A&B BNN: Add&Bit-Operation-Only Hardware-Friendly Binary Neural Network

Ruichen Ma, Guanchao Qiao, Yian Liu, Liwei Meng, Ning Ning, Yang Liu, Shaogang Hu*
University of Electronic Science and Technology of China
ruichen.ma@std.uestc.edu.cn, sghu@uestc.edu.cn

Abstract

Binary neural networks utilize 1-bit quantized weights and activations to reduce both the model's storage demands and computational burden. However, advanced binary architectures still incorporate millions of inefficient and non-hardware-friendly full-precision multiplication operations. A&B BNN is proposed to directly remove part of the multiplication operations in a traditional BNN and replace the rest with an equal number of bit operations, introducing the mask layer and the quantized RPReLU structure based on the normalizer-free network architecture. The mask layer can be removed during inference by leveraging the intrinsic characteristics of BNN with straightforward mathematical transformations to avoid the associated multiplication operations. The quantized RPReLU structure enables more efficient bit operations by constraining its slope to be integer powers of 2. Experimental results achieved 92.30%, 69.35%, and 66.89% on the CIFAR-10, CIFAR-100, and ImageNet datasets, respectively, which are competitive with the state-of-the-art. Ablation studies have verified the efficacy of the quantized RPReLU structure, leading to a 1.14% enhancement on the ImageNet compared to using a fixed slope RLeakyReLU. The proposed add&bit-operation-only BNN offers an innovative approach for hardware-friendly network architecture.

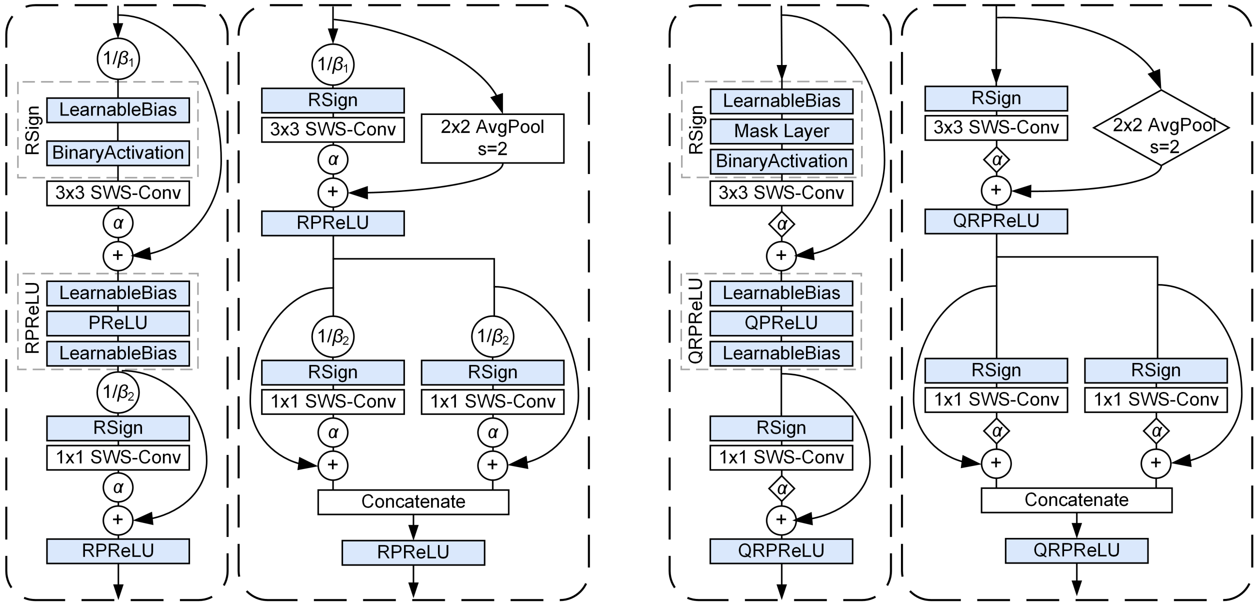
1. Introduction

Neural networks have made remarkable strides in tasks including image classification [1–3], object detection [4, 5], speech recognition [6, 7], and text generation [8–10], significantly advancing the development of various fields. However, as the scale of deep neural networks (DNNs) expands, the substantial computational and storage requirements make them feasible only for running on powerful but expensive GPUs, leaving edge devices unattainable [11, 12]. Hardware-efficient network architectures, such as spiking neural networks (SNNs), although they may not outperform traditional networks in numerous domains, offer the advantage of eliminating multiplication operations [13, 14]. This reduction in hardware complexity substantially reduces expenses associated with chip design, making them appealing to chip designers [15–17].

Numerous approaches have been devised to mitigate hardware overhead in traditional DNNs, albeit often at the expense of modest performance compromises. Binary neural networks (BNNs) stand out among these approaches, aiming to achieve 1-bit quantization of network parameters and activations to reduce storage and computational requirements. Pioneering study [18] has enabled BNNs to perform inference exclusively through logical operations, leading to a notable decrease in chip power consumption and design cost. Recognizing its suboptimal performance on large datasets such as ImageNet, subsequent studies have introduced various techniques to improve overall performance. These techniques have become nearly indispensable for advanced BNN models, including scaling factors [19], BN [20] layers, PReLU [21] layers, and real-value residuals [22]. However, these layers will unavoidably introduce full-precision multiplication operations that are not conducive to hardware efficiency, conflicting with the fundamental goal of BNN. Although the multiplication operand (MO) introduced is only in the order of millions, it still imposes a significant burden on the hardware, and chip designs that circumvent multipliers are preferable.

This study presents A&B BNN, a binary network architecture designed to eliminate all multiplication operations during inference, and was evaluated on three widely used structures ResNet-18/34 and MobileNet. The accuracies for various mainstream BNNs and A&B BNN on the ImageNet dataset are presented in Tab. 1, with corresponding results of 61.39%, 65.19%, and 66.89% for the three structures, respectively. The key to eliminating multiplication lies in removing the BN layer in the network topology while minimizing the loss. Studies [26] and [27] introduced normalizer-free network architecture and study [25] extended this technique to BNN and proposed BN-Free BNN.

*Corresponding author



(a) Baseline Network Block of BN-Free ReActNet

(b) Proposed A&B BNN Block

Figure 1. The architecture overview of the (a) baseline BN-Free network and (b) proposed A&B BNN. In contrast to the baseline network, the proposed A&B BNN eliminates all multiplication operations. The multiplication resulting from β is absorbed into the newly introduced mask layer and can be removed directly during inference. Multiplications induced by both average pooling and α are substituted by equal but more efficient bit operations. Additionally, we introduce the quantized RPReLU structure, effectively removing the multiplication associated with PReLU. Circles represent multiplication operations, diamonds represent bit operations, and $\bigoplus$ represents residual addition.

Binary Network Architecture	MO	Acc
BNN-ResNet-18 [23]	1.51 M	42.2%
XNOR-ResNet-18 [19]	3.20 M	51.2%
Bi-ResNet-18 [22]	20.86 M	56.4%
ReActNet-18 [24]	22.37 M	65.5%
ReActNet-18 (BN-Free) [25]	4.60 M	61.1%
Bi-ResNet-34 [22]	22.20 M	62.2%
ReActNet-A [24]	10.79 M	69.4%
ReActNet-A (BN-Free) [25]	14.65 M	68.0%
Bi-ResNet-18 (A&B BNN)	**0**	**60.38%**
ReActNet-18 (A&B BNN)	**0**	**61.39%**
ReActNet-34 (A&B BNN)	**0**	**65.19%**
ReActNet-A (A&B BNN)	**0**	**66.89%**

Table 1. Top-1 Accuracies of different BNNs evaluated on ImageNet dataset.

While this topology appears to double the multiplication operand, it can be obviated through characteristics of BNN and several mathematical transformations. The mask layer is proposed to execute these multiplication operations, serving the purpose of gradient scaling during training, and can be removed during inference. Furthermore, we introduce the quantized RPReLU structure, replacing the multiplication operations introduced by PReLU with an equal number of bit operations. Experiments show that the proposed approach achieves competitive performance compared to the state-of-the-art on CIFAR-10, CIFAR-100, and ImageNet datasets while eliminating multiplication operations, representing a valuable trade-off. Our code is available at https://github.com/Ruichen0424/AB-BNN.

2. Releted Work

Binary neural networks. Binary neural networks binarize weights and activations through the sign function, which renders the backpropagation algorithm ineffective due to its non-differentiable characteristic until [18] proposed the straight-through estimator (STE) technique. STE passes the gradient of binary values to full-precision values directly to disregard the influence of the sign function in the chain rule. Study [19] proposed using scaling factors to compensate the loss incurred during binarization, $\|\mathbf{W}\|_{\ell 1}/n$ for weights and $\|\mathbf{a}\|_{\ell 1}/n$ for activations. We retain the weight factor as it can be integrated into the weight matrix while excluding the activation factor to avoid introducing any multiplication operations. [22] suggested employing real values in the residual instead of binary to enhance the expressive capacity of BNN. Since this would introduce multiplication operations, we did not use this technique. To enforce BNNs to learn similar distribution as full-precision networks [19, 28, 29], [24] introduced the ReAct Sign (RSign) and ReAct PReLU

Techniques	MO
Scaling factor of activations	$(c+1)\cdot h\cdot w$
BN layer	$c\cdot h\cdot w$
PReLU layer	$c\cdot h\cdot w$
Real-value residual	$c_{in}\cdot c_{out}\cdot h_{out}\cdot w_{out}\cdot k_h\cdot k_w$
Average pooling layer	$c_{out}\cdot h_{out}\cdot w_{out}$
α in BN-Free	$c\cdot h\cdot w$
β in BN-Free	$c\cdot h\cdot w$

Table 2. Multiplication operands introduced by different techniques.

(RPReLU) structures which added some bias layers based on the original. [25] applies the BN-Free topology proposed in [26, 27] to BNN to eliminate the BN layer, which is the foundation of this work and is shown in Fig. 1. This topology employs the scaled weight standardization technique to regulate the mean and variance of each layer's activations and proposes the residual blocks of the form $x_{\ell+1} = x_\ell + \alpha \cdot f_\ell(x_\ell/\beta_\ell)$. α controls the variance growth rate of the residual block and β regulates the activation distribution, constituting the primary source of multiplication in BN-Free BNN.

Efforts to eliminate multiplication. In BNN, the seminal work [18] introduced XNOR-counts to substitute full-precision multiplication operations, reducing hardware overhead by over 200 times. Studies [22, 30–33] proposed diverse STE functions to enhance training effectiveness, while [34–38] employed a range of regularization terms to enhance weights and pre-activation distributions, thereby augmenting network expression. Additionally, there exist studies [39, 40] focused on enhancing network performance through optimizing training strategies and [41–43] dedicated to refining the loss function. Regrettably, though none of these techniques introduce multiplication operations, almost all of these studies employ methods that do. In spiking neural networks [13, 14], activations manifest as discrete spikes i.e. $\{0,\ 1\}$, enabling computation solely through addition operations. [44] proposed AdderNet framework wherein ℓ_1 distance is employed instead of convolution as a metric for assessing the relationship between features and filters within neural networks, with operations limited to addition and subtraction. Nevertheless, because of the presence of the BN layer, there are still millions of multiplication operations involved, making it not a true multiplication-operation-free network. Table 2 shows the multiplication operands introduced by different techniques.

3. Method

This section initially presents the foundational knowledge, followed by an assessment of the multiplication operands and their distribution within the BN-Free structure. Subsequently, we explore the removable mask layer and then introduce the bit operation and the quantized RPRuLU unit. Finally, we discuss the practical hardware benefits.

3.1. Preliminary

Scaled weight standardization. To address the mean shift and variance explode or vanish on activations resulting from BN removal, the scaled weight standardization (SWS) technique from [26] was introduced. Specifically, the weights are scaled as follows:

$$\hat{W}_{i,j} = \gamma \cdot \frac{W_{i,j} - \mu_i}{\sqrt{N}\sigma_i} \tag{1}$$

where $\mu_i = (1/N)\sum_j W_{i,j}$, $\sigma_i^2 = (1/N)\sum_j (W_{i,j} - \mu_i)^2$, N is the fan-in, and $\hat{W}_{i,j}$ is the corresponding standardized weights. γ is related to the activation function and is 1 for the sign function. SWS technique does not introduce any multiplication operation during inference.

Adaptive gradient clipping. Gradient clipping is commonly used to restrict the norm of gradients [45] to maintain training stability [46]. [27] proposed the adaptive gradient clipping (AGC) technique to improve the performance of the normalizer-free network, which can be described as:

$$G_i^l \to \begin{cases} \lambda \cdot \frac{\|W_i^l\|_F^*}{\|G_i^l\|_F} G_i^l, & \text{if } \frac{\|G_i^l\|_F}{\|W_i^l\|_F^*} > \lambda \\ G_i^l, & \text{otherwise} \end{cases} \tag{2}$$

where l represents the corresponding layer and i represents the i_{th} row of a matrix. $\|W_i^l\|_F^* = \max\{\|W_i\|_F, \epsilon\}$, $\epsilon = 10^{-3}$ and $\|\cdot\|_F$ is the Frobenius norm. The clipping threshold λ is a crucial hyperparameter that is usually tuned by grid search.

Distillation loss functions. Study [24] introduced Distillation loss to enforce the similarity of distribution between full-precision networks and binary networks to improve performance. The loss is calculated as follows:

$$\mathcal{L}_{\text{Dis}} = -\frac{1}{n}\sum_c \sum_{i=1}^{n} \rho_c^{\mathcal{R}}(X_i) \times \log\left(\frac{\rho_c^{\mathcal{B}}(X_i)}{\rho_c^{\mathcal{R}}(X_i)}\right) \tag{3}$$

where n is the batch size, c represents classes and X_i is the input image. $\rho_c^{\mathcal{R}}$ is the softmax output of the full-precision network and $\rho_c^{\mathcal{B}}$ represents the corresponding softmax output of the binary network.

3.2. Multiplication Operations in BN-Free BNN

The components that introduce full-precision multiplication operations in the BN-Free BNN architecture [25] include α, β, RPReLU, and the average pooling layer. α and β serve as manually designed scale factors, normalizing the input and output to preserve the advantages of the BN layer. To normalize the variance, $\beta = \sqrt{\text{Var}(x_{in})}$ is applied before the

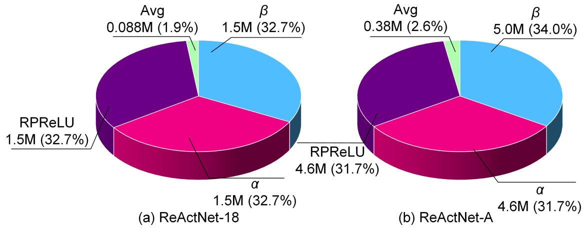


Figure 2. The multiplication operand and corresponding ratio within the BN-Free ReActNet-18 and ReActNet-A structures. For input images with a resolution of 224 × 224, the former generates approximately 4.6 million multiplication operations, while the latter yields approximately 14.7 million.

convolution operation, and then multiplied by α for further computations. Typically, β corresponds to the expected empirical standard deviation of the activation during initialization, whereas α is commonly assigned a small value, such as 0.2. Figure 2 illustrates the operand for the multiplication operations and the corresponding ratios of BN-Free-based ReActNet-18 and ReActNet-A on an input picture with the resolution of 224 × 224. The former performs 4.6 million multiplication operations, while the latter performs 14.7 million.

3.3. Removable Mask Layer

The mask layer is introduced first to comprehend the gradient transfer through the sign function in BNN. Then we explain the utilization of mathematical transformation to efficiently eliminate the multiplication operation caused by β without incurring any additional cost.

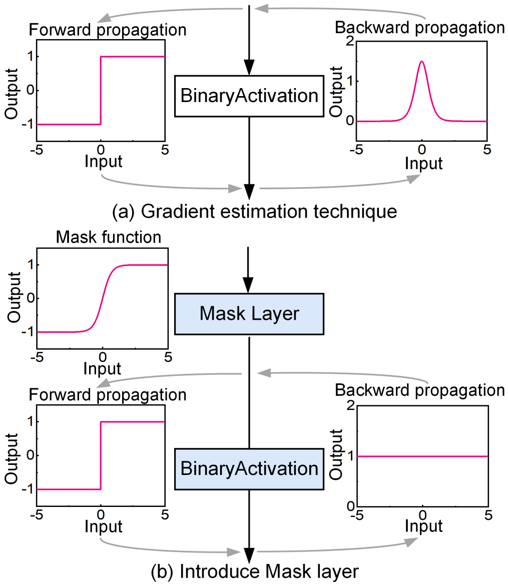
Study [18] proposed the STE technique to realize gradient transfer of the sign function, and subsequent work proposed various approximation functions $f_A(\cdot)$ for optimization. This technique involves using the sign function for binarization during forward propagation and utilizing the derivative of an approximation function during backpropagation to transfer gradients as shown in Fig. 3a and Eq. (4).

$$\text{BinaryActivation} \rightarrow \begin{cases} \text{Sign}(\cdot), & \text{forward pass} \\ f'_A(\cdot), & \text{backward pass} \end{cases} \tag{4}$$

Mathematically, the gradient approximation technique is equivalent to the introduction of a mask layer before the binary activation layer. This layer serves as an additional activation function, mapping the pre-activations to the binarization layer, and then the binarization layer transfers the gradient during backpropagation directly. The equivalent process is depicted in Fig. 3b.

It is crucial to maintain the numerical symbol of the mask layer unchanged before and after mapping. This layer satisfies the relation:

$$\text{Sign}(\text{Mask}(x)) = \text{Sign}(\text{Mask}(k \cdot x)) \equiv \text{Sign}(x) \tag{5}$$



(a) Gradient estimation technique

(b) Introduce Mask layer

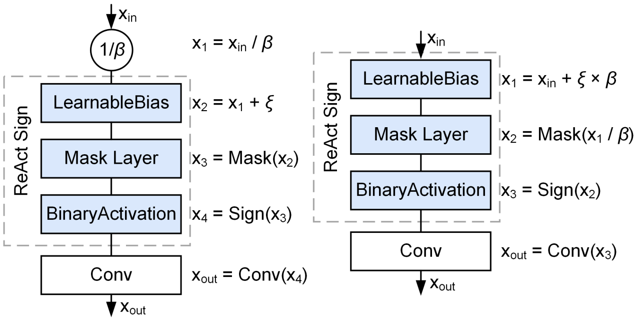
Figure 3. (a) A visualization of gradient approximation techniques. The gradient is transferred through an approximate function that resembles the impulse function. (b) Introduce the mask layer to achieve the same effect.

where k is a positive value. This relationship demonstrates the mathematical properties of the mask layer, enabling the absorption of a multiplication factor during forward propagation and effectively eliminating the multiplication operations caused by β. When backpropagating, it satisfies:

$$\frac{\partial y_{out}}{\partial x_{in}} = \frac{\partial y_{out}}{\partial y_{ML}} \frac{\partial y_{ML}}{\partial x_{in}} = 1 \cdot f'_A(x) \equiv f'_A(x) \tag{6}$$

The same gradient transfer effect can be obtained. Figure 4a illustrates the network structure near β, showcasing the operation of each step. In Fig. 4b, we propose an equivalent representation that equates one multiplication operation to two operations. At first glance, it might seem that instead of eliminating the multiplication operation, it increases it. However, after completing network training, the product of ξ and β becomes a constant value that does not require recalculation. Furthermore, by leveraging the characteristics of the mask layer, we can eliminate the second multiplication, achieving the same effect without any multiplications during inference. The mathematical characteristics of the mask layer imply that it does not play a role during inference and can be directly removed. It must be present during training as its significance lies in scaling and transforming values through mask function to regulate gradient transfer and prevent gradient vanishing. We employ the mask function described in Eq. (7) and set $\delta = 3$. Its non-zero characteristic improves the activation saturation issue in BNN.

$$\text{Sigmoid}(x, \delta) = \frac{1}{1 + e^{-\delta x}} \tag{7}$$



(a) With multiplication

(b) Without multiplication

Figure 4. (a) The original structure with multiplication. (b) The equivalent structure, although transforming one multiplication operation into two, can both be eliminated.

3.4. Bit Operations and Quantized RPRuLU Unit

Using bit operations for multiplication by powers of 2 is a straightforward and efficient technique, which relies on the binary representation of numbers and the properties of shifting operations. Multiplying a number by a power of 2 is equivalent to left-shifting its binary representation by a specific number of bits. Shifting operations are highly optimized in many architectures and can be performed quickly, often surpassing traditional multiplication algorithms. Within the BN-Free structure, the parameter α is typically assigned a small value, such as 0.2. By conducting the parameter search, it is possible to set α to a negative integer power of 2, effectively substituting multiplication operations with an equal number of bit operations. Similarly, all average pooling layer kernel sizes are set to 2×2, enabling the replacement of the division operation with a right shift of two bits.

When dealing with RPReLU, a straightforward approach is to substitute the PReLU with LeakyReLU and set the slope to a constant integer power of 2. Nevertheless, the results of ablation studies indicate a decrease in performance with this approach. To address this issue, we propose the quantized RPReLU unit. In this unit, the parameters of PReLU's each channel are quantized values and are constrained to integer powers of 2, the expression is as follows:

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i \geq 0 \\ 2^{\text{round}(a_i)} \cdot (y_i + \xi_{i_1}) + \xi_{i_2}, & \text{otherwise} \end{cases} \tag{8}$$

where a_i, ξ_{i_1}, and ξ_{i_2} are all learnable parameters correspond to the channel i. Figure 5 illustrates the function graph of PReLU and the proposed quantized RPReLU, respectively. The green area represents the allowed slope range, which can take any continuous value in RPReLU, whereas only discrete quantized values are allowed in quantized RPReLU. Ablation studies show that utilizing the quantized RPReLU can enhance accuracy by 1.14% when compared to using RLeakyReLU on the ImageNet dataset.

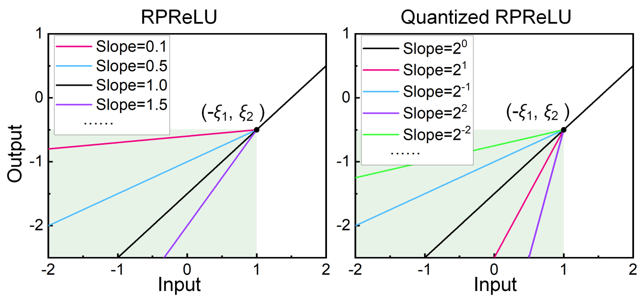


Figure 5. The slope of RPReLU can be any continuous value greater than 0, while the slope of the proposed quantized RPReLU is only allowed to be an integer power of 2.

Units	LUTs	Slice	DSPs
Multiplier	47	12	4
Bit-shift	32 (↓ 31.9%)	11 (↓ 8.3%)	0 (↓ 100%)
PReLU	57	17	4
QPReLU	32 (↓ 43.9%)	9 (↓ 47.1%)	0 (↓ 100%)

Table 3. Hardware overhead table for different units.

3.5. Hardware Benefits

The A&B BNN architecture proposed in this study holds significant practical importance. We conducted synthesis on various units using the Xilinx Zynq-7000 Z-7045 FPGA, and the hardware overhead is presented in Tab. 3. The resources integrated on the chip include 218,600 Look-Up Tables (LUTs), 54,650 slices, and 900 Digital Signal Processors (DSPs). The synthesis results demonstrate that the 32-bit full-precision multiplier consumes 47 LUTs, 12 slices, and 4 DSPs, whereas the bit-operator only necessitates 32 LUTs, 11 slices, and no DSP. Standard PReLU structures demand 57 LUTs, 17 slices, and 4 DSPs, while quantized PReLU structures utilize only 32 LUTs, 9 slices, and no DSP. We achieve this reduction by converting the multiplication operations introduced by the α parameter, the PRePU layer, and the average pooling layer into an equal number of bit-shift operations. Additionally, we eliminate the multiplication operations associated with β. Our approach does not introduce any additional computation or storage overhead. For one convolution layer without pooling, the reduction in consumption for three hardware resources is 57.6%, 51.2%, and 100%, respectively.

In scenarios where the chip lacks built-in multiplication support, yet the neural network demands it, a viable solution entails transmitting intermediate results to the host for computation and subsequently sending them back to the chip. This process results in frequent communication and introduces considerable delays. In contrast, the A&B BNN architecture enables the network to perform inference entirely within the chip, necessitating only a single round-trip communication. This streamlined approach significantly diminishes latency and improves real-time performance.

4. Experiments

4.1. Experimental Setup

We conducted experiments on the CIFAR-10 [47], CIFAR-100 [47], and ILSVRC12 ImageNet [48] datasets using the advanced ReActNet-18/34 and ReActNet-A network structures in binary networks and presented the results. In line with most binarization works, the input and output layers' weights are not quantized to ensure performance. All seeds were fixed to 2023 to ensure the experiments' repeatability.

Implementation details for ImageNet. ImageNet is a widely used benchmark dataset in computer vision, consisting of over 1.28 million training images, 50,000 test images, and 1,000 classes. Due to the significant success of ReActNet in the binary domain, we utilized our add&bit-operation-only network on ReActNet-18/34 and ReActNet-A, which are enhanced versions of ResNet-18/34 [3] and MobileNetv1 [49] structures, respectively. Additionally, we employed SWS and AGC techniques, setting λ to 0.02 for stability based on [25]. For training, we employed a two-step strategy [40], i.e., initially training from scratch and binarizing only the activations in the first stage, then fine-tuning based on the previous stage and binarizing both weights and activations in the second stage. At each stage, we utilized the Adam optimizer to minimize the Distillation loss functions [24] and training for 128 epochs, starting with an initial learning rate of 1e-3 and gradually decreasing to zero using a linear scheduler. The weight decay is set to 5e-6 in the first stage and 0 in the second stage. We use the same data augmentation as [24, 25], including random cropping, lighting, and random horizontal flipping, and use the same knowledge distillation scheme. For the test set, the image is resized to 256, center-cropped to 224, and then inputted into the network.

Implementation details for CIFAR-10 and CIFAR-100. CIFAR-10 and CIFAR-100 datasets consist of 50,000 training images and 10,000 testing images, divided into 10 and 100 classes, respectively. We conducted experiments using the ReActNet-18 and ReActNet-A structures, following the same two-step training strategy as in the ImageNet experiments, with each step trained for 256 epochs. To illustrate the versatility of the proposed algorithm, experiments were also conducted on other two mainstream binary structures Bi-real BNN and XNOR BNN. Since the ReActNet-A architecture includes more sub-sampling units and is not well-suited for handling small-sized datasets such as CIFAR, we adjusted the stride of the initial layer to 1 like ResNet. Data augmentation included random cropping and horizontal flipping. Additionally, we adopted the SWS and AGC techniques, setting λ to 0.001 based on [25]. The remaining experimental settings align with those used in the ImageNet experiments.

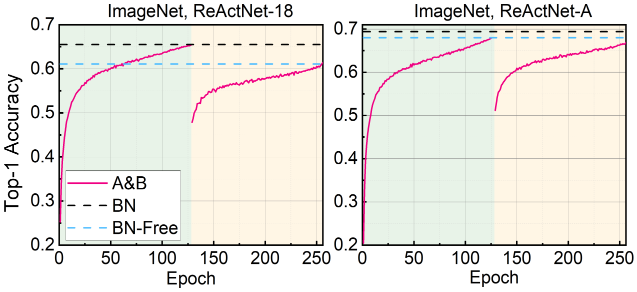


Figure 6. Top-1 accuracy on the ImageNet dataset, and comparison with two baselines. The green background represents the first training step and the orange represents the second step.

4.2. Comparison with State-of-the-Arts

Results on ImageNet. We first tested the performance of the proposed A&B ReActNet-18/34 and ReActNet-A models on ImageNet to validate the algorithm's effectiveness. The Top-1 accuracies for the three network structures were 61.39%, 65.19%, and 66.89%, while the Top-5 accuracies were 83.06%, 86.03%, and 86.83%, respectively. The comparison results are presented in Tab. 1, and Fig. 6 illustrates the Top-1 training results. Compared to the BN-Free structure, we achieved a reduction of 14.7 million multiplication operations with an accuracy loss of 1.11% on ReActNet-A. We achieved a reduction of 4.6 million multiplication operations on ReActNet-18 while improving the accuracy by 0.29%. Considering the actual application scenarios of BNN, its model complexity is much lower than MobileNet, and the task difficulty is much lower than ImageNet. The only 1.1% performance loss on ImageNet is difficult to feel on edge applications. Conversely, the hardware overhead is an urgent problem to be solved, the multiplier-less structure proposed is of great significance in actual production [50].

Results on CIFAR-10 and CIFAR-100. To further validate the algorithm's effectiveness and versatility, we conducted additional experiments using the widely used smaller datasets CIFAR-10 and CIFAR-100. The comparison results are presented in Tab. 4, while the Top-1 training results are depicted in Fig. 7. The algorithm demonstrated competitive results on both datasets. The results of ReActNet-A on CIFAR validate the intuition of modifying the structure. Merely reducing the stride of the first layer convolution can lead to a substantial improvement ranging from 5.53% to 12.93%. Since quantization methods often have higher variance, providing the mean and standard deviation would provide a better understanding of the

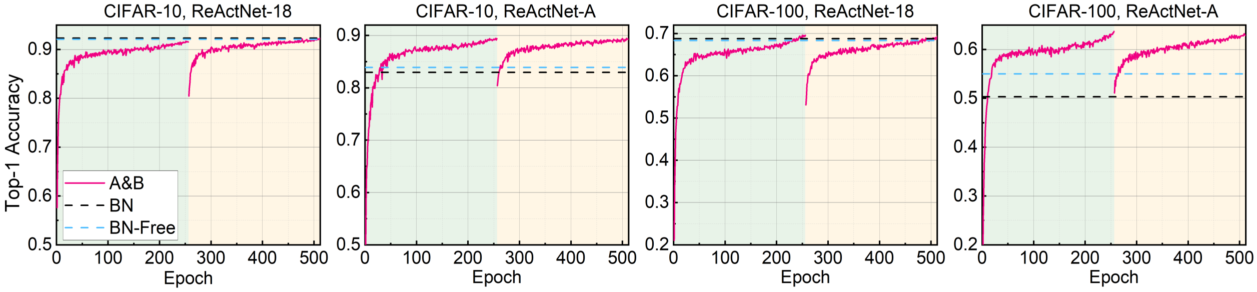


Figure 7. Top-1 accuracy on the CIFAR-10 and CIFAR-100 datasets, and comparison with two baselines.

BNNs	CIFAR10 (%)			CIFAR100 (%)		
	BN	BN-Free	A&B	BN	BN-Free	A&B
XnorNet18	90.21	79.67	89.94	65.35	53.76	64.51
BiResNet18	89.12	79.59	90.09	63.51	54.34	65.52
ReActNet18	92.31	92.08	92.30	68.78	68.34	69.35
ReActNetA	82.95	83.91	89.44	50.30	55.00	63.23

Table 4. Accuracy on the CIFAR-10 and CIFAR-100 datasets with two baselines.

Type	ImageNet (%)		CIFAR-10 (%)		CIFAR-100 (%)	
	Step1	Step2	Step1	Step2	Step1	Step2
RLeakyReLU (Slope=2^{-3})	65.00	60.25	91.67	**92.30**	69.22	69.31
RLeakyReLU (Slope=2^{-7})	64.66	60.48	91.61	92.07	67.42	67.90
Quantized RPReLU	**65.40**	**61.39**	**91.94**	91.94	**69.59**	**69.35**

Table 5. Ablation studies results of different ReLU structures.

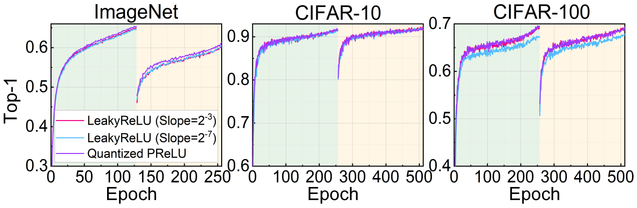


Figure 8. Ablation studies results of different ReLU structures. The findings demonstrate that employing the proposed quantized RPReLU architecture enhances performance on the ImageNet dataset by 1.14% when compared to the fixed-value LeakyReLU.

experimental results. Experiments with ReActNet-18 on CIFAR10/100 using five adjacent seeds were conducted, achieving $\mu = 92.31\%/69.37\%$, $\sigma = 6.35e-4/5.40e-4$.

4.3. Ablation Study

In the previous section, we demonstrated the effectiveness of the proposed A&B BNN and achieved competitive results. In this section, we further illustrate the necessity of the proposed quantized RPReLU and its impact on network performance through ablation studies and then explore the impact of the hyperparameter α. Due to computational constraints, all ablation studies are conducted using the ReActNet-18 structure.

Quantized RPReLU unit. To eliminate the multiplication operation introduced by RPReLU, we propose replacing PReLU with LeakyReLU or utilizing quantized PReLU. Although LeakyReLU with a fixed slope is a straightforward option, it can lead to performance degradation on complex datasets. To compare the effectiveness of different options, Tab. 5 provides a comparison between quantized RPReLU and RLeakyReLU with a fixed slope of $2^{-3}/2^{-7}$, which approximates the commonly used slopes of 0.1 and 0.01. Figure 8 depicts the Top-1 accuracies conducted on three datasets. The results exhibit a significant disparity in the impact of the activation function type between the simple dataset CIFAR and the more complex dataset ImageNet. In the case of the ImageNet dataset, the activation function type has only a 0.40% impact during the first stage of activations binarization, but the second-stage impact on the binarization of both activations and weights reaches 1.14%, demonstrating the contribution of activation function nonlinearity in enhancing network performance. The complexity of the convolution structure with both weight and activation binarized is substantially decreased. Weak nonlinearity in the activation function can lead to a deterioration in multi-layer convolutions, reducing the network's capacity to accommodate the data. Although it introduces only discrete integer powers of 2, quantized PReLU leads to a substantial improvement in activation function nonlinearity and enhances network expression. The experimental results based on the ReActNet-A structure in Tab. 1 employing BN-Free and A&B BNN also offer supporting evidence for this claim. The sole difference in network expression ability between the two resides in whether PReLU is quantized, leading to a 1.11% improvement, which further

α	ImageNet (%)		CIFAR-10 (%)		CIFAR-100 (%)	
	Step1	Step2	Step1	Step2	Step1	Step2
2^{-2}	**65.40**	**61.39**	**91.94**	**91.94**	**69.59**	**69.35**
2^{-3}	64.38	60.55	91.38	91.69	68.81	68.35

Table 6. Ablation studies results of different α values.

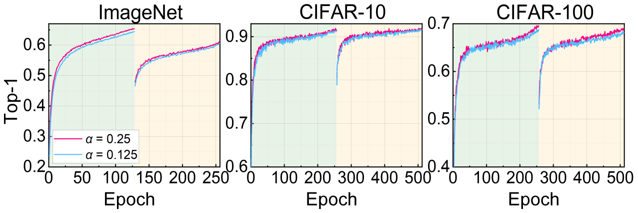


Figure 9. Ablation studies results using different α values. The results show that the value of 0.25 is consistently better than the value of 0.125.

highlights the significance of nonlinearity in ReActNet-A on ImageNet. For ReActNet-18, whether RPReLU is quantized has little effect on the results, which also applies to the small dataset CIFAR. This implies that network performance in this case is influenced by network complexity, rather than nonlinearity.

Hyperparameter α. The factor α is utilized after the convolutional layer to provide feedback to RPReLU, typically set to a small value such as 0.2. In [26, 27], α is used to control the growth rate of the variance of the residual structure, and a large α brings fast growth while a small α will weaken the effect of the residual [51], which requires a trade-off. To explore the impact of α's value on the network's performance, we set it to the closest quantized values to 0.2, specifically 2^{-2} and 2^{-3}. The results are presented in Tab. 6 and Fig. 9 shows the results of Top-1 ablation studies conducted on three datasets. The experimental results consistently demonstrate that employing $\alpha = 2^{-2}$ yields better effects, with the accuracy increasing by 0.25% to 1% compared to $\alpha = 2^{-3}$.

4.4. Visualization

The effectiveness of the proposed quantized RPReLU and its ability to enhance network nonlinearity are demonstrated through ablation studies. This section presents visualizations of the range of quantized values and their frequencies in the quantized RPReLU. Figure 10 illustrates the distribution of quantization slopes for the trained network, which quantizes RPReLU on three datasets using ReActNet-18 and ReActNet-A, respectively. The figure reveals that the quantization values are mainly distributed within the range of 2^{-19} to 2^{6}. For the small dataset CIFAR, the exponential distribution ranges from -7 to 2, with a concentration around -3. For the ImageNet dataset, the quantization distribution is wider, ranging from -19 to 7, with approximately two peaks centered around -6 and 2. A broader and more balanced distribution is more representative of non-linearities.

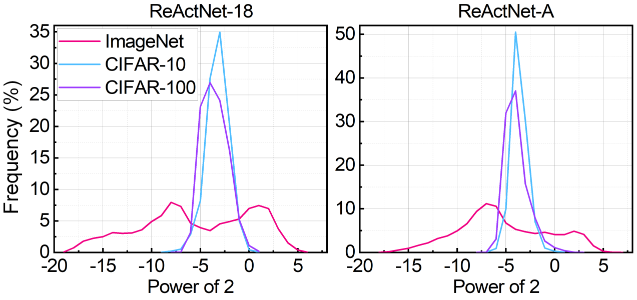


Figure 10. Visualization of the quantified RPReLU architecture shows the distribution of slopes for ReActNet-18 and ReActNet-A structures following training on CIFAR-10, CIFAR-100, and ImageNet datasets.

5. Conclusion

This study introduces a novel binary network architecture, A&B BNN, designed to perform network inference without any multiplication operation. Building upon the BN-Free architecture, we introduced the mask layer and the quantized RPReLU structure to completely remove all multiplication operations within the traditional binary network. The mask layer leverages the inherent features of BNNs and employs straightforward mathematical transformations, allowing for its direct removal during inference to eliminate associated multiplication operations. Additionally, the quantized RPReLU structure enhances efficiency by constraining its slope to an integer power of 2, employing bit operations rather than multiplications. Experimental results show that A&B BNN achieved accuracies of 92.30%, 69.35%, and 66.89% on CIFAR-10, CIFAR-100, and ImageNet datasets, respectively, which is competitive with state-of-the-art. This study introduces a novel insight for creating hardware-friendly binary neural networks.

6. Acknowledgment

This work was supported by STI 2030-Major Projects 2022ZD0209700.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[2] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR*

2015). Computational and Biological Learning Society, 2015.

[3] 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.

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[5] Xingkui Zhu, Shuchang Lyu, Xu Wang, and Qi Zhao. Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2778–2788, 2021.

[6] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar. Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7829–7833. IEEE, 2020.

[7] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4960–4964. IEEE, 2016.

[8] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[9] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[11] Lei Deng, Guoqi Li, Song Han, Luping Shi, and Yuan Xie. Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proceedings of the IEEE*, 108(4):485–532, 2020.

[12] Je-Min Hung, Cheng-Xin Xue, Hui-Yao Kao, Yen-Hsiang Huang, Fu-Chun Chang, Sheng-Po Huang, Ta-Wei Liu, Chuan-Jia Jhang, Chin-I Su, Win-San Khwa, et al. A four-megabit compute-in-memory macro with eight-bit precision based on cmos and resistive random-access memory for ai edge devices. *Nature Electronics*, 4(12):921–930, 2021.

[13] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural networks*, 111:47–63, 2019.

[14] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, Georgina Cosma, Liam P Maguire, and T Martin McGinnity. A review of learning in biologically plausible spiking neural networks. *Neural Networks*, 122:253–272, 2020.

[15] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

[16] Wenqiang Zhang, Bin Gao, Jianshi Tang, Peng Yao, Shimeng Yu, Meng-Fan Chang, Hoi-Jun Yoo, He Qian, and Huaqiang Wu. Neuro-inspired computing chips. *Nature electronics*, 3(7):371–382, 2020.

[17] Adnan Mehonic and Anthony J Kenyon. Brain-inspired computing needs a master plan. *Nature*, 604(7905):255–260, 2022.

[18] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

[19] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV*, pages 525–542. Springer, 2016.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[22] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *Proceedings of the European conference on computer vision (ECCV)*, pages 722–737, 2018.

[23] Dahyun Kim, Kunal Pratap Singh, and Jonghyun Choi. Learning architectures for binary networks. In *European conference on computer vision*, pages 575–591. Springer, 2020.

[24] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 143–159. Springer, 2020.

[25] Tianlong Chen, Zhenyu Zhang, Xu Ouyang, Zechun Liu, Zhiqiang Shen, and Zhangyang Wang. " bnn-bn=?": Training binary neural networks without batch normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4619–4629, 2021.

[26] Andrew Brock, Soham De, and Samuel L Smith. Characterizing signal propagation to close the performance gap in unnormalized resnets. In *International Conference on Learning Representations*, 2020.

[27] Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pages 1059–1071. PMLR, 2021.

[28] Zhe Xu and Ray CC Cheung. Accurate and compact convolutional neural networks with trained binarization. In *30th British Machine Vision Conference (BMVC 2019)*, 2019.

[29] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv preprint arXiv:1909.13863*, 2019.

[30] Charbel Sakr, Jungwook Choi, Zhuo Wang, Kailash Gopalakrishnan, and Naresh Shanbhag. True gradient-based training of deep binary activated neural networks via continuous binarization. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2346–2350. IEEE, 2018.

[31] Chunlei Liu, Wenrui Ding, Xin Xia, Baochang Zhang, Jiaxin Gu, Jianzhuang Liu, Rongrong Ji, and David Doermann. Circulant binary convolutional networks: Enhancing the performance of 1-bit dcnns with circulant back propagation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2691–2699, 2019.

[32] Haotong Qin, Ruihao Gong, Xianglong Liu, Mingzhu Shen, Ziran Wei, Fengwei Yu, and Jingkuan Song. Forward and backward information retention for accurate binary neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2250–2259, 2020.

[33] Peisong Wang, Xiangyu He, Gang Li, Tianli Zhao, and Jian Cheng. Sparsity-inducing binarized neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12192–12199, 2020.

[34] Ruizhou Ding, Ting-Wu Chin, Zeye Liu, and Diana Marculescu. Regularizing activation distribution for training binarized deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11408–11417, 2019.

[35] Tal Rozen, Moshe Kimhi, Brian Chmiel, Avi Mendelson, and Chaim Baskin. Bimodal-distributed binarized neural networks. *Mathematics*, 10(21):4107, 2022.

[36] Sajad Darabi, Mouloud Belbahri, Matthieu Courbariaux, and Vahid Partovi Nia. Bnn+: Improved binary network training. 2018.

[37] Hyungjun Kim, Jihoon Park, Changhun Lee, and Jae-Joon Kim. Improving accuracy of binary neural networks using unbalanced activation distribution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7862–7871, 2021.

[38] Yunqiang Li, Silvia-Laura Pintea, and Jan C van Gemert. Equal bits: Enforcing equally distributed binary network weights. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 1491–1499, 2022.

[39] Zechun Liu, Zhiqiang Shen, Shichao Li, Koen Helwegen, Dong Huang, and Kwang-Ting Cheng. How do adam and training strategies help bnns optimization. In *International conference on machine learning*, pages 6936–6946. PMLR, 2021.

[40] Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. In *International Conference on Learning Representations*, 2019.

[41] Yuzhang Shang, Dan Xu, Bin Duan, Ziliang Zong, Liqiang Nie, and Yan Yan. Lipschitz continuity retained binary neural network. In *European conference on computer vision*, pages 603–619. Springer, 2022.

[42] Jiaxin Gu, Junhe Zhao, Xiaolong Jiang, Baochang Zhang, Jianzhuang Liu, Guodong Guo, and Rongrong Ji. Bayesian optimized 1-bit cnns. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4909–4917, 2019.

[43] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. In *International Conference on Learning Representations*, 2016.

[44] Hanting Chen, Yunhe Wang, Chunjing Xu, Boxin Shi, Chao Xu, Qi Tian, and Chang Xu. Addernet: Do we really need multiplications in deep learning? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1468–1477, 2020.

[45] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.

[46] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing lstm language models. In *International Conference on Learning Representations*, 2018.

[47] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

[48] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[49] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[50] Taylor Simons and Dah-Jye Lee. A review of binarized neural networks. *Electronics*, 8(6):661, 2019.

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016.