# Improving Gradient Paths for Binary Convolutional Neural Networks

Baozhou Zhu[1]
B.Zhu-1@tudelft.nl
Peter Hofstee[1,2]
hofstee@us.ibm.com
Jinho Lee[3]
leejinho@snu.ac.kr
Zaid Alars[1]
z.al-ars@tudelft.nl

[1] Delft University of Technology, Delft, The Netherlands
[2] IBM Austin, Austin, TX, USA
[3] Seoul National University, Seoul, Korea

**Abstract**

Our starting point is a closer investigation of Bi-Real ResNet [34]. In our investigation of Bi-Real ResNet, we believe that the superiority of Bi-Real ResNet over binary ResNet requires a different explanation rather than being attributed to the representational capability. Instead, we study the gradient paths rather than representational capability for BCNNs. To our best knowledge, this is the first work to consider gradient paths for BCNNs. Improving gradient paths is realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. Regarding Bi-Real ResNet and BinaryDenseNet, the error of BCNNs decreases when the increased shortcuts improve gradient paths. In addition, we design two architectures by improving gradient paths for BCNNs: 1. Improving Gradient Paths for binary ResNet (IGP-ResNet), and 2. Improving Gradient Paths for binary DenseNet (IGP-DenseNet). Specifically, the Top-1 error of proposed IGP-ResNet37(41) and IGP-DenseNet51(53) on ImageNet gets lower than Bi-Real ResNet18(64) and BinaryDenseNet51(32) by 3.29% and 1.41%, respectively, with almost the same computational complexity.

## 1 Introduction

Convolutional Neural Networks (CNNs) have become the paradigm of choice for visual recognition. See [5, 12, 22, 33, 49] for recent often cited references. A significant amount research has been dedicated to increasing the efficiency of CNNs, including pruning [14, 52], quantization [4, 65], knowledge distillation [29, 43], and efficient network design [17]. Binarization [3, 42, 59] is the most efficient among the different bit-widths quantization methods. However, it results in a high error increase.

Binarization can be divided into two categories [69]: value approximation and structure approximation. In value approximation, we preserve the topology of the full-precision CNNs during the binarization and seek a better local minimum for binarized weights/activations by either minimizing the quantization error [2, 13, 31, 37, 39, 44, 68], improving the loss function of the network [9, 16, 25, 32, 38, 40, 41, 53, 57], or improving the quantization functions [8, 24, 26, 34, 35, 40, 60]. In structure approximation [1, 34, 67, 69], the architecture of the binary CNNs is redesigned to approximate the original full-precision CNNs. The structure approximation focuses on the architecture design principles for BCNNs, which is complementary to the value approximation. Bi-Real ResNet [34] and BinaryDenseNet [1] increase shortcuts and show significant error decrease.

A starting point of our work is a closer investigation of Bi-Real ResNet [34]. In our investigation of Bi-Real ResNet, we believe that the superiority of Bi-Real ResNet over binary ResNet requires a different explanation rather than being attributed to the representational capability. Thus, rather than representational capability, other aspects of BCNNs need to be fully explored.

In this paper, we study gradient paths rather than representational capability for BCNNs. Improving gradient paths is realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. [1] Bi-Real ResNet and BinaryDenseNet have better gradient paths and achieve lower error than binary ResNet and DenseNet. The error is not reduced when we increase shortcuts further for Bi-Real ResNet and Binary-DenseNet. In addition, we design two architectures by improving gradient paths for BCNNs: 1. Improving Gradient Paths for binary ResNet (IGP-ResNet), and 2. Improving Gradient Paths for binary DenseNet (IGP-DenseNet). Specifically, our proposed architectures have better gradient paths than Bi-Real ResNet and BinaryDenseNet. Improving gradient paths makes the gradient backpropagate more easily for BCNNs and results in an error decrease.

To our best knowledge, this is the first work to consider gradient paths for BCNNs. To make the gradient backpropagate more easily, there are efforts of employing a surrogate of the gradient [8, 11, 26, 34, 40, 60] while considering gradient paths is a new perspective for the BCNNs field.

# 2 Related work

## 2.1 Compact architecture design

Efficient architecture design has attracted lots of attention from researchers. $3 \times 3$ convolution has been replaced with $1 \times 1$ convolution in GoogLeNet [47] and SqueezeNet [20] to reduce the computational complexity. Group convolution [63], depthwise separable convolution [17], shuffle operations [36], and shift operations [54] have been shown to reduce the computational complexity of traditional convolution. Instead of relying on human experts, neural architecture search techniques [48, 56] can automatically provide optimized platform-specific architectures, achieving state-of-the-art efficiency.

## 2.2 Quantized Convolutional Neural Networks

Low bit-width quantization has been extensively explored in recent work, including reducing the gradient error [11], improving the loss function of the network [21, 70], and minimizing

[1] Exactly speaking, improving gradient paths is realized by reducing the smallest number of operations (or the second smallest number of operations or the third smallest number of operations) to compute gradient backpropagation for a gradient path.

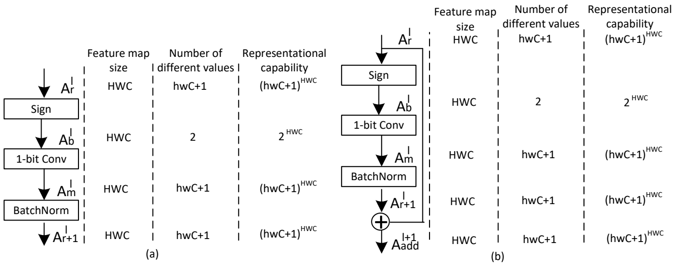


Figure 1: (a) The representational capability of each layer in BCNNs without shortcuts (b) The representational capability of each layer in BCNNs with shortcuts.

the quantization error [15]. Using neural architecture search, mixed-precision neural networks [10, 30, 55] are developed to find the optimal bit-width (i.e., precision) for weights and activations of each layer efficiently.

Improving network loss function [9, 16, 25, 32, 38, 40, 41, 53, 57], minimizing the quantization error [2, 13, 31, 37, 39, 44, 68], and improving the quantization functions [8, 24, 26, 34, 35, 40, 60] have been studied to provide a better value approximation for BCNNs. Channel-wise Interaction based Binary Convolutional Neural Network (CI-BCNN) [53] uses a reinforcement learning model to mine the interactions between channels and impose channel-wise priors to alleviate the inconsistency of signs in binary feature maps. [37] obtains significant error decrease by minimizing the discrepancy between the output of the binary and the corresponding real-valued convolution. The Information Retention Network (IR-Net) [40] has been proposed to retain the information that consists of the forward activations and backward gradients. Regarding structure approximation, [1, 34] adopts more shortcuts to reduce the error of BCNNs.

# 3 Improving gradient paths

In this section, we present a closer investigation of Bi-Real ResNet [34]. Considering gradient paths, we clarify the metric to evaluate the gradient path quality. Then, improving gradient paths can be realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. After that, we analyze the gradient paths for Bi-Real ResNet and BinaryDenseNet and introduce our proposed architectures by improving gradient paths for BCNNs. To ensure a fair comparison, we scale the number of base channels or the growth rate of our proposed architectures to have almost the same computational complexity as Bi-Real ResNet and BinaryDenseNet.

## 3.1 Investigation of Bi-Real work

**Representational capability analysis** As shown in Figure 1, $A_b^l$, $A_m^l$, $A_r^{l+1}$, and $A_{add}^{l+1}$ refer to the output of the Sign, 1-bit Conv, BatchNorm, and Add, respectively. $H$, $W$, $h$, $w$, $C$, and $l$ refer to the height and width of feature maps, the height and width of the kernels, the number of channels, and the layer index. The representational capability of a binary feature map $A_b^l$ is $\mathbb{R}(A_b^l) = 2^{HWC}$. In [34], the representational capability of the added activations (i.e., $A_{add}^{l+1} = A_r^l \oplus A_r^{l+1}$) in BCNNs with shortcuts is $(hwC+1)^{2HWC}$, which ignores the dependency between $A_r^l$ and $A_r^{l+1}$. The dependency between $A_r^l$ and $A_r^{l+1}$ is

| Model | Width | Top-1/Top-5 | Difficulty | Shortcuts |
|---|---|---|---|---|
| Bi-Real ResNet18(64) | $b=32$ | 23.01%/6.24% | $D_{d=18}$ | 13 |
| EBi-Real ResNet18(64) | $b=32$ | 23.07%/6.20% | $D_{d=18}$ | 18 |
| Bi-Real ResNet18(64) | $b=2$ | 26.71%/7.46% | $D_{d=18}$,$D_{b=2}$ | 13 |
| EBi-Real ResNet18(64) | $b=2$ | 26.86%/7.58% | $D_{d=18}$,$D_{b=2}$ | 18 |
| Bi-Real ResNet18(64) | $b=1$ | 28.48%/8.65% | $D_{d=18}$,$D_{b=1}$ | 13 |
| EBi-Real ResNet18(64) | $b=1$ | 28.74%/8.88% | $D_{d=18}$,$D_{b=1}$ | 18 |
| BinaryDenseNet51(32) | $b=32$ | 25.41%/7.30% | $D_{d=51}$ | 46 |
| EBinaryDenseNet51(32) | $b=32$ | 25.44%/7.27% | $D_{d=97}$ | 92 |
| BinaryDenseNet51(32) | $b=2$ | 26.61%/7.57% | $D_{d=51}$,$D_{b=2}$ | 46 |
| EBinaryDenseNet51(32) | $b=2$ | 26.71%/7.60% | $D_{d=97}$,$D_{b=2}$ | 92 |
| BinaryDenseNet51(32) | $b=1$ | 27.16%/7.77% | $D_{d=51}$,$D_{b=1}$ | 46 |
| EBinaryDenseNet51(32) | $b=1$ | 27.35%/7.88% | $D_{d=97}$,$D_{b=1}$ | 92 |

Table 1: Binary ResNet and DenseNet variants on CIFAR-100.

$A_r^{l+1} = \text{BatchNorm}(\text{1-bit Conv}(\text{Sign}(A_r^l)))$. Therefore, $\mathbb{R}(A_{add}^{l+1})$ should be $(hwC+1)^{HWC}$ rather than $(hwC+1)^{2HWC}$. Thus, the shortcuts will not change the representational capability of each layer in the BCNNs.

**Experiments related to increasing shortcuts further** For full-precision DCNNs, there is a training difficulty caused by their large depth $D_d$. For BCNNs, there is a training difficulty caused by the large depth $D_d$ and a training difficulty caused by the binarization $D_b$. In Table 1, there is no error decrease when comparing EBi-Real ResNet and EBinaryDenseNet to Bi-Real ResNet and BinaryDenseNet. These results are not consistent with the representational capability analysis in [34]. EBi-Real ResNet is obtained by adding more shortcuts to Bi-Real ResNet using the method in [62], and EBinaryDenseNet is obtained by adding more shortcuts to BinaryDenseNet following the method in [1]. The Top-1 error of Bi-Real ResNet will increase slightly with increasing shortcuts, by 0.06% for 32 bit-width, 0.15% for 2 bit-width, and 0.26% for 1 bit-width. Similarly, the Top-1 error of EBinaryDenseNet51 is slightly higher than that of BinaryDenseNet51 by 0.03% for 32 bit-width, 0.10% for 2 bit-width, and 0.19% for 1 bit-width.

In summary, we present a closer investigation of Bi-Real ResNet [34]. From the analysis side, [34] ignores the dependency between real-valued and binary activations when calculating the representational capability of Bi-Real ResNet. From the experiment side, there is no error decrease when we increase shortcuts further for Bi-Real ResNet and BinaryDenseNet, which cannot be explained with the representational capability. Thus, we believe that the superiority of Bi-Real ResNet over binary ResNet requires a different explanation rather than being attributed to the representational capability. Thus, other aspects of BCNNs need to be fully explored.

## 3.2 Gradient path metric

The gradient path length is adopted as the metric to evaluate the gradient path quality since the gradient information received by earlier layers from a loss at the end of the model is noisier than that received by deeper layers [18, 27, 66]. To overcome the training difficulty caused by the large depth of full-precision DCNNs, research has shown significant improvements by reducing gradient path length to improve gradient backpropagation, such as shortcut [12, 19], fractal architecture [27], deep supervision [28], and student-teacher paradigm

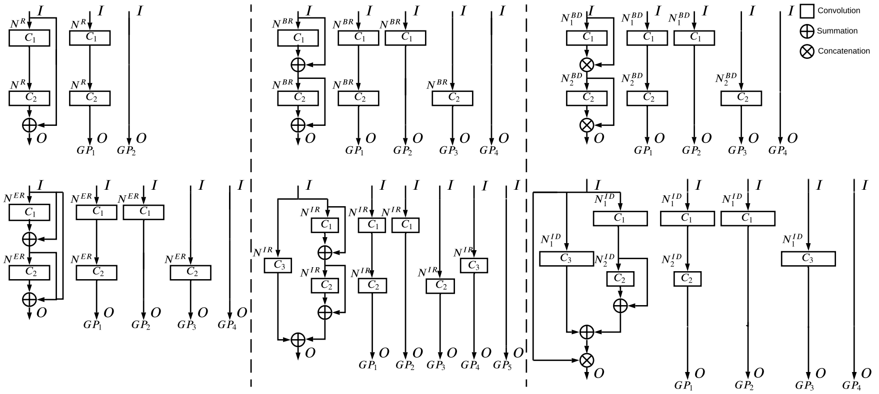


Figure 2: Gradient paths in binary ResNet and DenseNet variants. **Top left:** Gradient paths in a ResNet block. **Top middle:** Gradient paths in a Bi-Real ResNet block. **Top right:** Gradient paths in a BinaryDenseNet block. **Bottom left:** Gradient paths in a EBi-Real ResNet block. **Bottom middle:** Gradient paths in a IGP-ResNet block. **Bottom right:** Gradient paths in a IGP-DenseNet block. *GP* refers to the gradient path. The number of operations to compute gradient backpropagation for a binary convolution layer is $N^R$ in ResNet, $N^{BR}$ in Bi-Real ResNet, $N^{BD}$ in BinaryDenseNet, $N^{ER}$ in EBi-Real ResNet, $N^{IR}$ in IGP-ResNet, and $N^{ID}$ in IGP-DenseNet. It is worth noticing that BatchNorm and Relu are omitted.

[43]. A BCNN does not only suffer from the training difficulty caused by their depth, but also the training difficulty caused by their binarization. Thus, the gradient information noise in a BCNN is more than that in a full-precision DCNN. Then, we need to improve gradient paths further to make the gradient backpropagate more easily for BCNNs.

The gradient information noise accumulates with computing gradient backpropagation for gradient paths. In particular, the accumulation noise of gradient information $\Omega_a$ for a gradient path is estimated by the number of operations (i.e., multiplication and addition) $N_{ops}$. $\Omega_a = f_a(N_{ops})$, where $f_a$ is a monotonically non-decreasing function. $N_{ops} = L \times (H \times W \times C_{in} \times C_{out} \times h \times w)$, where $L$, $H$, $W$, $C_{in}$, $C_{out}$, $h$, and $w$ are the gradient path length, the height and width of feature maps, the number of input and output channels, the height and width of the weights of a convolutional layer. Thus, gradient path length can be adopted as the metric to roughly evaluate gradient path quality, while the number of operations to compute gradient backpropagation for the gradient path as the metric can make this evaluation more accurate.

In BCNNs, we use the smallest number of operations to compute gradient backpropagation for a gradient path as the metric to evaluate the gradient path quality. A smaller number of operations for a gradient path indicates less accumulation of gradient information noise. Thus, improving gradient paths can be realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. In particular, $N_{1st}$, $N_{2nd}$, and $N_{3rd}$ refer to the smallest, second smallest, and third smallest number of operations for a gradient path. Also, $L_{1st}$, $L_{2nd}$, and $L_{3rd}$ represent the shortest, second shortest, and third shortest gradient path length, which is listed. A full-precision layer accumulates much less gradient information noise than a binary convolutional layer. Thus, we consider binary convolutional layers and ignore full-precision layers.

### 3.3 Design architectures by improving gradient paths

We illustrate gradient paths in Figure 2 and evaluation of gradient path quality in Table 2, where we take binary model blocks with a depth of two layers as an example, which will work for other network architecture configurations.

**ResNet vs Bi-Real ResNet vs EBi-Real ResNet** $N_{1st}$ is 0 for Bi-Real ResNet and binary ResNet blocks. $N_{2nd}$ is $N^{BR}$ for Bi-Real ResNet and $2 \times N^{R}$ for binary ResNet blocks, where $N^{BR} = N^{R} < 2 \times N^{R}$. The gradient path quality for Bi-Real ResNet block is better than that for binary ResNet block. Then, it is reasonable that the error of Bi-Real ResNet is lower than that of binary ResNet. $N_{1st}$, $N_{2nd}$, and $N_{3rd}$ is 0, $N^{BR}$, and $2 \times N^{BR}$ for Bi-Real ResNet and EBi-Real ResNet blocks, where $N^{BR} = N^{ER}$. Thus, increasing residual connections further for Bi-Real ResNet cannot improve gradient paths and decrease error. Evaluating gradient path quality for BinaryDenseNet variants is in the appendix [2].

**Improving gradient paths with $N_{1st}$, $N_{2nd}$, and $N_{3rd}$** Improving gradient paths can be realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. We consider $N_{1st}$, $N_{2nd}$, and $N_{3rd}$ to design architectures for BCNNs. For $N_{1st}$, we can adopt shortcuts to set $N_{1st} = 0$. For $N_{2nd}$, $L_{2nd} = 1$ and improving gradient paths in an IGP-ResNet block is realized when $N^{IR} < N^{BR}$. To ensure a fair comparison, we set the computational complexity of different model blocks to be roughly the same, i.e., $M \times N^{IR} \approx 2 \times N^{BR}$. $M$ represents the number of convolutional layers in an IGP-ResNet block. Then, $M > 2$. For example, we have experimented with $M = 3$ for IGP-ResNet21(53), $M = 7$ for IGP-ResNet37(41), and $M = 15$ for IGP-ResNet69(31), which consistently shows lower error than Bi-Real ResNet18(64). Diagrams of IGP-ResNet and IGP-DenseNet with other network architecture configurations are in appendix [3].

**Bi-Real ResNet vs IGP-ResNet** $N_{1st}$ is 0 for Bi-Real ResNet and IGP-ResNet blocks. $N_{2nd}$ is $N^{BR}$ for the Bi-Real ResNet and $N^{IR}$ for IGP-ResNet blocks. To ensure a fair comparison, we set the computational complexity of different model blocks to be roughly the same, i.e., $3 \times N^{IR} \approx 2 \times N^{BR}$. Thus, $N^{BR} > N^{IR}$ and IGP-ResNet block has better gradient paths than Bi-Real ResNet block.

**BinaryDenseNet vs IGP-DenseNet** $N_{1st}$ is 0 for BinaryDenseNet and IGP-DenseNet blocks. $N_{2nd}$ is $N_1^{BD}(N_2^{BD})$ for BinaryDenseNet and $N_1^{ID}$ for IGP-DenseNet block. To ensure a fair comparison, we set the computational complexity of different model blocks to be roughly the same, i.e., $N_1^{BD} + N_2^{BD} \approx 2 \times N_1^{ID} + N_2^{ID}$. Thus, $N_1^{BD} \approx N_2^{BD} \approx N_1^{ID}$. $N_{3rd}$ is $N_1^{BD} + N_2^{BD}$ for BinaryDenseNet and $N_1^{ID} + N_2^{ID}$ for IGP-DenseNet block. $N_1^{BD} + N_2^{BD} > N_1^{ID} + N_2^{ID}$ and the gradient paths in IGP-DenseNet are better than those in BinaryDenseNet.

## 4 Experimental results

Compared with Bi-Real ResNet and BinaryDenseNet on ImageNet and CIFAR-100, our proposed architectures with various network architecture configurations consistently show

[2]This sentence is a message to the reviewers only.
[3]This sentence is a message to the reviewers only.

| Block | $N_{1st}\vert L_{1st}$ | $N_{2nd}\vert L_{2nd}$ | $N_{3rd}\vert L_{3rd}$ |
|---|---|---|---|
| ResNet | $0\vert 0$ | $2\times N^{R}\vert 2$ | $-\vert -$ |
| Bi-Real ResNet | $0\vert 0$ | $N^{BR}\vert 1$ | $-\vert -$ |
| Bi-Real ResNet | $0\vert 0$ | $N^{BR}\vert 1$ | $2\times N^{BR}\vert 2$ |
| EBi-Real ResNet | $0\vert 0$ | $N^{ER}\vert 1$ | $2\times N^{ER}\vert 2$ |
| Bi-Real ResNet | $0\vert 0$ | $N^{BR}\vert 1$ | $-\vert -$ |
| IGP-ResNet | $0\vert 0$ | $N^{IR}\vert 1$ | $-\vert -$ |
| BinaryDenseNet | $0\vert 0$ | $N_1^{BD}(N_2^{BD})\vert 1$ | $N_1^{BD}+N_2^{BD}\vert 2$ |
| IGP-DenseNet | $0\vert 0$ | $N_1^{ID}\vert 1$ | $N_1^{ID}+N_2^{ID}\vert 2$ |

Table 2: Evaluation of gradient path quality for binary model blocks. $(\cdot|\cdot)$ refers to the smallest number of operations to compute gradient backpropagation for a gradient path and the shortest gradient path length. For example, $0|0$ indicates that the smallest operation number and the shortest gradient path length for a binary model block are 0. If $N_{1st}$ is the same for two different model blocks, we compare $N_{2nd}$. Similar, if $N_{1st}$ and $N_{2nd}$ are the same, we compare $N_{3rd}$.

significant performance improvement. In addition, we demonstrate the essential role of the gradient path that requires the smallest number of operations to compute gradient backpropagation, which supports that the key of our proposal is to design architectures by improving gradient paths for BCNNs. Experimental details are in the appendix [4].

## 4.1 Experimental results on ImageNet

**ResNet variants on ImageNet** As shown in Table 3, we present the experimental results of IGP-ResNet on ImageNet. Our IGP-ResNet variants with various network architecture configurations, including IGP-ResNet21(53), IGP-ResNet37(41), and IGP-ResNet69(31), consistently achieve significant performance improvement compared with Bi-Real ResNet18. In particular, IGP-ResNet37(41) and IGP-ResNet41(48) reduce the Top-1 error by 3.29% and 1.12% compared with Bi-Real ResNet18(64) and Bi-Real ResNet34(64), respectively. Regarding the computational complexity, IGP-ResNet37(41) increases the run-time memory size by 10.44MB but saves the number of parameters by 0.94Mbit and the number of Flops by $0.36\times 10^8$ (21.95%) compared with Bi-Real ResNet18(64). Similarly, the number of parameters and the number of Flops required for our proposed IGP-ResNet41(48) are 0.67Mbit and $0.29\times 10^8$ less than those needed for Bi-Real ResNet34(64).

**DenseNet variants on ImageNet** As shown in Table 3, we present the experimental results of our IGP-DenseNet on ImageNet. The Top-1 error of IGP-DenseNet51(53) and IGP-DenseNet69(48) is 1.41% and 1.06% lower than those of BinaryDenseNet51(32) and BinaryDenseNet69(32), respectively. In terms of the computational complexity, IGP-DenseNet51(53) and IGP-DenseNet69(48) require $0.27\times 10^8$ Flops and $0.24\times 10^8$ Flops less compared with BinaryDenseNet51(32) and BinaryDenseNet69(32), respectively, while they save the number of parameters by 0.37Mbit and 0.37Mbit, respectively, and decrease the run-time memory size by 52.98MB and 77.07MB, respectively.

[4]This sentence is a message to the reviewers only.

| Model | Top-1 | Top-5 | Storage | Computation | Run-time memory |
|---|---|---|---|---|---|
| Bi-Real ResNet18(64) | 40.42% | 18.29% | 33.18Mbit | $1.64 \times 10^8$Flops | 154.14MB |
| IGP-ResNet21(53) | 37.58% | 16.06% | 32.63Mbit | $1.46 \times 10^8$Flops | 170.20MB |
| IGP-ResNet37(41) | **37.13%** | **15.63%** | 32.24Mbit | $1.28 \times 10^8$Flops | 164.58MB |
| IGP-ResNet69(31) | 37.66% | 15.77% | 32.16Mbit | $1.14 \times 10^8$Flops | 149.32MB |
| Bi-Real ResNet34(64) | 36.74% | 15.36% | 43.28Mbit | $1.93 \times 10^8$Flops | 154.14MB |
| IGP-ResNet41(48) | **35.62%** | **14.53%** | 42.61Mbit | $1.64 \times 10^8$Flops | 154.14MB |
| IGP-ResNet77(35) | 36.66% | 15.07% | 41.53Mbit | $1.44 \times 10^8$Flops | 140.49MB |
| BinaryDenseNet51(32) | 38.14% | 16.80% | 34.80Mbit | $2.70 \times 10^8$Flops | 359.66MB |
| IGP-DenseNet51(53) | **36.73%** | **15.54%** | 34.53Mbit | $2.97 \times 10^8$Flops | 306.68MB |
| BinaryDenseNet69(32) | 36.26% | 15.24% | 41.95Mbit | $2.82 \times 10^8$Flops | 359.66MB |
| IGP-DenseNet69(48) | **35.20%** | **14.59%** | 41.52Mbit | $3.06 \times 10^8$Flops | 282.59MB |

Table 3: Binary ResNet and DenseNet variants on ImageNet. There are four blocks in this Table. **First block:** ResNet18(64) and IGP-ResNet variants to compare with ResNet18(64). **Second block:** ResNet34(64) and IGP-ResNet variants to compare with ResNet34(64). **Third block:** BinaryDenseNet51(32) and IGP-DenseNet variants to compare with BinaryDenseNet51(32). **Fourth block:** BinaryDenseNet69(32) and IGP-DenseNet variants to compare with BinaryDenseNet69(32).

## 4.2 Comparison to State-of-the-Art

In Table 4, we compare with state-of-the-art BCNNs on ImageNet. Except for the FULW-ResNet18 [57], ProxyResNet18 [13], Real-to-bin ResNet18 [37], ReActNet-ResNet18 [35], and DIR-Net$^2$-ResNet18 [41], the Top-1 error of IGP-ResNet37(41), IGP-ResNet41(48), IGP-DenseNet51(53), and IGP-DenseNet69(48) achieve 37.13%, 35.62%, 36.73%, and 35.20%, respectively, and are lower other binary ResNet and DenseNet variants by a large margin.

Here we have the following clarifications for the fact that the error of our proposed architectures does not achieve the lowest among all the references.

FULW-ResNet18 [57] explores the role of $W$ in training besides acting as a latent variable. ProxyResNet18 [13] reduces the weights quantization error by introducing an appropriate proxy matrix. Real-to-bin ResNet18 [37] minimizes the discrepancy between the output of the binary and the corresponding real-valued convolution. ReActNet-ResNet18 [35] proposes to generalize the traditional Sign and PreLU functions, denoted as RSign and RPReLU for the respective generalized functions. DIR-Net$^2$-ResNet18 [41] introduces a novel DIR-Net that retains the information during the forward/backward propagation of BNNs. All these references [13, 35, 37, 41, 57] belong to value approximation since they preserve the topology of the full-precision CNNs during the binarization and try to seek a better local minimum for binarized weights/activations. But, our work is about architecture design and belongs to structure approximation, which is complementary to the value approximation. Thus, it is reasonable to expect that we can improve the performance of BCNNs in these references further with our proposed architectures. Given a stronger BCNN baseline trained with a more advanced value approximation from these references, the error of our proposed architectures can decrease and achieve better performance.

Besides, our proposed architectures outperform all the references about the architecture design, even automated BNAS-E [23]. Almost all our experiments use the baseline of Bi-Real ResNet and BinaryDenseNet to show the effectiveness of our proposed architecture design principle since improving gradient paths for Bi-Real ResNet and BinaryDenseNet indeed decrease their error.

| Model | Top-1/Top-5 | Storage | Computation |
|---|---|---|---|
| BNN ResNet18#* [7] | 57.80%/30.80% | 27.9Mbit | $0.14 \times 10^9$Flops |
| XNOR ResNet18#* [42] | 48.80%/26.80% | 28.0Mbit | $0.14 \times 10^9$Flops |
| $S^2$-Bi-Real ResNet18* [45] | 48.76%/24.11% | 33.2Mbit | $0.16 \times 10^9$Flops |
| Bin ResNet18#* [58] | 45.80%/22.10% | 27.9Mbit | $0.14 \times 10^9$Flops |
| TBN-ResNet18#* [50] | 44.40%/25.80% | 27.9Mbit | $0.17 \times 10^9$Flops |
| Bi-Real ResNet18* [34] | 43.60%/20.50% | 33.2Mbit | $0.16 \times 10^9$Flops |
| CI-Net ResNet18#* [53] | 43.30%/19.90% | 27.9Mbit | $0.14 \times 10^9$Flops |
| XNOR-Net++ ResNet18#* [2] | 42.90%/20.10% | 28.0Mbit | $0.14 \times 10^9$Flops |
| IR-ResNet18* [40] | 41.90%/20.00% | 33.1Mbit | $0.16 \times 10^9$Flops |
| BNAS-E* [23] | 41.24%/19.39% | 33.1Mbit | $0.16 \times 10^9$Flops |
| Bi-Real ResNet18(64) [34] | 40.42%/18.29% | 33.2Mbit | $0.16 \times 10^9$Flops |
| Si-ResNet18* [51] | 40.30%/18.20% | 33.2Mbit | $0.16 \times 10^9$Flops |
| CI-Net ResNet18* [53] | 40.10%/17.80% | 33.2Mbit | $0.16 \times 10^9$Flops |
| RBNN-ResNet18* [31] | 40.10%/18.10% | 33.2Mbit | $0.16 \times 10^9$Flops |
| FT-ResNet18* [46] | 39.80%/– | 33.2Mbit | $0.16 \times 10^9$Flops |
| DGRL-ResNet18 (K=1)* [61] | 39.55%/– | 33.2Mbit | $0.16 \times 10^9$Flops |
| UaBNN-ResNet18* [64] | 39.40%/17.80% | 33.2Mbit | $0.16 \times 10^9$Flops |
| BinaryDuo ResNet18* [24] | 39.10%/17.40% | 33.2Mbit | $0.16 \times 10^9$Flops |
| ReActNet-18 (BN-Free)* [6] | 38.90%/– | 33.2Mbit | $0.16 \times 10^9$Flops |
| SA-BNN-ResNet18* [32] | 38.30%/17.20% | 33.2Mbit | $0.16 \times 10^9$Flops |
| IA-BNN-ResNet18* [25] | 37.20%/15.70% | 33.2Mbit | $0.16 \times 10^9$Flops |
| IGP-ResNet37(41) | 37.13%/15.63% | 32.2Mbit | $0.13 \times 10^9$Flops |
| FULW-ResNet18* [57] | **36.60%/15.40%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| ProxyResNet18* [13] | **36.30%/15.20%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| Real-to-bin ResNet18*[37] | **34.60%/13.80%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| ReActNet-ResNet18* [35] | **34.10%/–** | 33.2Mbit | $0.16 \times 10^9$Flops |
| DIR-$Net^2$-ResNet18* [41] | **33.90%/13.60%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| TBN-ResNet34#* [50] | 41.80%/19.00% | 38.0Mbit | $0.23 \times 10^9$Flops |
| Bi-Real ResNet34* [34] | 37.80%/16.10% | 43.3Mbit | $0.19 \times 10^9$Flops |
| Bi-Real ResNet34(64) [34] | 36.74%/15.36% | 43.3Mbit | $0.19 \times 10^9$Flops |
| IGP-ResNet41(48) | **35.62%/14.53%** | 42.6Mbit | $0.16 \times 10^9$Flops |
| BinaryDenseNet51(32)* [1] | 39.30%/17.60% | 34.8Mbit | $0.27 \times 10^9$Flops |
| BinaryDenseNet51(32) [1] | 38.14%/16.80% | 34.8Mbit | $0.27 \times 10^9$Flops |
| IGP-DenseNet51(53) | **36.73%/15.54%** | 34.5Mbit | $0.30 \times 10^9$Flops |
| BinaryDenseNet69(32)* [1] | 37.50%/16.10% | 42.0Mbit | $0.28 \times 10^9$Flops |
| BinaryDenseNet69(32) [1] | 36.26%/15.24% | 42.0Mbit | $0.28 \times 10^9$Flops |
| IGP-DenseNet69(48) | **35.20%/14.59%** | 41.5Mbit | $0.31 \times 10^9$Flops |
| Full-precision ResNet18* | 30.70%/10.80% | 374.1Mbit | $1.81 \times 10^9$Flops |
| Full-precision ResNet34* | 26.80%/8.60% | 697.3Mbit | $3.66 \times 10^9$Flops |

Table 4: Comparison with state-of-the-art methods on ImageNet. * refers to the baseline from the published papers. # indicates the downsampling layers are binarized.

# 5 Conclusion

We present a closer investigation of Bi-Real ResNet [34] and believe that the superiority of Bi-Real ResNet over binary ResNet requires a different explanation rather than being attributed to the representational capability. Instead, we study gradient paths rather than representational capability for BCNNs. Improving gradient paths is realized by reducing the smallest number of operations to compute gradient backpropagation for a gradient path. Under a given computational complexity budget, the Top-1 error of our proposed architectures is lower than the state-of-the-art Bi-Real ResNet18(64) by 3.29%, Bi-Real ResNet34(64) by 1.12%, BinaryDenseNet51(32) by 1.41%, and BinaryDenseNet69(32) by 1.06% on ImageNet classification.

This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

# References

[1] Joseph Bethge, Haojin Yang, Marvin Bornstein, and Christoph Meinel. Binarydensenet: Developing an architecture for binary neural networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[2] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv*, pages arXiv–1909, 2019.

[3] Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. High-capacity expert binary networks. *arXiv preprint arXiv:2010.03558*, 2020.

[4] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13169–13178, 2020.

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[6] 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.

[7] 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.

[8] Sajad Darabi, Mouloud Belbahri, Matthieu Courbariaux, and Vahid Partovi Nia. Bnn+: Improved binary network training. *arXiv preprint arXiv:1812.11800*, 2018.

[9] Ruizhou Ding, Ting-Wu Chin, Zeye Liu, and Diana Marculescu. Regularizing activation distribution for training binarized deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11408–11417, 2019.

[10] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 293–302, 2019.

[11] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4852–4861, 2019.

[12] 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.

[13] Xiangyu He, Zitao Mo, Ke Cheng, Weixiang Xu, Qinghao Hu, Peisong Wang, Qingshan Liu, and Jian Cheng. Proxybnn: Learning binarized neural networks via proxy matrices. In *European Conference on Computer Vision*, pages 223–241. Springer, 2020.

[14] Yang He, Yuhang Ding, Ping Liu, Linchao Zhu, Hanwang Zhang, and Yi Yang. Learning filter pruning criteria for deep convolutional neural networks acceleration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[15] Zhezhi He and Deliang Fan. Simultaneously optimizing weight and quantizer of ternary neural network using truncated gaussian approximation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11438–11446, 2019.

[16] Lu Hou, Quanming Yao, and James T Kwok. Loss-aware binarization of deep networks. *arXiv preprint arXiv:1611.01600*, 2016.

[17] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1314–1324, 2019.

[18] Hanzhang Hu, Debadeepta Dey, Allison Del Giorno, Martial Hebert, and J Andrew Bagnell. Log-densenet: How to sparsify a densenet. *arXiv preprint arXiv:1711.00002*, 2017.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[20] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[21] Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Jae-Joon Han, Youngjun Kwak, Sung Ju Hwang, and Changkyu Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4350–4359, 2019.

[22] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, pages 1–62, 2020.

[23] Dahyun Kim, Kunal Pratap Singh, and Jonghyun Choi. Bnas v2: Learning architectures for binary networks with empirical improvements. *arXiv preprint arXiv:2110.08562*, 2021.

[24] Hyungjun Kim, Kyungsu Kim, Jinseok Kim, and Jae-Joon Kim. Binaryduo: Reducing gradient mismatch in binary activation network by coupling binary activations. *arXiv preprint arXiv:2002.06517*, 2020.

[25] 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.

[26] Fayez Lahoud, Radhakrishna Achanta, Pablo Márquez-Neila, and Sabine Süsstrunk. Self-binarizing networks. *arXiv preprint arXiv:1902.00730*, 2019.

[27] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016.

[28] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pages 562–570, 2015.

[29] Changlin Li, Jiefeng Peng, Liuchun Yuan, Guangrun Wang, Xiaodan Liang, Liang Lin, and Xiaojun Chang. Block-wisely supervised neural architecture search with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1989–1998, 2020.

[30] Yuhang Li, Wei Wang, Haoli Bai, Ruihao Gong, Xin Dong, and Fengwei Yu. Efficient bitwidth search for practical mixed precision neural network. *arXiv preprint arXiv:2003.07577*, 2020.

[31] Mingbao Lin, Rongrong Ji, Zihan Xu, Baochang Zhang, Yan Wang, Yongjian Wu, Feiyue Huang, and Chia-Wen Lin. Rotated binary neural network. *Advances in neural information processing systems*, 33:7474–7485, 2020.

[32] Chunlei Liu, Peng Chen, Bohan Zhuang, Chunhua Shen, Baochang Zhang, and Wenrui Ding. Sa-bnn: State-aware binary neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2091–2099, 2021.

[33] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[34] 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.

[35] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *European Conference on Computer Vision*, pages 143–159. Springer, 2020.

[36] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[37] Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. *arXiv preprint arXiv:2003.11535*, 2020.

[38] Asit Mishra and Debbie Marr. Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy. *arXiv preprint arXiv:1711.05852*, 2017.

[39] Asit Mishra, Eriko Nurvitadhi, Jeffrey J Cook, and Debbie Marr. Wrpn: wide reduced-precision networks. *arXiv preprint arXiv:1709.01134*, 2017.

[40] 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.

[41] Haotong Qin, Xiangguo Zhang, Ruihao Gong, Yifu Ding, Yi Xu, et al. Distribution-sensitive information retention for accurate binary neural network. *arXiv preprint arXiv:2109.12338*, 2021.

[42] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

[43] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[44] Mingzhu Shen, Kai Han, Chunjing Xu, and Yunhe Wang. Searching for accurate binary neural architectures. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[45] Zhiqiang Shen, Zechun Liu, Jie Qin, Lei Huang, Kwang-Ting Cheng, and Marios Savvides. S2-bnn: Bridging the gap between self-supervised real and 1-bit neural networks via guided distribution calibration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2165–2174, 2021.

[46] Zhuo Su, Linpu Fang, Deke Guo, Dewen Hu, Matti Pietikäinen, and Li Liu. Ftbnn: Rethinking non-linearity for 1-bit cnns and going beyond. *arXiv preprint arXiv:2010.09294*, 2020.

[47] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[48] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[49] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[50] Diwen Wan, Fumin Shen, Li Liu, Fan Zhu, Jie Qin, Ling Shao, and Heng Tao Shen. Tbn: Convolutional neural network with ternary inputs and binary weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 315–332, 2018.

[51] 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*, pages 12192–12199, 2020.

[52] Tianzhe Wang, Kuan Wang, Han Cai, Ji Lin, Zhijian Liu, Hanrui Wang, Yujun Lin, and Song Han. Apq: Joint search for network architecture, pruning and quantization policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2078–2087, 2020.

[53] Ziwei Wang, Jiwen Lu, Chenxin Tao, Jie Zhou, and Qi Tian. Learning channel-wise interactions for binary convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 568–577, 2019.

[54] Bichen Wu, Alvin Wan, Xiangyu Yue, Peter Jin, Sicheng Zhao, Noah Golmant, Amir Gholaminejad, Joseph Gonzalez, and Kurt Keutzer. Shift: A zero flop, zero parameter alternative to spatial convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9127–9135, 2018.

[55] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*, 2018.

[56] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019.

[57] Weixiang Xu, Qiang Chen, Xiangyu He, Peisong Wang, and Jian Cheng. Improving binary neural networks through fully utilizing latent weights. *arXiv preprint arXiv:2110.05850*, 2021.

[58] Zhe Xu and Ray CC Cheung. Accurate and compact convolutional neural networks with trained binarization. *arXiv preprint arXiv:1909.11366*, 2019.

[59] Ping Xue, Yang Lu, Jingfei Chang, Xing Wei, and Zhen Wei. Self-distribution binary neural networks. *arXiv preprint arXiv:2103.02394*, 2021.

[60] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7308–7316, 2019.

[61] Jianming Ye, Jingdong Wang, and Shiliang Zhang. Distillation-guided residual learning for binary convolutional neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[62] Ke Zhang, Miao Sun, Tony X Han, Xingfang Yuan, Liru Guo, and Tao Liu. Residual networks of residual networks: Multilevel residual networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(6):1303–1314, 2017.

[63] Zhaoyang Zhang, Jingyu Li, Wenqi Shao, Zhanglin Peng, Ruimao Zhang, Xiaogang Wang, and Ping Luo. Differentiable learning-to-group channels via groupable convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3542–3551, 2019.

[64] Junhe Zhao, Linlin Yang, Baochang Zhang, Guodong Guo, and David Doermann. Uncertainty-aware binary neural networks. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021.

[65] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[66] Ligeng Zhu, Ruizhi Deng, Michael Maire, Zhiwei Deng, Greg Mori, and Ping Tan. Sparsely aggregated convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 186–201, 2018.

[67] Hofstee Peter Zhu et al., Al-Ars Zaid. Nasb: Neural architecture search for binary convolutional neural networks. *arXiv preprint arXiv:2008.03515*, 2020.

[68] Wei Pan Zhu et al., Zaid Al-Ars. Towards lossless binary convolutional neural networks using piecewise approximation, 2020.

[69] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–422, 2019.

[70] Bohan Zhuang, Lingqiao Liu, Mingkui Tan, Chunhua Shen, and Ian Reid. Training quantized neural networks with a full-precision auxiliary module. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1488–1497, 2020.