# Improving Gradient Paths for Binary Convolutional Neural Networks

Baozhou Zhu [1 2] Peter Hofstee [1 3] Jinho Lee [4] Zaid Alars [1]

[1]Delft University of Technology, Delft, The Netherlands [2]National University of Defense Technology, Changsha, China [3]IBM Austin, Austin, TX, USA [4]Seoul National University, Seoul, Korea

## Abstract

Our starting point is a closer investigation of Bi-Real ResNet [1]. 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.

## Introduction

Convolutional Neural Networks (CNNs) have become the paradigm of choice for visual recognition. A significant amount research has been dedicated to increasing the efficiency of CNNs, including pruning, quantization, knowledge distillation, and efficient network design. Binarization is the most efficient among the different bit-widths quantization methods. However, it results in a high error increase. 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 while considering gradient paths is a new perspective for the BCNNs field.

## Investigation of Bi-Real work

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 [1], 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 $A_r^{l+1} = \text{BatchNorm}(\text{1-bit Conv}(\text{Sign}(A_r^l)))$. 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.

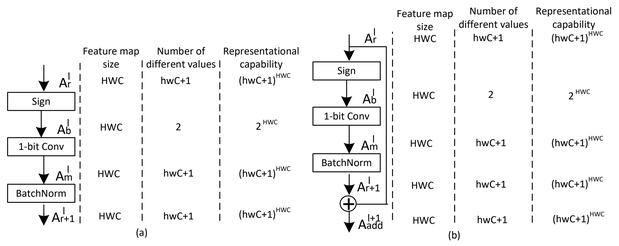


Figure 1. (a) The representational capability of each layer in BCNNs without shortcuts (b) The representational capability of each layer in BCNNs with shortcuts.

## Gradient path metric

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.

## Comparison to State-of-the-Art

In Table 1, we compare with state-of-the-art BCNNs on ImageNet. Except for the FULW-ResNet18, ProxyResNet18, Real-to-bin ResNet18, ReActNet-ResNet18, and DIR-Net$^2$-ResNet18, 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 explores the role of $W$ in training besides acting as a latent variable. ProxyResNet18 reduces the weights quantization error by introducing an appropriate proxy matrix. Real-to-bin ResNet18 minimizes the discrepancy between the output of the binary and the corresponding real-valued convolution. ReActNet-ResNet18 proposes to generalize the traditional Sign and PreLU functions, denoted as RSign and RPReLU for the respective generalized functions. DIR-Net$^2$-ResNet18 introduces a novel DIR-Net that retains the information during the forward/backward propagation of BNNs. All these references 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. 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#* | 57.80%/30.80% | 27.9Mbit | $0.14 \times 10^9$Flops |
| XNOR ResNet18#* | 48.80%/26.80% | 28.0Mbit | $0.14 \times 10^9$Flops |
| S$^2$-Bi-Real ResNet18* | 48.76%/24.11% | 33.2Mbit | $0.16 \times 10^9$Flops |
| Bin ResNet18#* | 45.80%/22.10% | 27.9Mbit | $0.14 \times 10^9$Flops |
| TBN-ResNet18#* | 44.40%/25.80% | 27.9Mbit | $0.17 \times 10^9$Flops |
| Bi-Real ResNet18* | 43.60%/20.50% | 33.2Mbit | $0.16 \times 10^9$Flops |
| CI-Net ResNet18#* | 43.30%/19.90% | 27.9Mbit | $0.14 \times 10^9$Flops |
| XNOR-Net++ ResNet18#* | 42.90%/20.10% | 28.0Mbit | $0.14 \times 10^9$Flops |
| IR-ResNet18* | 41.90%/20.00% | 33.1Mbit | $0.16 \times 10^9$Flops |
| BNAS-E* | 41.24%/19.39% | 33.1Mbit | $0.16 \times 10^9$Flops |
| Bi-Real ResNet18(64) | 40.42%/18.29% | 33.2Mbit | $0.16 \times 10^9$Flops |
| Si-ResNet18* | 40.30%/18.20% | 33.2Mbit | $0.16 \times 10^9$Flops |
| CI-Net ResNet18* | 40.10%/17.80% | 33.2Mbit | $0.16 \times 10^9$Flops |
| RBNN-ResNet18* | 40.10%/18.10% | 33.2Mbit | $0.16 \times 10^9$Flops |
| FT-ResNet18* | 39.80%/— | 33.2Mbit | $0.16 \times 10^9$Flops |
| DGRL-ResNet18 (K=1)* | 39.55%/— | 33.2Mbit | $0.16 \times 10^9$Flops |
| UaBNN-ResNet18* | 39.40%/17.80% | 33.2Mbit | $0.16 \times 10^9$Flops |
| BinaryDuo ResNet18* | 39.10%/17.40% | 33.2Mbit | $0.16 \times 10^9$Flops |
| ReActNet-18 (BN-Free)* | 38.90%/— | 33.2Mbit | $0.16 \times 10^9$Flops |
| SA-BNN-ResNet18* | 38.30%/17.20% | 33.2Mbit | $0.16 \times 10^9$Flops |
| IA-BNN-ResNet18* | 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* | **36.60%**/**15.40%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| ProxyResNet18* | **36.30%**/**15.20%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| Real-to-bin ResNet18* | **34.60%**/**13.80%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| ReActNet-ResNet18* | **34.10%**/— | 33.2Mbit | $0.16 \times 10^9$Flops |
| DIR-Net$^2$-ResNet18* | **33.90%**/**13.60%** | 33.2Mbit | $0.16 \times 10^9$Flops |
| TBN-ResNet34#* | 41.80%/19.00% | 38.0Mbit | $0.23 \times 10^9$Flops |
| Bi-Real ResNet34* | 37.80%/16.10% | 43.3Mbit | $0.19 \times 10^9$Flops |
| Bi-Real ResNet34(64) | 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)* | 39.30%/17.60% | 34.8Mbit | $0.27 \times 10^9$Flops |
| BinaryDenseNet51(32) | 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)* | 37.50%/16.10% | 42.0Mbit | $0.28 \times 10^9$Flops |
| BinaryDenseNet69(32) | 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 1. Comparison with state-of-the-art methods on ImageNet. * refers to the baseline from the published papers. # indicates the downsampling layers are binarized.

## Design architectures by improving gradient paths

We illustrate gradient paths in Figure 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.

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).

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.

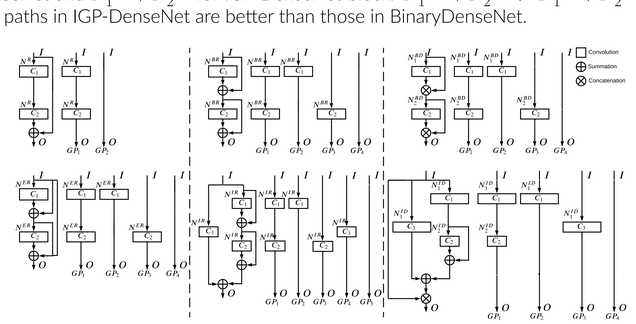


Figure 2. Gradient paths in binary ResNet and DenseNet variants. **Top left:** Gradient paths in a ResNet block. **Top middle:** Gradient paths in a Bi-Real ResNet block. **Top right:** Gradient paths in a BinaryDenseNet block. **Bottom left:** Gradient paths in a EBi-Real ResNet block. **Bottom middle:** 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.

## Conclusion

We present a closer investigation of Bi-Real ResNet 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.

## References

[1] 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.