HyQ: Hardware-Friendly Post-Training Quantization for CNN-Transformer Hybrid Networks

Nam Joon Kim[1] , **Jongho Lee**[1,2] and **Hyun Kim**[1]
[1]Department of Electrical and Information Engineering and Research Center for Electrical and Information Technology, Seoul National University of Science and Technology
[2]SqueezeBits Inc.
rlarla2626@seoultech.ac.kr, jongho.lee@squeezebit.com, hyunkim@seoultech.ac.kr

Abstract

Hybrid models that combine CNNs and ViTs have recently emerged as state-of-the-art computer vision models. To efficiently deploy these hybrid models on resource-constrained mobile/edge devices, quantization is emerging as a promising solution. However, post-training quantization (PTQ), which does not require retraining or labeled data, has not been extensively studied for hybrid models. In this study, we propose a novel PTQ technique specialized for CNN-transformer hybrid models by considering the hardware design of hybrid models on AI accelerators such as GPUs and FPGAs. First, we introduce quantization-aware distribution scaling to address the large outliers caused by inter-channel variance in convolution layers. Furthermore, in the transformer block, we propose approximating the integer-only softmax with a linear function. This approach allows us to avoid costly FP32/INT32 multiplications, resulting in more efficient computations. Experimental results show that the proposed quantization method with INT8 precision demonstrated a 0.39% accuracy drop compared with the FP32 baseline on MobileViT-s with the ImageNet-1k dataset. Furthermore, when implemented on the FPGA platform, the proposed linear softmax achieved significant resource savings, reducing the look-up table and flip-flop usage by $1.8 \sim 2.1\times$ and $1.3 \sim 1.9\times$, respectively, compared with the existing second-order polynomial approximation. The code is available at https://github.com/IDSL-SeoulTech/HyQ.

1 Introduction

Convolutional neural networks (CNNs), which are widely used in computer vision tasks, learn translation equivariance properties using small local receptive fields and thus have high computational and parameter efficiencies [He *et al.*, 2016]. However, the local connectivity structure limits their ability to model global contexts and long-range dependencies in the data [Naseer *et al.*, 2021]. Conversely, emerging vision transformers (ViTs) apply global self-attention mechanisms to capture non-local relationships between all spatial locations, achieving higher accuracy than CNNs owing to their large representational capacity [Dosovitskiy *et al.*, 2020]. Although ViTs can acquire such a high self-attention capacity, they also increase computational complexity and parameter size [Liu *et al.*, 2021a]. For example, the ViT-Base [Dosovitskiy *et al.*, 2020] and Swin-Base [Liu *et al.*, 2021a] models have 86M and 87M parameters, respectively. Accordingly, these models require massive DRAM access during inference, resulting in significant energy consumption[Nguyen *et al.*, 2020b], making them unsuitable for deployment on resource-constrained mobile/edge platforms [Zhou *et al.*, 2022; Nguyen *et al.*, 2020a].

To address this limitation, hybrid models that combine CNNs and ViTs have recently achieved state-of-the-art (SOTA) performance in various vision tasks [Mehta and Rastegari, 2021; Yang *et al.*, 2022; Li *et al.*, 2022]. To exploit the strengths of both CNNs and ViTs in CNN-transformer hybrid models, mobile convolution (MBConv) layers [Sandler *et al.*, 2018] are used in the early stages to extract local features and then passed to transformer encoders to learn global representations [Mehta and Rastegari, 2021; Yang *et al.*, 2022]. Although these hybrid models achieve SOTA performance in the trade-off between computation and accuracy, their high computational cost and memory footprint still impede the real-time inference of neural network (NN) models on resource-constrained mobile/edge devices. For example, MobileViT-s [Mehta and Rastegari, 2021] has 2G floating-point operations (FLOPs), which exceeds the typical mobile inference budget (<600M FLOPs) [Cai *et al.*, 2019].

Quantization has become the most widely used approach for ensuring efficient deployment of NN models on resource-constrained mobile/edge devices [Kim and Kim, 2021]. Among the various quantization techniques, quantization-aware training (QAT) compensates for accuracy drops by retraining a quantized model using a full training set [Jacob *et al.*, 2018]. However, QAT complicates the training process and adds significant time costs. Specialized training techniques (such as straight-through estimators [Yin *et al.*, 2019]) or hyper-parameter tuning may also be required. Conversely, post-training quantization (PTQ) compresses the high-precision (*e.g.*, FP32) weights and activations of a pre-trained NN into low-precision values (*e.g.*, INT8) to accelerate hardware inference without requiring access to training data or retraining [Nagel *et al.*, 2021]. This makes PTQ

highly useful when training data are unavailable or additional training is infeasible. However, applying PTQ to neural networks (NNs) without considering their characteristics significantly reduces accuracy, particularly for complex ViT models compared with CNNs [Liu *et al.*, 2021b]. Based on in-depth analyses, new quantization techniques tailored to ViTs have recently been proposed, including FQ-ViT [Lin *et al.*, 2021], PTQ4ViT [Yuan *et al.*, 2022], I-ViT [Li and Gu, 2022], and NoisyQuant [Liu *et al.*, 2023]. However, these methods are limited to standard transformer models and are difficult to apply to hybrid models containing convolutional blocks that have different characteristics than transformer models. The quantization solution for the hybrid model, Q-HyViT [Lee *et al.*, 2023], determines the mixed quantization granularity and schemes for each layer using Hessian information. However, mixed granularity and schemes require complex implementation and dedicated kernels for inference acceleration. In addition, Q-HyViT is specific to the MobileViT architecture and consequently exhibits poor compatibility.

In this study, we propose HyQ, a novel quantization method for CNN-transformer hybrid models. First, we introduce quantization-aware distribution scaling (QADS) to eliminate large outliers caused by inter-channel variances in CNNs (*i.e.*, MBConvs). We also propose approximating the computationally expensive softmax function in transformer blocks with an integer-only linear exponential function while considering hardware efficiency. As shown in Fig.1, we quantize various hybrid models, including MobileViT [Mehta and Rastegari, 2021], MobileViTv2 [Mehta and Rastegari, 2022], and EfficientFormer [Li *et al.*, 2022], using the ImageNet-1k dataset [Deng *et al.*, 2009] to evaluate HyQ. Using HyQ, the quantized MobileViT-s can achieve a 75% parameter size reduction with only a 0.39% accuracy drop decrease from the FP32 baseline, demonstrating SOTA results. Moreover, using hardware-friendly techniques, HyQ overcomes the limitations of previous research in terms of hardware usage. As a result of implementing the proposed linear softmax at the register-transfer level (RTL), the proposed linear softmax significantly reduces lookup table (LUT) and flip-flop (FF) usage without the utilization of digital signal processor (DSP), outperforming the existing second-order polynomial approximations. Our contributions can be summarized as follows:

- The QADS addresses the inter-channel variance challenges caused by depth-wise convolutions in MBConvs. Before quantization, QADS scales the original distributions with large outliers in the activations and weights in a quantization-aware manner.
- We propose a method to approximate the computationally expensive softmax, originally performed in FP32 precision, as a simple yet efficient linear function with integer-only computation.
- We demonstrate high hardware-friendliness and compatibility of QADS and linear softmax, emphasizing their practical and broad applicability.

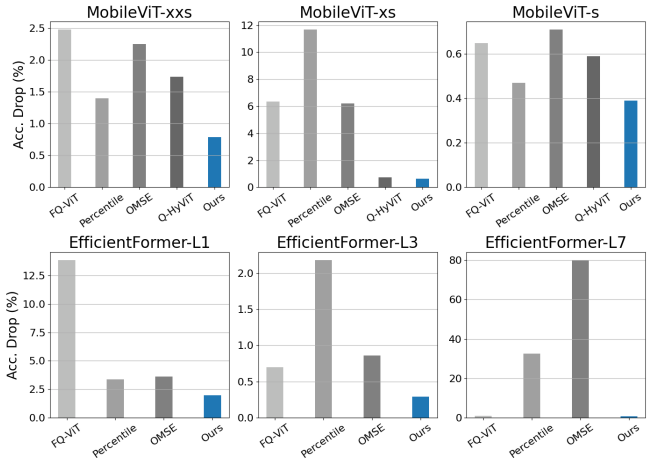

Figure 1: Comparison of Top-1 accuracy drops with various quantization methods and our proposed HyQ on the ImageNet-1k dataset for MobileViTs and EfficientFormers. All weights and activations were quantized to 8-bit integers (INT8).

2 Related Works

2.1 CNN-Transformer Hybrid Models

ViTs using self-attention have significant interest in computer vision tasks owing to their remarkable performance improvements in various tasks, such as image classification, object detection, and segmentation [Dosovitskiy *et al.*, 2020; Liu *et al.*, 2021a; Liu *et al.*, 2022]. Such advantages of ViTs over CNNs stem from the self-attention mechanism's ability to capture global contexts more accurately, whereas CNNs extract local features [Steiner *et al.*, 2021]. However, ViTs are often computationally costly and require substantial training data to achieve such performance gains. Therefore, various CNN-transformer hybrid models have recently been proposed, achieving improved trade-offs between accuracy and efficiency by combining the high computational efficiency of CNNs and the global modeling capacity of transformers.

MobileViT [Mehta and Rastegari, 2021] was designed to combine the strengths of CNNs and ViTs in mobile vision tasks. MobileViT reduces model complexity while maintaining accuracy by selectively replacing certain convolutions in MobileNetV2 [Sandler *et al.*, 2018]. Conversely, MOAT [Yang *et al.*, 2022] uses a different approach, which combines mobile convolution and transformer blocks into a single block, simplifying the model architecture. Moreover, MOAT can handle tasks that require high-resolution inputs by converting global attention to window attention. EfficientFormer [Li *et al.*, 2022] addresses the challenge of ViTs being slower than lightweight CNNs, which makes their deployment on mobile devices challenging, by using the dimension-consistent design and latency-driven slimming method. Despite being well-designed, hybrid models still have considerable model sizes of up to 82M parameters [Li *et al.*, 2022], requiring additional model compression techniques, such as quantization [Nagel *et al.*, 2021] and pruning [Kim and Kim, 2023], for efficient deployment on resource-constrained mobile/edge devices.

2.2 Post-Training Quantization

Quantization is an approach to improve the memory and computational efficiency of NNs by reducing the high precision of parameters to low precision. Quantization techniques are divided into two categories: QAT and PTQ. To mitigate accuracy losses caused by quantization, the existing QAT research necessitates access to the full training dataset, followed by extensive retraining across multiple QAT epochs [Yin *et al.*, 2019; Gysel *et al.*, 2018]. However, retraining quantized networks becomes infeasible in scenarios where data access is restricted or hardware resources are limited. Given these challenges, there is a growing preference for PTQ methods that do not require additional retraining or data access. This shift is further motivated by the rising costs of fine-tuning increasingly complex ViTs and language models.

Several PTQ techniques for CNNs have been proposed. Data-free quantization [Nagel *et al.*, 2019] proposed weight equalization to remove the large outliers of the weights and bias correction to compensate for the quantization error induced by the shifted output. OMSE [Choukroun *et al.*, 2019] utilized the mean squared error (MSE) function to identify the scaling factor with the least quantization error. On the other hand, for transformer models, NoisyQuant [Liu *et al.*, 2023] demonstrated that introducing minor random biases to activations can significantly reduce quantization errors. PTQ4ViT [Yuan *et al.*, 2022] applied twin uniform quantization to post-softmax and post-GeLU activations [Hendrycks and Gimpel, 2016a] and determined the optimal scaling factor using a hessian-guided metric. FQ-ViT [Lin *et al.*, 2021] proposed a power-of-two factor that enables integer-only operations to address the significant inter-channel variance problem in LayerNorm. SmoothQuant [Xiao *et al.*, 2023] relieves the difficulties of activation quantization by migrating channel-wise outliers in LLM activation to weights. OutlierSuppression [Wei *et al.*, 2023] shifts the channel-wise center and scales channel-wise outliers. However, one significant limitation of these methods is their reliance on the specific structures of CNNs or transformers. In other words, the direct application of these techniques to hybrid models often results in significant accuracy drops owing to the distinct parameter distributions and layer characteristics of CNNs and transformers.

To address this, Q-HyViT [Lee *et al.*, 2023] analyzed the activation distributions in hybrid models and then determined the optimal quantization granularity (per-tensor or per-channel factors) and scheme (asymmetric or symmetric) using Hessian-based analysis. However, the mixed granularity and schemes of Q-HyViT result in sub-optimal hardware efficiency and increased resource demands, making it less suitable for edge devices. Additionally, its applicability is confined to a specific MobileViT architecture, limiting its compatibility. In this study, we introduce HyQ, a hardware-optimized quantization method specialized for the efficient deployment of various hybrid models on edge devices.

3 Hardware-Friendly PTQ for Hybrid Models

3.1 Overview of the Proposed Quantization

For the CNN-transformer hybrid model, the proposed quantization maps all layers with core operations represented by high-precision values (*e.g.,* FP32) to low-precision values (*e.g.,* INT8). We apply INT quantization to all operations, including stem, convolution, MatMul, FC, softmax, LayerNorm, and shortcut layers. In particular, for fully integer quantization, we fold the BatchNorm weights into the preceding convolution weights before quantization. In addition, we employ uniform quantization with fixed step sizes for efficient support of diverse hardware (*i.e.*, GPU, FPGA, ASIC). For a b-bit unsigned uniform quantization, the quantized input $\mathbf{X_q}$ is defined as follows:

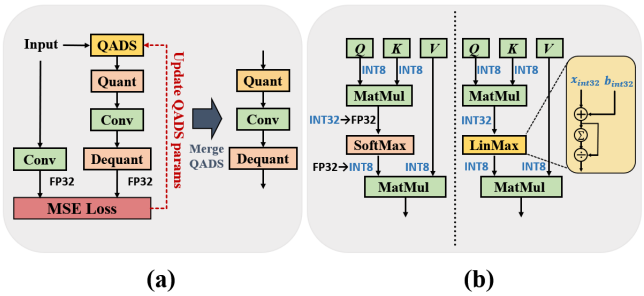

Figure 2: Overview of the proposed HyQ. (a) The QADS optimization process. First, a calibration set is used to determine the QADS parameters that minimize the difference between the FP32 and INT32 outputs. The obtained QADS parameters are then merged into the existing quantizer module. (b) Attention operation using conventional quantization (Left) vs. using the proposed linear softmax (Right).

$$\mathbf{X_q} \approx Q(\mathbf{X}) = \text{clip}(\text{round}(\frac{\mathbf{X}}{s}) + z,\ 0,\ 2^b - 1), \tag{1}$$

$$s = \frac{\max(\mathbf{X}) - \min(\mathbf{X})}{2^b - 1}, \tag{2}$$

$$z = \text{clip}(\text{round}(-\frac{\min(\mathbf{X})}{s}),\ 0,\ 2^b - 1), \tag{3}$$

where $Q(\mathbf{X})$ denotes the quantization function. The input $\mathbf{X} \in R^{C\times H\times W}$ denotes a floating-point real-value input with channel C and $H \times W$ dimensions. In contrast, $\mathbf{X_q} \in R^{C\times H\times W}$ denotes a b-bit integer value. The scaling factor $s \in R^1$ denotes the step size of the quantizer, and the zero point $z \in R^1$ denotes the real value zero as an integer.

First, we describe the overall flow of the proposed HyQ. Fig. 2(a) illustrates the QADS process. Initially, during the calibration step, an optimization process is conducted to determine the optimal QADS parameters using the MSE loss (as shown in the left side of Fig. 2(a)). Following the completion of calibration, the obtained QADS parameters are merged into the quantizer (as shown in the right side of Fig. 2(a)). Fig. 2(b) depicts the attention mechanism of the transformer. The left side depicts the conventional quantization method that processes softmax in FP32. In contrast, on the right side, the proposed linear softmax performs exponential operations on the INT32 input using only add operations.

3.2 Quantization-Aware Distribution Scaling

MBConv is a module used in various lightweight models (*e.g.,* MobileNetV2 [Sandler *et al.*, 2018], EfficientNet

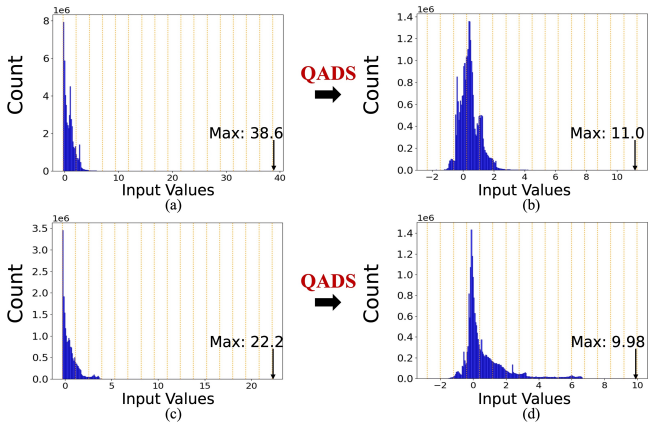

Figure 3: Input activation histograms of the third convolution layer in the MBConv blocks of MobileViT-xxs. The x-axis and y-axis represent the input values and the count, respectively. The yellow dotted lines indicate the quantization bins when quantizing the input activation to 4-bit using min-max quantization. (a), (c): Activation distributions with large outliers before applying QADS for the point-wise convolution layer of the first and second stages, respectively. (b), (d): Activation distributions after applying QADS for each stage, respectively.

[Tan and Le, 2019], and MobileViT [Mehta and Rastegari, 2021]) for efficient model usage on resource-constrained mobile/edge devices. It has recently been widely used in hybrid models that combine CNN and transformers to optimize the trade-off between accuracy and efficiency in both computations and parameters. However, the depth-wise convolution in MBConv causes significant inter-channel variance owing to its channel-wise operation [Kulkarni *et al.*, 2021]. As a result, this leads to large outliers, making quantization considerably challenging. As shown in Fig. 3, the maximum values of input activation for the point-wise convolution layer of the first and second stage of MobileViT-xxs are 38.6 and 22.2, respectively. When min-max quantization is applied to such a distribution, most values are mapped to only a few quantization bins (indicated by yellow dotted lines), resulting in most values being quantized to zero. This implies that only a few bits were used. Moreover, incorrect quantization of the MBConv blocks in the initial layers of hybrid models results in significant quantization errors. These quantization errors accumulate gradually, leading to incorrect predictions in the final layers.

To address this issue, we propose QADS. The core idea of QADS is to scale down channels with large input values (*i.e.,* outliers) to reduce the maximum value. This helps to map values more evenly to the quantization bin. First, we apply a per-channel scaling parameter $\boldsymbol{\alpha} \in R^C$ to the input activation $\mathbf{X}$ and weights $\mathbf{W}$. The operation of the convolution layer using the scaling parameter $\boldsymbol{\alpha}$ is expressed as follows:

$$\mathbf{Y} = (\frac{\mathbf{X}}{\boldsymbol{\alpha}})(\boldsymbol{\alpha}\mathbf{W}) = \widehat{\mathbf{X}}\widehat{\mathbf{W}}, \tag{4}$$

where $\widehat{\mathbf{X}}$ and $\widehat{\mathbf{W}}$ denote the scaled input and weight, respectively. Convolution using the quantization function Q is defined as follows:

$$\mathbf{Y_q} = Q(\widehat{\mathbf{X}}) * Q(\widehat{\mathbf{W}}). \tag{5}$$

The new $\widehat{\mathbf{W}}(= \mathbf{W}\boldsymbol{\alpha})$ can be pre-computed offline, but $\mathbf{X}$ changes dynamically and cannot be pre-scaled like weights. Therefore, to avoid additional scaling operations on $\mathbf{X}$, we absorb $\boldsymbol{\alpha}$ for $\mathbf{X}$ into the quantization scale factor s as follows:

$$\mathbf{Q}\left(\frac{\mathbf{X}}{\boldsymbol{\alpha}}\right) = \text{clip}\left(\text{round}\left(\frac{\mathbf{X}}{\mathbf{s}^*}\right) + z, 0, 2^b - 1\right), \tag{6}$$

$$\mathbf{s}^* = \boldsymbol{\alpha} \cdot s, \tag{7}$$

where $\mathbf{s}^* = \boldsymbol{\alpha} \cdot s$ denotes the new quantization scale factor. Hence, QADS uses channel-wise scaling factors while still achieving computationally efficient layer-wise quantization. These additional scaling parameters accounted for only 0.05% and 0.07% of the total number of parameters in the MobileViT-xxs and MobileViT-xs models, respectively, indicating a negligible overhead.

However, determining the optimal QADS factor $\boldsymbol{\alpha}$ poses a significant challenge. If $\boldsymbol{\alpha}$ corresponding to input channels is very large, quantization of the input $\mathbf{X}$ becomes easier while weight quantization becomes more difficult. Conversely, if $\boldsymbol{\alpha}$ is very small, the large outliers in input $\mathbf{X}$ make input quantization more difficult. Therefore, the optimal $\boldsymbol{\alpha}$ must be carefully determined by considering both input and weight values. To automatically explore $\boldsymbol{\alpha}$ without requiring heuristic knowledge, we employ a training-based method using the following objective function to determine the optimal $\boldsymbol{\alpha}^*$:

$$\boldsymbol{\alpha}^* = \arg\min\ L(\boldsymbol{\alpha}) = \arg\min \left\| Q(\frac{\mathbf{X}}{\boldsymbol{\alpha}})Q(\boldsymbol{\alpha}\mathbf{W}) - \mathbf{X}\mathbf{W} \right\|. \tag{8}$$

In other words, we search for $\boldsymbol{\alpha}$ that minimizes the MSE between the quantized output of the scaled input and weights and the original FP32 output. As shown in Fig. 3(b) and (d), the resulting scaled activation distributions using the optimal $\boldsymbol{\alpha}^*$ have lower maximum values (*i.e.,* 38.6 $\rightarrow$ 11.0, 22.2 $\rightarrow$ 9.98) and a more even mapping to quantization bins than the original distributions. In conclusion, the proposed QADS can significantly reduce quantization errors caused by activation. It is noteworthy that we have applied this approach to non-linear functions, such as swish activation, addressing issues that SmoothQuant [Xiao *et al.*, 2023] or OutlierSuppression [Wei *et al.*, 2023] cannot.

3.3 Linear Softmax: LinMax

In the transformer, softmax is used to convert the input values into a probability distribution between 0 and 1, and is expressed as follows:

$$\text{Softmax}(\mathbf{x}_i) = \frac{\exp^{\mathbf{x}_i}}{\sum_{j=1}^{k} \exp^{\mathbf{x}_j}}, \tag{9}$$

where $i = 1, ..., k$. The exponential (exp) function in softmax generally does not have linear properties like MatMul, making integer arithmetic operations infeasible (*i.e.*, MatMul(s $\cdot$ $\mathbf{X_q}$) = s $\cdot$ MatMul($\mathbf{X_q}$)); however, Softmax(s $\cdot$ $\mathbf{X_q}$) $\neq$ s

Algorithm 1 Integer-only Linear Exponential

1: **Input**: q, s: quantized input and scale
2: **Output**: q_{out},s_{out}: quantized output and scale
3: **function** Linear-EXP(q, s)
4: $\quad q_{\ln 2} \leftarrow round(-\ln 2/s)$
5: $\quad z \leftarrow round(q/q_{\ln 2})$
6: $\quad q_p \leftarrow q - z \cdot q_{\ln 2}$
7: $\quad q_b \leftarrow floor(b/s)$
8: $\quad s_L \leftarrow floor(a \cdot s)$
9: $\quad q_L = q + q_b$
10: $\quad q_{out}, s_{out} \leftarrow q_L << n - z, s_L/2^n$
11: **return** q_{out}, s_{out}
12: **end function**

$\cdot$ Softmax($\mathbf{X_q}$)). Hence, enabling integer computations for softmax is challenging. To address this, I-BERT [Kim *et al.*, 2021] approximated the exponential over a very small range $(-\ln 2, 0]$ using a second-order polynomial (*i.e.*, ax^2+bx+c) and processed it with INT32 precision to maintain network accuracy. However, INT32 multiplications consume substantial energy on hardware platforms, such as ASICs and FPGAs. For example, INT32 consumes energy $\times$15.5 and $\times$98 more than INT8 on ASIC and FPGA implementations, respectively [You *et al.*, 2020]. Furthermore, I-BERT requires additional QAT training.

Considering the efficiency of implementing CNN-transformer hybrid models on hardware platforms, we approximated softmax using a lower-order polynomial (*i.e.*, first-order) rather than the second-order polynomial used in I-BERT. Algorithm 1 presents the proposed integer-only linear exponential process. First, to ensure numerical stability, we restrict the range of the original exponential input by subtracting the maximum value as follows:

$$\text{Softmax}(\mathbf{x}_i) = \frac{\exp^{(\mathbf{x}_i - \mathbf{x}_{\max})}}{\sum_{j=1}^{k} \exp^{(\mathbf{x}_j - \mathbf{x}_{\max})}} = \frac{\exp^{\tilde{\mathbf{x}}_i}}{\sum_{j=1}^{k} \exp^{\tilde{\mathbf{x}}_j}}, \tag{10}$$

where $\mathbf{x}_{\max}$ denotes $\max(\mathbf{x}_i)$. $\tilde{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{x}_{\max}$ is a non-positive value. We can then decompose $\tilde{\mathbf{x}}$ as $(-\ln 2)z + p$, where z is the non-negative integer quotient of dividing $\tilde{\mathbf{x}}$ by $-\ln 2$ and p is the remainder in the range $(-\ln 2, 0]$. Then, the exponential for the input $\tilde{\mathbf{x}}_i$ can be expressed as follows:

$$\exp(\tilde{\mathbf{x}}) = 2^{-z} \exp(p) = \exp(p) >> z, \tag{11}$$

where 2^{-z} is efficiently converted to bit-shifting in hardware design. We seek approximation with a more hardware-efficient first-order polynomial (*i.e.*, a linear function) than the second-order polynomial approximation in I-BERT. A naive approach would be to use a linear function that passes through the points $(-\ln 2, \exp(-\ln 2))$ and $(0, 1)$ within the range $(-\ln 2, 0]$. Alternatively, we can reduce the approximation error by narrowing the approximation range, such as using the range $(-\ln \sqrt{2}, 0]$. However, these methods would incur significant approximation errors compared with second-order polynomials. Therefore, rather than approximating the original exponential function, we use a smoothed exponential (*e.g.,* $\exp(x)/8$, $\exp(x)/16$). For example, $\exp(x)/16$ has a shape more akin to a linear function than $\exp(x)$ over $(-\ln 2, 0]$. We reformulate softmax by scaling the exponential by $1/16$ as follows:

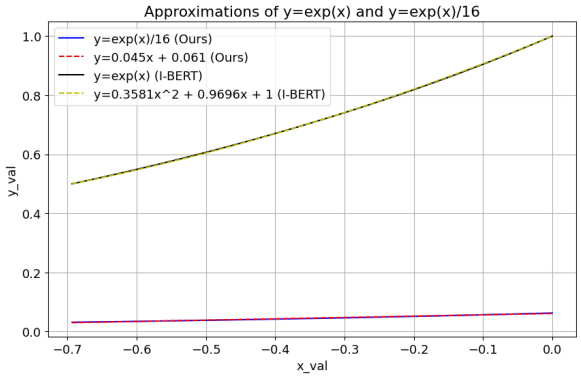

Figure 4: Comparison between the approximated second-order polynomial and the proposed first-order polynomial

$$\text{Softmax}(\mathbf{x}_i) = \frac{\frac{\exp^{\tilde{\mathbf{x}}_i}}{16}}{\sum_{j=1}^{k} \frac{\exp^{\tilde{\mathbf{x}}_j}}{16}}, \tag{12}$$

Then, the smoothed exponential for the input $\tilde{\mathbf{x}}_i$ can be expressed as follows:

$$\frac{\exp(\tilde{\mathbf{x}})}{16} = 2^{-z} \frac{\exp(p)}{16} = \frac{\exp(p)}{16} >> z, \tag{13}$$

where we approximate the smoothed exponential using a first-order polynomial (*i.e.*, $ax + b$) over $(-\ln 2, 0]$. We determine the coefficients a and b by minimizing the L2 distance between the smoothed exponential and first-order polynomials. Finally, we obtain the following approximated linear function as follows:

$$L(p) = 0.045p + 0.061 \approx \frac{\exp(p)}{16}, \tag{14}$$

$$\frac{\exp(\tilde{\mathbf{x}})}{16} \approx \text{Linear} - \exp(\tilde{\mathbf{x}}) = L(p) >> z, \tag{15}$$

where $z = round(-\tilde{\mathbf{x}} \ln 2)$ and $p = \tilde{\mathbf{x}} + z \ln 2$. Fig. 4 plots the results of the proposed linear softmax. The smoothed exponential and first-order functions are nearly identical. Particularly, the largest gap between these two functions over $(-\ln 2, 0]$ is 1.89×10^{-3}. Compared with the second-order polynomial approximation in I-BERT, which has a maximum gap of 1.91×10^{-3}, our proposed method provides superior approximation while retaining the hardware implementation advantages. Furthermore, the first coefficient (*i.e.* 'α' in $\alpha(x + \beta)$) is absorbed into the scaling factor s (in lines 8-9 of Algorithm 1). Therefore, the linear exponential needs only a simple and hardware-efficient operation of adding an integer value q_b. It should be noted that the second-order polynomial exponential in I-BERT can also absorb the first coefficient, but it requires two addition operations and one multiplication operation in INT32 precision (*i.e.*, $\alpha((x + \beta)^2 + \gamma)$).

Model	Method	Prec. (W/A)	Size (MB)	Top-1 Acc. (%)	Acc. Drop (%)
MobileViT-xxs	Baseline	32/32	1.27M	68.94	-
	FQ-ViT	8/8	0.32M	66.46	2.48
	Percentile	8/8	0.32M	67.54	1.40
	OMSE	8/8	0.32M	66.69	2.25
	Q-HyViT	8/8	0.32M	67.20	1.74
	Ours	8/8	0.32M	68.15	0.79
MobileViT-xs	Baseline	32/32	2.32M	74.63	-
	FQ-ViT	8/8	0.58M	68.28	6.35
	Percentile	8/8	0.58M	62.96	11.67
	OMSE	8/8	0.58M	68.41	6.22
	Q-HyViT	8/8	0.58M	73.89	0.75
	Ours	8/8	0.58M	73.99	0.64
MobileViT-s	Baseline	32/32	5.58M	78.32	-
	FQ-ViT	8/8	1.40M	77.67	0.65
	Percentile	8/8	1.40M	77.85	0.47
	OMSE	8/8	1.40M	77.61	0.71
	Q-HyViT	8/8	1.40M	77.72	0.59
	Ours	8/8	1.40M	77.93	0.39

Table 1: Comparison of top-1 accuracy with other quantization methods on ImageNet-1k

Model	Method	Prec. (W/A)	Top-1 Acc. (%)	Acc. Drop (%)
EfficientFormer-L1	Baseline	32/32	80.50	-
	FQ-ViT	8/8	66.63	13.87
	Percentile	8/8	77.15	3.35
	OMSE	8/8	76.90	3.6
	Ours	8/8	78.55	1.95
EfficientFormer-L3	Baseline	32/32	82.55	-
	FQ-ViT	8/8	81.85	0.7
	Percentile	8/8	80.37	2.18
	OMSE	8/8	81.69	0.86
	Ours	8/8	82.26	0.29
EfficientFormer-L7	Baseline	32/32	83.38	-
	FQ-ViT	8/8	82.47	0.91
	Percentile	8/8	50.94	32.44
	OMSE	8/8	3.23	80.15
	Ours	8/8	82.66	0.72
MobileViTv2-50	Baseline	32/32	70.16	-
	Q-HyViT	8/8	68.73	1.43
	Ours	8/8	69.16	1.00
MobileViTv2-75	Baseline	32/32	75.61	-
	Q-HyViT	8/8	74.36	1.25
	Ours	8/8	74.47	1.14

Table 2: Performance comparison of HyQ using SOTA hybrid models (*i.e.*, EfficientFormer and MobileViTv2) on ImageNet-1k

4 Experiments

4.1 Experimental Environments

The proposed HyQ framework was validated using the ImageNet-1k benchmark dataset [Deng *et al.*, 2009]. The hybrid models used in the experiments include MobileViT [Mehta and Rastegari, 2021], MobileViT-v2 [Mehta and Rastegari, 2022], and EfficientFormer [Li *et al.*, 2022]. Additionally, ViT, DeiT, and Swin models are used as ViT-type models for the ablation study. We applied the INT8 uniform PTQ to all layers except the activation function (*e.g.,* Swish [Ramachandran *et al.*, 2017], GeLU [Hendrycks and Gimpel, 2016b]) and attention output (following FQ-ViT, we used a more challenging INT4 logarithmic quantization for the attention output). To optimize the QADS parameters, we used 100 unlabeled images from the ImageNet-1k training set. We performed 1,000 iterations with a batch size of 100 using the Adam optimizer [Kingma and Ba, 2014]. We used the PyTorch framework [Paszke *et al.*, 2019] for all experiments and quantized the pre-trained models provided by the PyTorch Image Models library [Wightman, 2019]. Notably, applying QADS to MobileViT-xxs took only 2.5 minutes on a single NVIDIA GeForce RTX 3090 GPU.

4.2 Comparison with Other Quantization Methods on MobileViT Models

In this subsection, we compare the performances of MobileViT on the ImageNet-1k dataset with other quantization methods. As shown in Table 1, the proposed method demonstrated SOTA performance for the quantized MobileViT. In particular, for MobileViT-xxs, our method mitigates a 0.95% accuracy drop compared with Q-HyViT [Lee *et al.*, 2023] with the same quantization precision (*i.e.,* W8A8). For larger MobileViT-s models, our proposed HyQ outperforms Q-HyViT, despite using fully integer quantization with hardware-friendly linear softmax. Furthermore, for MobileViT-xs, due to zero point overflow [Lee *et al.*, 2023], existing quantization methods like OMSE [Choukroun *et al.*, 2019] and Percentile [Li *et al.*, 2019] exhibit severe performance degradation. In contrast, HyQ can substantially mitigate this issue by exploiting the fact that the minimum value of Swish used in MobileVit is fixed (-0.2785) when determining the scale and zero point for activation quantization.

4.3 Performance of HyQ with SOTA Hybrid Models

In Table 2, we highlight the impressive performance of HyQ on SOTA hybrid models. For the MobileViTv2 model, we quantize all layers following the same approach as we employ for MobileViT. Notably, unlike Q-HyViT, we achieve a smaller accuracy drop even without resorting to reconstruction techniques based on the Hessian matrix. Moreover, for the EfficientFormer model, we consistently maintain the smallest accuracy drop across EfficientFormer-L1, L3, and L7 models even though we apply quantization to the convolution, softmax, pooling, LayerScale, and LayerNorm layers. In contrast, other methods exhibited variable performance depending on model size. Specifically, FQ-ViT suffered from unacceptable performance degradation in the smallest models, while Percentile and OMSE methods encountered significant accuracy drops in the largest models. These results show the robustness of the proposed HyQ approach across hybrid models of various sizes.

4.4 Performance Analysis of Linear Softmax

Table 3 presents the performance of various ViT models (*i.e.*, DeiT-T/S, ViT-B/L, and Swin-T/S) as well as MobileViT series, applied with only the proposed linear softmax without QADS. We compare the accuracy of each model with I-BERT [Kim *et al.*, 2021], which proposed a second-order polynomial approximation of the exponential function. As shown in Table 3, our well-approximated first-order polynomials achieve nearly identical accuracy to the second-order

	DeiT-		ViT-		Swin-		MobileViT-		
	Tiny	Small	Base	Large	Tiny	Small	xxs	xs	s
Ours	70.90	78.47	82.63	84.92	79.94	82.37	67.18	73.95	77.83
I-BERT	70.82	78.36	82.64	84.91	79.97	82.43	67.05	73.93	77.88

Table 3: Top-1 Accuracy comparison of the proposed linear exponential and the second-order polynomial exponential approximation in I-BERT

Method	Unroll	LUT	FF	DSP	Power (mW)
Ours / I-BERT	16	2096 / 3776	1068 / 2096	0 / 48	251 / 371
	32	4237 / 7597	2087 / 3306	0 / 96	324 / 569
	64	8391 / 17676	4131 / 5573	0 / 174	450 / 988

Table 4: Comparison of hardware resource utilization between the proposed linear exponential and the second-order polynomial approximation in I-BERT when implemented on the FPGA platform

polynomial of I-BERT. Moreover, our proposed method outperforms I-BERT in terms of hardware implementation and computational complexity because it uses a first-order polynomial approximation and avoids expensive multiplications.

Based on these advantages, we validate the hardware efficiency of the proposed approach. To demonstrate that our proposed method is hardware-friendly, we compared the synthesis results of two approaches (*i.e.*, the proposed linear exponential and I-BERT) designed in RTL using the XC7Z0101 chipset in the Zynq-7000 board at 125 MHz. In Table 4, unroll [Rahman *et al.*, 2016] indicates how much the input of the exponential is pipelined, and LUT, FF, DSP, and Power are used as evaluation metrics. In detail, we used an input size of (4, 4, 64, 64) and the input data were unrolled for parallel processing. In addition, to reduce power, the data was loaded into BRAM and then processed. Experimental results show that our linear exponential uses fewer hardware resources (approximately $1.8 \sim 2.1\times$ LUT savings and $1.3 \sim 1.9\times$ FF savings) and consumes less power than the approximation method in I-BERT for all unroll factors. Importantly, our proposed method, which avoids INT32 multiplication, achieves this hardware efficiency without requiring DSP.

4.5 Ablation Study: QADS

We further analyzed the impact of QADS alone. The accuracies of MobileVit-xxs, xs, and s without QADS are 67.05%, 68.28%, and 77.67%, respectively. Conversely, with QADS, the accuracies become 68.15%, 73.99%, and 77.93%, respectively. These results show that MobileVit-xxs gains most from QADS, whereas MobileVit-s shows minimal improvement. This can be interpreted that the smaller the model, the more critical the impact of quantization of the backbone model, and it can be seen that QADS makes a significant contribution to minimizing this decrease in accuracy.

4.6 Ablation Study: Smoothed Exponentials

We demonstrate performance according to various smoothed exponential factors, specifically, $\frac{\exp(x)}{1}$, $\frac{\exp(x)}{2}$, $\frac{\exp(x)}{4}$, $\frac{\exp(x)}{8}$, and $\frac{\exp(x)}{16}$. This experiment was conducted using MobileViT-xxs on ImageNet without applying QADS to verify the effectiveness of linear softmax alone. The accuracies for smooth factors = 1, 2, 4, 8, 16 are 67.07%, 67.11%, 67.17%, 67.18%, 67.18%, respectively. We can observe that as the smooth factor decreases, the accuracy increases slightly and then saturates; thus, we set this factor empirically based on these findings.

Model	Dataset	Base Top-1/5 Acc. (%)	FQ-ViT Top-1/5 Acc. (%)	HyQ Top-1/5 Acc. (%)
MobileViT-xxs	V2	56.86/79.59	54.42/78.09	**56.1/79.08**
	Sketch	14.50/28.67	12.00/24.15	**12.98/25.94**
MobileViT-xs	V2	62.94/84.65	52.34/76.36	**62.38/84.42**
	Sketch	18.30/33.19	12.79/25.45	**17.07/31.36**
MobileViT-s	V2	66.75/87.01	65.91/86.58	**66.12/86.44**
	Sketch	22.48/38.67	21.01/36.26	**21.81/37.65**

Table 5: Accuracy comparison using MobileViT-xxs, xs, and s on the ImageNet-V2 and ImageNet-Sketch datasets

4.7 Ablation Study: Robustness on Other Datasets

The robustness of the proposed HyQ model is further evaluated using the ImageNet-v2 and ImageNet-Sketch datasets to examine performance against minor distributional shifts and significant domain alterations, respectively. This evaluation demonstrates the capability of HyQ to preserve feature recognition and model efficiency in diverse conditions. We compared the Top-1 & Top-5 accuracy of HyQ with FQ-ViT[Lin *et al.*, 2021] for MobileViT-xxs, xs, and s models. For QADS parameter optimization, we utilized 100 unlabeled images from the ImageNet-1k training set. As shown in Table 5, on the ImageNet-V2 dataset, we mitigated the Top-1 accuracy drop by 1.68%, 10.04%, and 0.21% for MobileViT-xxs, xs, and s, respectively, compared to FQ-ViT. On the ImageNet-Sketch dataset, HyQ exhibits slightly more accuracy degradation than ImageNet-V2 but still significantly outperforms FQ-ViT in reducing the accuracy loss.

5 Conclusion

In this study, we propose HyQ, a novel quantization technique for CNN-transformer hybrid models. HyQ adaptively employs QADS to handle outliers in a CNN and approximates softmax in the transformer as an integer-only linear function. Compared with existing methods, HyQ achieves significant performance gains in complex hybrid architectures. In particular, we demonstrate the superior hardware efficiency of HyQ by considering implementations on AI accelerators. Additionally, we verified the broad compatibility of HyQ by applying it to diverse hybrid models, and its high compatibility was confirmed by applying the proposed QADS and linear softmax to various CNN and ViT models, respectively.

Acknowledgments

This research was partly supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2024-RS-2022-00156295) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation and the Industrial Fundamental Technology Development Program (No. 20019367, Development of Low Power AI Architecture for AIoT) funded by the Ministry of Trade, Industry & Energy (MOTIE) of Korea.

Contribution Statement

Nam Joon Kim and Jongho Lee designed the algorithm and wrote the paper (equal contribution). Hyun Kim is the project leader who supervised this work.

References

[Cai *et al.*, 2019] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019.

[Choukroun *et al.*, 2019] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3009–3018. IEEE, 2019.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[Gysel *et al.*, 2018] Philipp Gysel, Jon Pimentel, Mohammad Motamedi, and Soheil Ghiasi. Ristretto: A framework for empirical study of resource-efficient inference in convolutional neural networks. *IEEE transactions on neural networks and learning systems*, 29(11):5784–5789, 2018.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hendrycks and Gimpel, 2016a] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[Hendrycks and Gimpel, 2016b] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[Jacob *et al.*, 2018] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 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.

[Kim and Kim, 2021] Sungrae Kim and Hyun Kim. Zero-centered fixed-point quantization with iterative retraining for deep convolutional neural network-based object detectors. *IEEE Access*, 9:20828–20839, 2021.

[Kim and Kim, 2023] Nam Joon Kim and Hyun Kim. Fp-agl: Filter pruning with adaptive gradient learning for accelerating deep convolutional neural networks. *IEEE Transactions on Multimedia*, 25:5279–5290, 2023.

[Kim *et al.*, 2021] Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pages 5506–5518. PMLR, 2021.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kulkarni *et al.*, 2021] Uday Kulkarni, SM Meena, Sunil V Gurlahosur, and Gopal Bhogar. Quantization friendly mobilenet (qf-mobilenet) architecture for vision based applications on embedded platforms. *Neural Networks*, 136:28–39, 2021.

[Lee *et al.*, 2023] Jemin Lee, Yongin Kwon, Jeman Park, Misun Yu, and Hwanjun Song. Q-hyvit: Post-training quantization for hybrid vision transformer with bridge block reconstruction. *arXiv preprint arXiv:2303.12557*, 2023.

[Li and Gu, 2022] Zhikai Li and Qingyi Gu. I-vit: integer-only quantization for efficient vision transformer inference. *arXiv preprint arXiv:2207.01405*, 2022.

[Li *et al.*, 2019] Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2810–2819, 2019.

[Li *et al.*, 2022] Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. *Advances in Neural Information Processing Systems*, 35:12934–12949, 2022.

[Lin *et al.*, 2021] Yang Lin, Tianyu Zhang, Peiqin Sun, Zheng Li, and Shuchang Zhou. Fq-vit: Post-training quantization for fully quantized vision transformer. *arXiv preprint arXiv:2111.13824*, 2021.

[Liu *et al.*, 2021a] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[Liu *et al.*, 2021b] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021.

[Liu *et al.*, 2022] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.

[Liu *et al.*, 2023] Yijiang Liu, Huanrui Yang, Zhen Dong, Kurt Keutzer, Li Du, and Shanghang Zhang. Noisyquant: Noisy bias-enhanced post-training activation quantization

for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20321–20330, 2023.

[Mehta and Rastegari, 2021] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*, 2021.

[Mehta and Rastegari, 2022] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers. *arXiv preprint arXiv:2206.02680*, 2022.

[Nagel *et al.*, 2019] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1325–1334, 2019.

[Nagel *et al.*, 2021] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart Van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021.

[Naseer *et al.*, 2021] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308, 2021.

[Nguyen *et al.*, 2020a] Duy Thanh Nguyen, Nguyen Huy Hung, Hyun Kim, and Hyuk-Jae Lee. An approximate memory architecture for energy saving in deep learning applications. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 67(5):1588–1601, 2020.

[Nguyen *et al.*, 2020b] Duy Thanh Nguyen, Hyun Kim, and Hyuk-Jae Lee. Layer-specific optimization for mixed data flow with mixed precision in fpga design for cnn-based object detectors. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(6):2450–2464, 2020.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[Rahman *et al.*, 2016] Atul Rahman, Jongeun Lee, and Kiyoung Choi. Efficient fpga acceleration of convolutional neural networks using logical-3d compute array. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1393–1398. IEEE, 2016.

[Ramachandran *et al.*, 2017] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[Sandler *et al.*, 2018] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[Steiner *et al.*, 2021] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[Tan and Le, 2019] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[Wei *et al.*, 2023] Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *arXiv preprint arXiv:2304.09145*, 2023.

[Wightman, 2019] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. Accessed: 2023-08-08.

[Xiao *et al.*, 2023] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

[Yang *et al.*, 2022] Chenglin Yang, Siyuan Qiao, Qihang Yu, Xiaoding Yuan, Yukun Zhu, Alan Yuille, Hartwig Adam, and Liang-Chieh Chen. Moat: Alternating mobile convolution and attention brings strong vision models. *arXiv preprint arXiv:2210.01820*, 2022.

[Yin *et al.*, 2019] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. *arXiv preprint arXiv:1903.05662*, 2019.

[You *et al.*, 2020] Haoran You, Xiaohan Chen, Yongan Zhang, Chaojian Li, Sicheng Li, Zihao Liu, Zhangyang Wang, and Yingyan Lin. Shiftaddnet: A hardware-inspired deep network. *Advances in Neural Information Processing Systems*, 33:2771–2783, 2020.

[Yuan *et al.*, 2022] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European Conference on Computer Vision*, pages 191–207. Springer, 2022.

[Zhou *et al.*, 2022] Minxuan Zhou, Weihong Xu, Jaeyoung Kang, and Tajana Rosing. Transpim: A memory-based acceleration via software-hardware co-design for transformer. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 1071–1085. IEEE, 2022.