@zju.edu.cn, {xuejiaoliu, albertliu, pengluo, luhong}@ust.hk, {sliuau, xhuangbs}@connect.ust.hk, {zhuhuaiyu, panyun}@zju.edu.cn, anfw@sustech.edu.cn, timcheng@ust.hk

ABSTRACT

Non-linear functions are prevalent in Transformers and their lightweight variants, incurring substantial and frequently underestimated hardware costs. Previous state-of-the-art works optimize these operations by piece-wise linear approximation and store the parameters in look-up tables (LUT), but most of them require unfriendly high-precision arithmetics such as FP/INT 32 and lack consideration of integer-only INT quantization. This paper proposed a genetic LUT-Approximation algorithm namely *GQA-LUT* that can automatically determine the parameters with quantization awareness. The results demonstrate that *GQA-LUT* achieves negligible degradation on the challenging semantic segmentation task for both vanilla and linear Transformer models. Besides, proposed *GQA-LUT* enables the employment of INT8-based LUT-Approximation that achieves an area savings of 81.3~81.7% and a power reduction of 79.3~80.2% compared to the high-precision FP/INT 32 alternatives. Code is available at *https://github.com/PingchengDong/GQA-LUT*.

KEYWORDS

Non-linear function, quantization-aware training, integer-only arithmetic, Transformer, look-up table, genetic algorithm

ACM Reference Format:

Pingcheng Dong, Yonghao Tan, Dong Zhang, Tianwei Ni, Xuejiao Liu, Yu Liu, Peng Luo, Luhong Liang, Shih-Yang Liu, Xijie Huang, Huaiyu Zhu, Yun Pan, Fengwei An, Kwang-Ting Cheng. Genetic Quantization-Aware Approximation for Non-Linear Operations in Transformers. In *Proceedings of DAC 2024: 61st IEEE/ACM Automation Conference. (DAC'24).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3657314

1 INTRODUCTION

The advent of the Transformer-based neural networks marked a new era for natural language processing [1] and computer vision tasks [2, 3]. The performance greatly benefits from the self-attention mechanism in Transformers, which could capture long-range dependencies well, but with a substantial overhead in computation and memory. Extensive research has been conducted to facilitate the deployment of Transformers on edge devices. Techniques like lightweight structure integrating convolution and linear attention [4, 5] emerge, while quantization [6–8] and run-time pruning [9] has become favored approaches to further reduced the hardware burden. However, the optimization of non-linear operations is frequently neglected in Transformer-based models which can be costly due to the frequent involvement of high-precision operations like 32-bit floating-point (FP32) or integer (INT32) arithmetic. As reported by [10], the inefficiency of non-linear operations seriously impedes the speed-up of Transformers at lots of hardware platforms.

Several prior works have endeavored to tackle the computational overheads posed by non-linear operations. Kim et al. [6] proposed to approximate the GELU, Softmax, and LayerNorm with quantization-aware 32-bit integer (INT32) arithmetic. Stevens et al. [10] customized a low-precision Softmax with a base replacement strategy. These methods can approximate and accelerate non-linear operations with minimal accuracy degradation but lack generality since each optimized operator contains distinct computational dataflow. To this end, a neural network-based general LUT-Approximation framework called *NN-LUT* is designed by [11]. However, the LUT contains numerous parameters that are independent of the input data range. To utilize fewer hardware resources and take into account the range of input, the above methods are simplified in [12] by factorizing floating-point data and performing single-entry LUT-Approximation on the mantissa in a small range. Although these excellent works progressively improve the generality and hardware-friendliness of the LUT-Approximation, they are still towards the computations in high-precision such as FP/INT32.

As for some customized neural network accelerators [9], the quantization scheme is generally in single-precision (e.g., INT8) or mixed-bit format [13, 14], and the inference may follow the dyadic arithmetic pipeline [15] to achieve integer-only computation. The I-BERT [6] is in the same category of integer-only quantization, but its management of the scaling factor and the input bit-width varies from operators. If high-precision LUT-Approximation is applied to INT8 inputs, it inevitably leads to resource wastage since the expressiveness of INT8 is much more limited compared to FP/INT32. Moreover, the method in [12] cannot be utilized directly due to it

*Both authors contributed equally.

†Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0601-1/24/06…$15.00
https://doi.org/10.1145/3649329.3657314

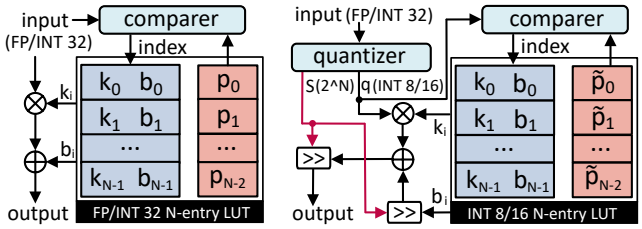


Figure 1: Taxonomy of LUT-Approximation: (a) FP/INT32 LUT storage pattern, (b) INT8/16 LUT storage pattern with quantization awareness.

is based on the floating-point decomposition strategy. The study from [12] also highlights a notable variation in the representation range across layers. However, the *NN-LUT* struggles to encapsulate this attribute unless subjected to calibration, a process that proves to be considerably time-consuming. This study begins with an in-depth analysis of the integer-only quantization scheme and LUT-Approximation. We identify the scaling factor in quantization as a crucial element that significantly impacts the precision of approximation parameters, noting that larger scaling factors lead to less accurate approximations. Building on this insight, we proposed a genetic quantization-aware algorithm to automatically determine the breakpoints in LUT-Approximation for non-linear functions. Termed as *GQA-LUT*, this method paves the way for efficient hardware design with compact resource utilization, capitalizing on low-bit integer arithmetic. To effectively manage larger scaling factors, we further implement an algorithm that images the fixed-point (FXP) conversion as a form of mutation to boost the accuracy. The key contributions of our work are summarized as follows:

- We delve into the interplay between the scaling factor and LUT parameters, and formulate a general quantization-aware LUT-Approximation computing flow.
- A genetic algorithm *GQA-LUT* for automatic approximation is proposed, to overcome constraints of existing quantization algorithms that fail to adjust parameters by scaling factors.
- A rounding mutation algorithm, which incorporates rounding error into the *GQA-LUT*, is proposed to solve the breakpoint deviation issue while handling intractable scales
- The area and power performance synthesized with TSMC 28-nm technology demonstrates that the INT8-based arithmetics achieve significant improvements, compared to the high-precision FP/INT32 units.

2 PRELIMINARIES AND RELATED WORK

2.1 Non-Linear Operations in Transformers

Non-linear operations are ubiquitous in vanilla transformer models [16]. The multi-head self-attention employs Softmax to capture the correlation in different dimensions, while GELU acts as the activation function within the feed-forward network (FFN), and LayerNorm is for the normalization. However, as the demand for edge computing increases, numerous lightweight transformer architectures have emerged, such as Softmax-free linear attention [17] and hybrid models that incorporate depthwise-separable convolution into early stages or FFNs [4]. The non-linear operations in lightweight transformers are diverse such as HSWISH and cosine. Generally, these non-linear operators require high-precision computations with disparate dataflow to ensure accuracy. Hence, it becomes imperative to explore methods for handling diverse non-linear operations within a unified and simple hardware engine.

2.2 LUT-based Approximation

Adopting piece-wise linear approximation (*pwl*) for various non-linear operations, coupled with parameter storage in LUT, has become a widely accepted strategy for enhancing hardware efficiency. This approach has been validated and endorsed by numerous recent and noteworthy studies [11, 12]. All these LUT-based *pwl* approximations can be formulated as follows:

$$pwl(x) := \begin{cases} k_0 x + b_0 & \text{if } x < p_0 \\ k_1 x + b_1 & \text{if } p_0 \leq x < p_1 \\ \vdots & \\ k_{N-1} x + b_{N-1} & \text{if } x \geqslant p_{N-2} \end{cases} \tag{1}$$

The $pwl(\cdot)$ function serves to approximate any nonlinear function $f(x)$ through a piece-wise linear approximation, comprising N entries $\{pwl_i(x)\}_{i=0:N-1}$, each represented as a first-order function like $pwl_i(x) = k_i x + b_i$. The breakpoints $\{p_i\}_{i=0:N-2}$ determine the position of each entry. The parameters for an N-Entry *pwl* are then stored in LUT as shown in Figure 1(a). The accuracy could be readily maintained owing to the high representation ability of FP/INT32-based LUT storage and input data. However, challenges may arise in scenarios involving low-bit quantization, a topic we delve into more in Section 3.

2.3 Integer-Only Quantization

Quantization is a popular technique to reduce the overhead of computation and memory after being deployed on chips since it can enable inference in a low-bit fashion. The quantization function is formulated as:

$$\tilde{x} = S \cdot q = S \cdot \left\lfloor \text{Clip}\left(\frac{x}{S}, Q_n, Q_p\right) \right\rceil \tag{2}$$

where $\lfloor \cdot \rceil$ represents the rounding function, S is the scaling factor that bridges quantized values q and high-precision (FP/INT32) inputs or weights x. In k-bit quantization, q is obtained by clipping $\frac{x}{S}$ within $[-2^{k-1}, 2^{k-1} - 1]$ for signed data or $[0, 2^k - 1]$ for unsigned data, with the range defined by lower and upper bounds Q_n and Q_p respectively. The q can be also dequantized to high-precision $\tilde{x}$ by multiplying with S. Commonly, S is determined using the min-max method [6] or learned via the straight through estimator (STE) [18], with the latter being more popular due to its proven effectiveness [7, 19]. In this work, we employ the LSQ [19] for quantization.

Integer-only quantization can be achieved by converting several scaling factors S into dyadic numbers [15] or by excluding S during inference [6]. However, the former method performs the non-linear function on the dequantized $\tilde{x}$ predominantly, which turns around to high-precision computation again. This stems from the fact that the non-linear function is linearly inseparable as discussed in [6], e.g., $exp(S \cdot q) \neq S \cdot exp(q)$. The approximation in [6] can be directly applied on q to achieve integer-only arithmetic, but lacks universality due to the diverse dataflow. As for the *pwl* method, it

Algorithm 1 Genetic Piece-Wise Linear Approximation.

Input: Non-linear function $f(\cdot)$, breakpoint size N_b, population size N_p, mutation function $M(\cdot)$, cross probability θ_c, mutant probability θ_m and θ_r, search range $[R_n, R_p]$, iteration size T, and the decimal bitwidth λ of slopes and intercepts.

Output: Sets of slopes $\mathcal{K}$, intercepts $\mathcal{B}$ and breakpoints $\mathcal{P}$

Initialize a breakpoint population $O = \{\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_{N_p-1}\}$, each $\mathcal{P}_{i\in[0,N_p-1]}$ is a set of N_b random FP32 values in $[R_n, R_p]$
for $n = 0$ to $T-1$ **do** ▷ T-round evolution
  for $\mathcal{P}_i$ in O_n **do** ▷ O_n : The n^{th} generation
    Initialize the Mean Squared Error (MSE): $E_i = 0$
    Create $pwl(\cdot)$ based on breakpoint set $\mathcal{P}_i$
    for $x = R_n$ to R_p with step 0.01 **do**
      Update $E_i = E_i + \frac{(pwl(x)-f(x))^2}{(R_p-R_n)/0.01}$
    end for
    $rand_c/rand_m \leftarrow$ random number in $[0, 1]$
    if $rand_c < \theta_c$ **then**
      Randomly select $\mathcal{P}_j$ from $O \setminus \{\mathcal{P}_i\}$
      Swap a random segment between $\mathcal{P}_j$ and $\mathcal{P}_i$
    end if
    if $rand_m < \theta_m$ **then**
      $M(\mathcal{P}_i, \theta_r)$ // Perform mutation function $M(\cdot)$
    end if
  end for
  $O_{n+1} \leftarrow$ Perform 3-size tournament selection on O_n
end for
$\mathcal{P} = \mathcal{P}^* \leftarrow$ Select the best individual from O_T
$\mathcal{K}^*, \mathcal{B}^* \leftarrow$ Derived from $\mathcal{P}^*$
$\mathcal{K} = \frac{\lfloor \mathcal{K}^* \cdot 2^\lambda \rceil}{2^\lambda}$, $\mathcal{B} = \frac{\lfloor \mathcal{B}^* \cdot 2^\lambda \rceil}{2^\lambda}$ ▷ Round to FXP based on λ

can bring separability to the non-linear function by its inherence of $pwl(S \cdot q) = S \cdot pwl(q)$, whereas current pwl-based works [11, 12] only focus on optimizing $pwl(S \cdot q)$, their arithmetics are still in FP/INT32 format.

3 METHOD

3.1 Quantization-Aware Approximation

Nothing that the pwl affords us a unique opportunity to conduct pwl directly on q with the separation of scaling factor S. Instead of following the dyadic pipeline, we enforce the scaling factor to be the power-of-two by rounding the logarithmic value of a learnable parameter α to its nearest integer. Then, the scaling factor can be derived by $S = 2^{\lfloor log_2^\alpha \rceil}$, and the STE is adopted to approximate the gradient in the non-differentiable round function. This power-of-two adaptation of S is aimed at streamling the quantization-aware approximation, particularly for the parameters $\{b_i\}_{i=0:N-1}$ and $\{p_i\}_{i=0:N-2}$, expressed as:

$$\tilde{b}_i = \frac{b_i}{S} = b_i \gg \lfloor log_2^\alpha \rceil,\ \tilde{p}_i = \left\lfloor \text{Clip}\left(\frac{p_i}{S}, Q_n, Q_p\right)\right\rceil \tag{3}$$

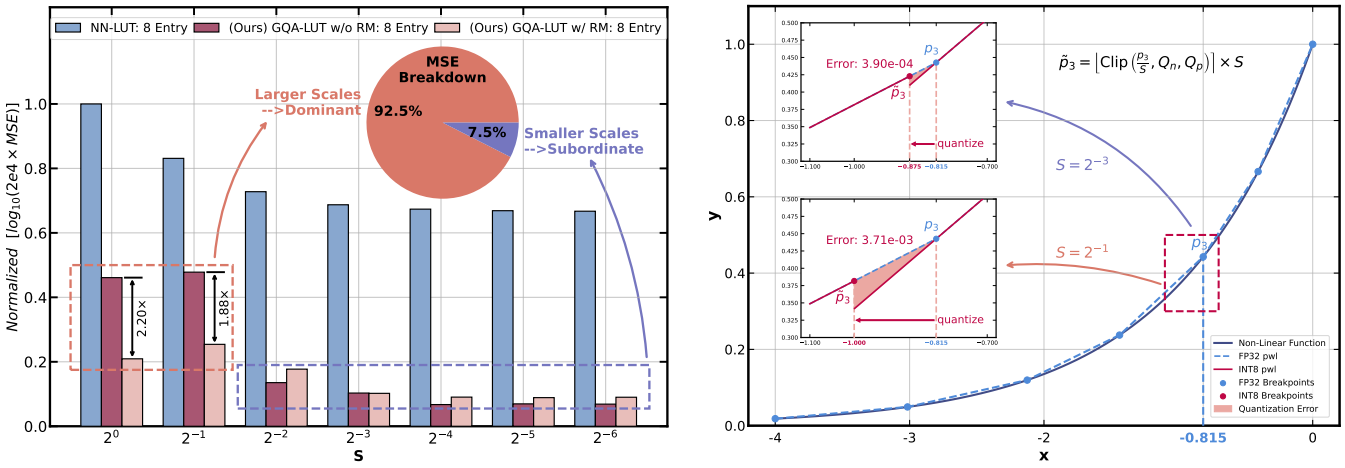
where the $\gg$ symbolizes the right shift operation, α represents the learnable parameter inherent to S, and $\tilde{b}_i$, $\tilde{p}_i$ denote the respective intercepts and breakpoints, scaled by the factor S for i ranging from 0 to $N-1$ and $N-2$. Benefiting from the learnable power-of-two transformation, the low-cost shift operator can replace the divider while ensuring accuracy. Since the quantized input q is an integer within $[Q_n, Q_p]$, the LUT only needs to store the original slopes w_i, intercepts b_i and the quantized breakpoints $\tilde{p}_i$, where the $\tilde{b}_i$ is computed run-time by a shifter and the slopes are kept the same. Building upon this, we present a quantization-aware LUT-based approximation method in Figure 1(b).

Nonetheless, specific non-linear operations like the divider (DIV) in Softmax and reciprocal of square root (RSQRT) in LayerNorm process intermediate fixed-point (FXP) outcomes directly, suspending the input quantization and consequently leading to extensive ranges of input data. To mitigate this, *NN-LUT* implements input scaling, amplifying small values by multiplying them with a substantial constant. The *RI-LUT* [12] factorizes the floating-point values into a mantissa and a power-of-two scale. In essence, both methodologies can be categorized as quantization, with the scaling factor being chosen manually. In this work, we propose a *Multi-Range Input Scaling* strategy that splits the input range outside the defined breakpoints interval $IR = [R_n, R_p]$ into Ns distinct sub-ranges $SR_i = [SR_{ni}, SR_{pi}]$, where $0 \le i < Ns-1$. The data in each sub-range is quantized to specific fixed-point values by a power-of-two scaling factor S_i' chosen manually. The pwl results will be multiplied with the S_i' and $\sqrt{S_i'}$ respectively for DIV and RSQRT to ensure the correct approximation. In contrast, the multi-range input scaling approach renders the scaling of breakpoints and intercepts unnecessary, as showcased in Equation (3), due to an intrinsic re-scaling process. Thus, the breakpoint and the intercepts could be rounded to a specific FXP format in this case.

3.2 Genetic Piece-Wise Linear Approximation

The performance of pwl critically hinges on the judicious selection of optimal breakpoints. To address this, the *NN-LUT* employs a novel neural network-based methodology. However, it necessitates substantial training data (100K) and presents challenges in managing the bitwidth of breakpoints during the training process. In this work, we propose to utilize a genetic algorithm namely *GQA-LUT*, an optimization technique [20] inspired by the process of natural selection, to implement the pwl as described in Algorithm 1. The main idea revolves around establishing a population containing several individuals which are the breakpoints $\mathcal{P}$ of different $pwl(\cdot)$. The individuals in the population stochastically undergo crossover and mutation to enhance diversity and explore new solutions. The crossover swaps segments between pairs, while mutation introduces a normal distribution of noise. Subsequently, the individual exhibiting the highest fitness, which corresponds to the lowest mean squared error (MSE) in approximating the non-linear function $f(\cdot)$, is identified and selected. The *GQA-LUT* mirrors the principles of natural selection, where MSE serves as the criterion for selection.

To facilitate storage and computation in a low-bit INT format, we initially employ a straightforward approach, wherein the optimal sets of slopes and intercepts are converted from FP32 to FXP values based on a decimal bitwidth λ. At the same time, the breakpoints are quantized as illustrated in Equation (3). The choice of λ is dictated by the search range $[R_n, R_p]$ of a specific $f(\cdot)$. We compare the accuracy of 8-entry *GQA-LUT* and *NN-LUT* under



(a) (b)

Figure 2: (a) Comparison of normalized MSE among *NN-LUT*, *GQA-LUT*, and *GQA-LUT* with *RM* strategy for GELU approximation using an 8-entry LUT, (b) breakpoint quantization analysis of GQA-LUT without *RM* for EXP under different scaling factors.

Table 1: Configurations of *GQA-LUT* with *RM* Strategy

Hyper-parameters*	GELU	HSWISH	EXP	DIV	RSQRT
$[R_n, R_p]$	$(-4, 4)$	$(-4, 4)$	$(-8, 0)$	$(0.5, 4)$	$(0.25, 4)$
$\boldsymbol{\theta_r}$	0.05	0.05	0.05	0	0
$[\boldsymbol{m_a}, \boldsymbol{m_b}]_8$	$[2, 6]$	$[0, 6]$	$[0, 6]$	-	-
$[\boldsymbol{m_a}, \boldsymbol{m_b}]_{16}$	$[0, 6]$	$[2, 6]$	$[0, 6]$	-	-
Data Size	0.8K	0.8K	0.8K	0.35K	0.36K

*: $N_b = 7$, $N_p = 50$, $\theta_c = 0.7$, $\theta_m = 0.2$, $T = 500$, and $\lambda = 5$ by default.

Table 2: Setup of *Multi-Range Input Scaling* for Wide-Range *DIV* and *RSQRT* Operations under INT8 *pwl*

Ops*	IR	SR_0/S'_0	SR_1/S'_1	SR_2/S'_2
DIV	$(0.5, 4)$	$[4, 32)/2^{-3}$	$[32, 256)/2^{-6}$	$[256, +\infty)/2^{-6}$
RSQRT	$(0.25, 4)$	$[4, 64)/2^{-4}$	$[64, 1024)/2^{-8}$	$[1024, +\infty)/2^{-12}$

*: The breakpoints are rounded to 8bit FXP with λ decimal bits.

identical FXP conversion methods for GELU approximation in Figure 2(a). The results reveal that the *GQA-LUT* exhibits superior accuracy to *NN-LUT* across all scaling factors. Upon analyzing the MSE breakdown of *GQA-LUT*, it becomes evident that the *GQA-LUT* predominantly struggles with large S. Specifically, the scaling factors below 2^{-2} contribute to over 90 percent of the total error, highlighting a significant limitation in handling large S.

3.3 Rounding Mutation

To investigate why approximation errors predominate at larger values of S, we closely examine the approximation curves of the exponential function (EXP) depicted in Figure 2(b). When a breakpoint p undergoes quantization to a specific integer value according to Equation (3), the resulting approximation error demonstrates variability across different scaling factors S. In particular, at larger S, the breakpoint is susceptible to significant shifts, leading to noticeable approximation offsets—a phenomenon we term as **breakpoint deviation**. In contrast, a smaller S tends to yield a minimal deviation, thereby mitigating the error. This observation highlights the

Algorithm 2 Rounding Mutation (RM) Algorithm.

Input: Breakpoint set $\mathcal{P}$, entry size $\boldsymbol{e}$, mutate range $[\boldsymbol{m_a}, \boldsymbol{m_b}]_{\boldsymbol{e}}$, and mutation probability $\boldsymbol{\theta_r}$.
Output: Mutated breakpoint set $\hat{\mathcal{P}}$

$\hat{\mathcal{P}} \leftarrow \emptyset$ ▷ Initialize the mutated set
for each p in $\mathcal{P}$ **do** ▷ Mutate all individuals
  $rand_p \leftarrow$ random number in $[0, 1]$
  for $i = \boldsymbol{m_a}$ to $\boldsymbol{m_b}$ **do** ▷ $0 \le \boldsymbol{m_a} \le \boldsymbol{m_b}$
    if $i \cdot \boldsymbol{\theta_r} \le rand_p < (i+1) \cdot \boldsymbol{\theta_r}$ **then**
      $p' \leftarrow \lfloor p \cdot 2^i \rceil / 2^i$ ▷ Rounding mutation
      Append p' to $\hat{\mathcal{P}}$
      break ▷ Mutate only once
    end if
  end for
end for
Sort $\hat{\mathcal{P}}$ in ascending order ▷ Ensure correct order

limitation of applying a straightforward FXP conversion to *GQA-LUT*, especially at larger scaling factors, where breakpoint deviation becomes substantially more prominent. To tackle this challenge, we propose a novel strategy called Rounding Mutation (*RM*), detailed in Algorithm 2. The *RM* method images the quantization of breakpoints as a stochastic mutation, wherein quantization is applied randomly across various scales S, influencing each element of an individual set of breakpoints throughout the evolutionary process. In *GQA-LUT*, we substitute the conventional mutation function, which relies on normally distributed noise, with our *RM* approach, all the while preserving the straightforward FXP conversion for both slopes and intercepts. As depicted in Figure 2(a), the integration of the *RM* strategy within *GQA-LUT* markedly diminishes the MSE when S is large. Though there is a minor escalation in MSE for smaller S that is originally subordinate, the increment is minimal and can be practically disregarded. On the other hand, incorporating *RM* into *NN-LUT* is intricate, as it relies on a neural network-based framework where breakpoints are deduced from

Table 3: Comparison of Average MSE on Different Methods

Methods*	Entry	GELU	HSWISH	EXP	DIV	RSQRT
NN-LUT	8	$1.3e^{-3}$	$1.2e^{-3}$	$6.4e^{-4}$	$2.7e^{-3}$	$1.1e^{-2}$
	16	$2.7e^{-4}$	$7.9e^{-4}$	$2.3e^{-4}$	$2.4e^{-3}$	$2.8e^{-3}$
GQA-LUT w/o *RM*	8	$1.5e^{-4}$	$3.1e^{-4}$	$1.3e^{-4}$	$\mathbf{7.8e^{-4}}$	$\mathbf{1.2e^{-3}}$
	16	$9.9e^{-5}$	$2.8e^{-4}$	$1.1e^{-4}$	$1.3e^{-3}$	$5.0e^{-4}$
GQA-LUT w/*RM*	8	$\mathbf{9.4e^{-5}}$	$\mathbf{2.9e^{-4}}$	$\mathbf{1.2e^{-4}}$	$8.3e^{-4}$	$1.7e^{-3}$
	16	$9.6e^{-5}$	$2.2e^{-4}$	$7.4e^{-5}$	$1.4e^{-3}$	$1.2e^{-3}$

*: All methods focus on INT8 LUT approximation.

Table 4: Fine-tuning mIoU of Segformer-B0 on Cityscapes

Replacement*	*NN-LUT*	*GQA-LUT w/o RM*	*GQA-LUT w/RM*
None	74.60%	74.60%	74.60%
EXP only	73.88%	74.31%	74.59%
GELU only	73.61%	74.24%	74.57%
DIV only	74.15%	74.37%	74.58%
RSQRT only	73.94%	74.17%	74.51%
Altogether	73.46%$_{-1.14}$	74.28%$_{-0.32}$	74.53%$_{-0.07}$

*: The non-linear function that is replaced by 8-Entry pwl.

the slopes and intercepts, a methodology that is inherently inverse to that of *GQA-LUT*. The latter determines the slopes and intercepts directly from the breakpoints with varying precision levels.

4 EXPERIMENTAL RESULTS

In this section, a comprehensive comparison between *GQA-LUT* and *NN-LUT* is presented, initially focusing on operator-level accuracy for a variety of non-linear functions. Following this, the fine-tuning accuracy of two models, Segformer [3] and EfficientViT [4], specifically in the context of semantic segmentation tasks under integer-only quantization is evaluated. We further implement LUT-based pwl with various precision using Verilog HDL and benchmark their hardware performances including area and power dissipation.

4.1 Operator-Level Accuracy

We investigate five common non-linear functions in the Transformer and its variants, specifically: GELU, EXP, HSWISH, DIV, and the RSQRT. The hyperparameters associated with each function, as implemented under *GQA-LUT* are listed in Table 1. As for wide-range operators DIV and RSQRT, the multi-range input scaling strategy is adopted whose setup is illustrated in Table 2. We also re-implement *NN-LUT* following the training procedure described in [11] and directly convert the slopes, intercepts, and breakpoints to the same precision as *GQA-LUT*. To comprehensively evaluate operator-level accuracy with quantization awareness, we emphasize dequantized data instead of randomly selecting input data in the floating-point range [11]. During the evaluation, input data is orderly sampled from the dequantized range, for instance, $[Q_nS, Q_pS]$ with an incremental step size of S. Given that only GELU, EXP, and HSWISH are affected by the scaling factor S, we present a detailed MSE comparison for various S values in Figure (3). It can be seen that the *GQA-LUT w/RM* achieves a more stable and better performance across different S than *NN-LUT*. Furthermore, the accuracy for all operators presented in Table 3 distinctly illustrate the superior performance of *GQA-LUT w/RM* for GELU, EXP, and HSWISH that containing input scaling factor, outshining the

Table 5: Fine-tuning mIoU of EfficientViT-B0 on Cityscapes

Replacement*	*NN-LUT*	*GQA-LUT w/o RM*	*GQA-LUT w/RM*
None	74.17%	74.17%	74.17%
HSWISH only	73.55%	73.98%	74.20%
DIV only	73.21%	73.30%	74.08%
Altogether	73.27%$_{-0.90}$	73.79%$_{-0.38}$	74.15%$_{-0.02}$

*: The non-linear function that is replaced by 8-Entry pwl.

Table 6: Hardware Costs under TSMC 28-nm Technology

Precision*	Entry	Area (um^2)	Power (mW)
INT8	8	961	0.40
	16	1640	0.78
INT16	8	2080	0.85
	16	3521	1.47
INT32	8	5243	1.93
	16	8040	3.14
FP32	8	5135	2.02
	16	7913	3.47

*: Denotes the precision of input and LUT parameters.

NN-LUT even though both adopt INT8 integer-only quantization. But DIV and RSQRT that receive merely quantized input are apt to *GQA-LUT w/o RM* which also further proves the *RM* is born to handle data with changeful precision. Impressively, the required data of *GQA-LUT w/RM* (0.35~0.8K) is extremely lower than that of the *NN-LUT* (100K, reported in [11]).

4.2 Fine-tuning Accuracy

We extend our research by examining fine-tuning accuracy on the challenging and demanding Cityscapes dataset, a benchmark in semantic segmentation [21]. The Cityscapes is annotated at the pixel level, depicting urban scenes across 19 varied categories. It comprises 2,975 images for training, 500 for validation, and 1,525 for testing, all showcased in a high-resolution (1024×2048) format. In our analysis, we focus on two specific Transformer models: the Segformer-B0 at 1024×1024 resolution, a vanilla Transformer incorporating EXP, GELU, DIV, and RSQRT as non-linear operators, and the EfficientViT-B0 at 1920×1024 resolution, a lightweight Transformer variant proposed recently, it only contains HSWISH and DIV operators based on linear attention for edge devices. To begin with, we apply INT8 integer-only quantization to both the weights and activations of the aforementioned models, utilizing the LSQ method [19] and adhering to the dyadic pipeline [15]. In contrast to the approach in I-BERT, which involves updating the scaling factor in the non-linear function, we restrict the scaling factor for the input of the non-linear function as a power-of-two format, in alignment with the methodology outlined in Section 3.1. These quantized models subsequently serve as our baseline for comparison. Following the existing methods [3, 4, 22], we employ the standard mean Intersection over Union (mIoU) as our primary metric for evaluating fine-tuning performance. The accuracy of fine-tuning Segformer-B0 and EfficientViT-B0 is presented in Table 4 and Table 5, respectively. The results highlight that the *GQA-LUT w/RM* effectively maintains the fine-tuning accuracy for both Segformer-B0 and EfficientViT-B0, with minimal loss of 0.07% and

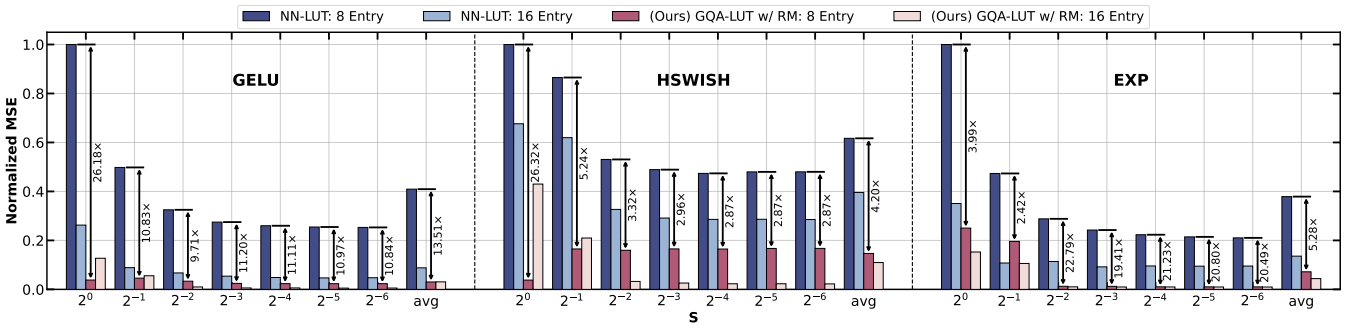


Figure 3: Comparison of normalized MSE for GELU, HSWISH, and EXP across various INT8 quantization scaling factors S using *NN-LUT*, *GQA-LUT*, and the *RM* strategy with 8/16-entry LUT approximation.

0.02% respectively. Compared to *NN-LUT*, it achieves improvements of 1.07% and 0.88% for these models.

4.3 Hardware Performance

In order to further prove the necessity and superiority of INT8 LUT-based *pwl*, we implement the two types of hardware units depicted in Figure (1) using Verilog HDL. The area and power dissipation of each LUT-based unit are obtained through synthesis using Synopsys Design Compiler, utilizing TSMC's 28-nm Technology. To ensure a fair comparison, the operating frequency is set to 500MHz for all hardware units. The comparison results are shown in Table 6 where the FP32 denotes the LUT-based *pwl* without input quantization representing the methods employed by *NN-LUT* and *RI-LUT*. The results reveal that an 8-entry INT8 LUT-based *pwl* requires merely 961um^2 of area, achieving remarkable reductions of 81.3% and 81.7% compared to high-precision FP32 and INT32 units, respectively. In terms of power dissipation, it consumes 0.4mW resulting in substantial savings of 80.2% in FP32 and 79.3% in INT32. Moreover, expanding the LUT storage to 16 entries results in an approximate 1.71× increase in area and 1.95× in power relative to the 8-entry INT8 configuration. Therefore, a smaller entry size and low-precision are crucial for *pwl* under integer-only quantization.

5 CONCLUSION

In this study, we introduce a new approach to LUT approximation emphasizing quantization awareness and reveal the phenomenon of breakpoint deviation in the integer-only quantization containing large scaling factors. To handle this, we propose a unique genetic quantization-aware *pwl* algorithm *GQA-LUT*, and an enhancement technique *RM*, which images the FXP conversion as a mutation process. The experimental results demonstrate that *GQA-LUT* with *RM* surpasses the previous state-of-the-art work *NN-LUT*, in both operator-level performance and fine-tuning accuracy. Additionally, the INT8 hardware *pwl* units integrated into *GQA-LUT* yield significant savings of 81.3~81.7% in area and 79.3~80.2% in power dissipation compared to their high-precision counterparts.

6 ACKNOWLEDGEMENT

This work was supported by ACCESS – AI Chip Center for Emerging Smart Systems, sponsored by InnoHK funding, Hong Kong.

REFERENCES

[1] Jacob Devlin et al. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2, 2019.

[2] Ze Liu et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[3] Enze Xie et al. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021.

[4] Han Cai et al. Efficientvit: Lightweight multi-scale attention for high-resolution dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17302–17313, October 2023.

[5] Dong Zhang et al. Augmented fcn: rethinking context modeling for semantic segmentation. *Science China Information Sciences*, 66(4):142105, 2023.

[6] Sehoon Kim et al. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pages 5506–5518. PMLR, 2021.

[7] Shih-Yang Liu et al. Oscillation-free quantization for low-bit vision transformers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 21813–21824. PMLR, 23–29 Jul 2023.

[8] Shih-yang Liu et al. Llm-fp4: 4-bit floating-point quantized transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 592–605, 2023.

[9] Fengbin Tu et al. Multcim: Digital computing-in-memory-based multimodal transformer accelerator with attention-token-bit hybrid sparsity. *IEEE Journal of Solid-State Circuits*, 2023.

[10] Jacob R Stevens et al. Softermax: Hardware/software co-design of an efficient softmax for transformers. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 469–474. IEEE, 2021.

[11] Joonsang Yu et al. Nn-lut: neural approximation of non-linear operations for efficient transformer inference. In *2023 59th ACM/IEEE Design Automation Conference (DAC)*, pages 577–582, 2022.

[12] Janghyeon Kim et al. Range-invariant approximation of non-linear operations for efficient bert fine-tuning. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2023.

[13] Xijie Huang et al. Sdq: Stochastic differentiable quantization with mixed precision. In *International Conference on Machine Learning*, pages 9295–9309. PMLR, 2022.

[14] Xianghong Hu et al. A tiny accelerator for mixed-bit sparse cnn based on efficient fetch method of simo spad. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 70(8):3079–3083, 2023.

[15] Benoit Jacob et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[16] Ashish Vaswani et al. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[17] Dongchen Han et al. Flatten transformer: Vision transformer using focused linear attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5961–5971, 2023.

[18] Yoshua Bengio et al. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[19] Steven K. Esser et al. Learned step size quantization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

[20] John Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press, 1992.

[21] Marius Cordts et al. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[22] Dong Zhang et al. Graph reasoning transformer for image parsing. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2380–2389, 2022.