A Simple and General Binarization Method for Image Restoration Neural Networks

Mengxue Wang WANGMENGXUE1203@FOXMAIL.COM
Huazhong University of Science and Technology
Yue Zhang YUE1554415@163.COM and **Xiaodong Zhang** SANCTUM.ZHANG@GMAIL.COM
Wuhan likelihood technology Co.,Ltd
Run Min MINRUN@HUST.EDU.CN
Huazhong University of Science and Technology

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

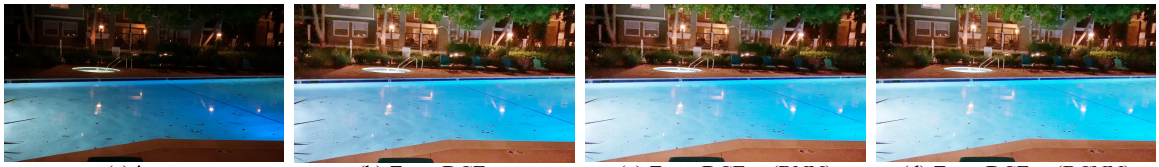
With the advancement of deep learning techniques, image restoration (IR) performance has improved significantly. However, these techniques often come with high computational costs, which pose challenges in meeting the processing latency requirements of resource-constrained hardware in edge computer vision systems. To address this issue, we propose a simple binarization technique and an efficient training strategy called Gentle Approximation Method (GAM) to extend the application of binary neural networks (BNNs) to various IR tasks, including low-light image enhancement, deraining, denoising, and super-resolution. Our results demonstrate the effectiveness of our method in binarizing full-precision deep neural networks. By binarizing these networks, we achieve a significant reduction in computational and memory demands while maintaining satisfactory performance. For instance, in the denoising task, the FLOPs can be reduced to only 3% of the original network while preserving most of the performance.

Keywords: Binary Neural Network; Image Restoration;

1. Introduction

Image restoration (IR) addresses the challenge of recovering clean latent images from degraded observations, which is a significant problem in image processing, understanding, and representation. IR has widespread applications in fields like medical imaging, astronomy, and microscop. Over the past century, IR has received considerable attention due to its impact on subsequent computer vision tasks such as classification and detection. In recent decades, deep neural networks (DNNs) have revolutionized computer vision tasks, including IR, by replacing traditional methods.Su et al. (2022) However, the computational and memory demands of DNN-based IR algorithms pose challenges for edge computer vision systems and resource-limited devices.

Considerable methods have been proposed to address this problem, including compact models, tensor decomposition, data quantization, and network sparsificationDeng et al. (2020). Among these, Binary Neural Networks (BNNs)Courbariaux et al. (2016) and Binary Complex Neural Networks (BCNNs)Li et al. (2021) have emerged as promising solutions



(a) input (b) Zero_DCE++ (c) Zero_DCE++(BNN) (d) Zero_DCE++(BCNN)

Figure 1: An example of performance from float point network and binarized networks. Zero-DCE++Guo et al. (2020) is a low-light enhancement network. The binarized networks have similar effect with the original network with more than 85% reduction in computation and more than 83% in model size.

for resource-limited edge devices. These approaches effectively reduce memory and access requirements by replacing arithmetic operations with bit-wise operations.

BNN and BCNN aim to represent activation and weight using one or two bits, replacing floating-point multiply-accumulate operations (MAdds) with `xnor` and `popcnt` operationsRastegari et al. (2016). While BNN has achieved success in various semantic-level tasks such as classification, recognition, and segmentationBethge et al. (2021); Liu et al. (2020); Sun et al. (2018); Wang et al. (2020); Zhuang et al. (2019), its application in image restoration (IR) tasks, apart from super-resolution (SR) neural networksMa et al. (2019); Xin et al. (2020); Jiang et al. (2021); Huang et al. (2021); Nie et al. (2022); Xia B (2022), remains relatively unexplored.

This work proposes a simple and universal strategy for binarizing IR neural networks, involving architecture modification and a novel training strategy called the Gentle Approximation Method (GAM). We consolidate and abstract methods from prior studies to derive a universal approach for architecture modification. In GAM, we depart from conventional techniques and use floating-point numbers to simulate the distribution in BNN and BCNN during training, ensuring optimal gradient descent. Our work achieves universal binarization of IR neural networks, yielding remarkable results in four tasks. Notably, we introduce binarization in low-light enhancement and rain removal networks for the first time. Figure 1 provides a comparison between the floating-point and binarized networks in low-light enhancement, demonstrating resource savings with similar performance.

This work makes the following contributions. First, it achieves universal binarization of IR neural networks, demonstrating the significant potential of BNN and BCNN in IR applications through experiments on four different tasks. Second, it proposes a simple and universal binarization technique that transforms full-precision IR networks into standard BNN or BCNN architectures. Finally, it introduces an efficient training strategy called the Gentle Approximation Method (GAM) to address the issue of non-convergence in traditional training strategies without introducing additional loss functions or parameters.

2. Related Work

2.1. Image Restoration Neural Network

IR encompasses tasks like super-resolution, deblurring, denoising, deraining, dehazing, and more, aiming to enhance image qualitySu et al. (2022). Traditional methods, such as maximum likelihood or Bayesian algorithms, were commonly used in the pre-deep learning era. However, deep learning methods have surpassed these approaches in performance and ease of implementation, particularly with the use of high-level hardware resources.

Two main baseline models for IR neural networks are Convolutional Neural Network (CNN) and Generative Adversarial Network (GAN)Su et al. (2022). HI-Net achieves high-performance denoising, deblurring, and deraining using instance normalization and CNNChen et al. (2021). g-UNet is a dehazing network based on U-Net and residual blocks, commonly used structures in CNNSong et al. (2022). GANs have gained popularity in IR tasks for generating realistic images with fine texture. PULSE focuses on facial super-resolution with a scale-factor of up to 64×Menon et al. (2020). DeblurGANv2 strikes a balance between performance and computational efficiency using Feature Dynamic NetworksKupyn et al. (2019).

Although GANs offer high capacity and compatibility, they come with larger and deeper networks and training challenges. The dynamic characteristics of some GANs demand higher hardware requirements, even during inference. Therefore, this work primarily focuses on binarizing IR tasks based on CNNs.

2.2. Binary Neural Network

BNN, introduced in Courbariaux et al. (2016), represents weights and activations using 1-bit values. It offers reduced computational complexity and hardware-friendly characteristics while maintaining considerable accuracy compared to full-precision networks. Various methods have been proposed to enhance BNN's performance and address training challenges. One method focuses on reducing information loss during binarization by introducing additional parameters, as seen in XNOR-NetRastegari et al. (2016), Dorefa-NetZhou et al. (2016), ABC-NetLin et al. (2017), Bi-RealLiu et al. (2018), IR-NetQin et al. (2020), and Bi-NealNie et al. (2022). Bi-Neal demonstrates the generality of this approach across various tasks and successful hardware implementation. Another method involves modifying the network's structure, as demonstrated by Group-NetZhuang et al. (2019), MeliusNetBethge et al. (2021), and ReActNetLiu et al. (2020). Notably, MeliusNet and ReActNet even outperform full-precision networks.

An extension of BNN worth mentioning is the Binary Complex Neural Network (BCNN)Li et al. (2021). BCNN incorporates complex representations and operations to enhance network capabilities. Complex networks have shown improved accuracy compared to real-number networks of the same sizeTrabelsi et al. (2017), primarily due to the utilization of phase information. To exploit the advantages of complex numbers and demonstrate the universality of our proposed method, we conducted experiments on binarizing IR neural networks using binary complex numbers.

Recent research has focused on developing specialized training strategies for BNNs to improve training and enhance performance. The conventional method, introduced in Cour-

bariaux et al. (2016), approximates the gradient of the Sign function while keeping real-valued weights during training using $\text{Clip}(-1, x, 1)$. Other studies have built upon this method by modifying the gradient approximation for the Sign functionLiu et al. (2018); Gong et al. (2019); Qin et al. (2020); Ma et al. (2019); Huang et al. (2021). These methods utilize binary weights and activations during forward propagation, except for Ma et al. (2019), which focuses on binarizing weights only. However, our experiments have revealed challenges when applying these methods to binarize IR neural networks, particularly for BCNNs. Hence, we propose a novel training strategy, named the Gentle Approximation Method (GAM), which will be detailed in Section 3.2. Another approach to address training difficulties involves introducing redundant parametersLiu et al. (2018); Ma et al. (2019); Xin et al. (2020), although most of these parameters can be absorbed by normalization layers. Bi-NealNie et al. (2022) provides a comprehensive overview in this area. Therefore, normalization plays a crucial role in our proposed method.

Many studies have focused on binarizing classification networks, while other computer vision applications such as segmentationZhuang et al. (2019), detectionSun et al. (2018); Wang et al. (2020), and matchingNie et al. (2022) have also been explored. However, the binarization of low-level computer vision tasks, particularly super-resolution (SR), has received limited attention. Ma et al. Ma et al. (2019) initially attempted to binarize weights only. BAMXin et al. (2020) introduced additional parameters and a new loss function, along with a specialized model for binary super-resolution networks. IBTMJiang et al. (2021) eliminated batch normalization and replaced ReLU with PReLU. Bi-NealNie et al. (2022) achieved state-of-the-art performance with a $1.5\times$ increase in the number of channels. BBRUXia B (2022) extensively transformed IR networks into BNNs but focused on networks with three functionalities: super-resolution, JPEG compression, and image denoising.

The exploration of binarizing IR neural networks remains limited. This work presents a novel and efficient binarization technique specifically designed for IR neural networks. The proposed technique not only extends to super-resolution but also encompasses low-light enhancement, denoising, and deraining tasks.

3. Methodology

There are two significant challenges associated with binarizing IR neural networks. Firstly, binarization inevitably reduces the representation capacity of hidden layers, which conflicts with the abundant information present in the outputs of IR tasks. Secondly, training 1-bit CNN models steadily poses a difficult problem, and this holds true for binarizing IR tasks as well.

To address these challenges, this study proposes a straightforward binarization technique primarily utilizing convolutional operations (Sec. 3.1). Additionally, a universal training strategy, known as the Gentle Approximation Method (GAM, Sec. 3.2), is introduced to facilitate the smooth optimization of the modified networks.

3.1. Binarization Methods

To portray the general architecture of a binarized IR neural network, this study introduces two methods of binarization along with two compensations for information loss. Typically,

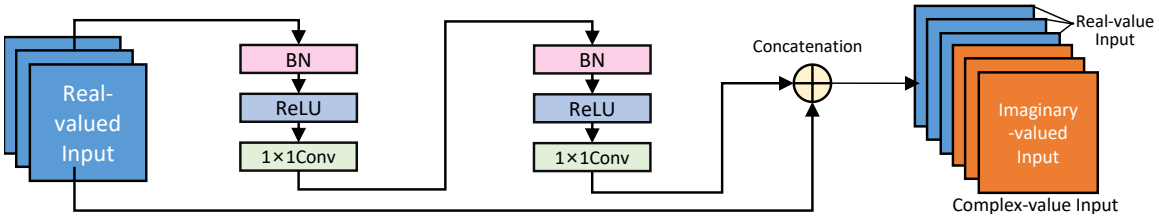


Figure 2: Complex input generation blockTrabelsi et al. (2017).

pooling layers and fully-connected layers are not present in IR neural networks; therefore, designing binarization methods for these layers was not considered.

Basic Binarization. The objective of network binarization is to represent both weights and activations using 1-bit values. In this study, we adopt the common practice of binarizing convolution layers, which can be expressed as follows:

$$x_b = \text{Sign}(x_f) = \begin{cases} -1, & \text{if } x_f < 0 \\ +1, & \text{otherwise} \end{cases} \tag{1}$$

where x_f denotes the float point activation or weight and x_b indicates the corresponding binary value. This approach significantly reduces the computational complexity of the convolution operation, particularly for hardware implementations, as the `xnor` and `popcount` operation replace multiplication and addition, as demonstrated in prior worksRastegari et al. (2016).

Complex Binarization. To convert a full-precision network into a binary complex network, we adopt the methods proposed in BCNNLi et al. (2021). An additional block is inserted before the first layer to generate the imaginary part of the input (Fig. 2). When binarizing these complex numbers, we employ quadrant binarization, as illustrated below:

$$x_b = \text{Sign}(x_r + ix_i) = \text{Sign}(x_r) + i\text{Sign}(x_i) = (x_r)_b + i(x_i)_b \tag{2}$$

where x_b is the binarized input complex activation or weight, while x_r and x_i correspond to the real and imaginary parts, respectively. This approach decouples the real and imaginary parts, enabling them to be treated separately, similar to ordinary binary numbers.

Thus far, we have employed the vanilla binarization method, as many variations of BNNs can be effectively summarized by normalization layers, which will be elaborated in subsequent sections.

Nomalization. In this method, a normalization layer is incorporated immediately after each binary convolutional layer. Normalization techniques, such as Batch Normalization (BN)Ioffe and Szegedy (2015), are commonly employed in neural etworks. The BN operation can be defined as follows:

$$\text{BN}(x) = \gamma_j \frac{x - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} + \beta_j \tag{3}$$

where x indicates the input data. μ_j, σ_j indicate average value and standard deviation in j-th channel respectively while γ_j and β_j are trainable parameters. These four

parameters$(\mu_j, \sigma_j, \gamma_j, \beta_j)$ are obtained during training and remain constant in inference. Generally, normalization serves as an implicit regularization method and aids in promoting effective generalization of the network.

Converging 1-bit CNNs without Batch Normalization (BN)Courbariaux et al. (2016) is challenging. Some studies have introduced redundant parameters to facilitate the training process and enhance network performance, but these parameters can be absorbed by BNNie et al. (2022). Consequently, models incorporating such parameters are mathematically equivalent to models without them, but with modified BN parameters. In this work, we posit that alternative training strategies can yield similar effects as redundant parameters, which will be elaborated in Section 3.2.

Furthermore, normalization plays a crucial role in binarizing IR neural networks as it effectively shifts the mean activation to zero channel-wise. This shift is essential for improving information entropyQin et al. (2020), wherein higher entropy indicates a greater amount of retained information after binarization. ReActNet also highlights the significance of activation shifting in 1-bit CNNs, and they achieve this through a combination of basic activation functions and BN (see Eq. 4). While Qin et al. (2020) normalize weights to achieve this objective, the use of traditional normalization layers is a more straightforward implementation approach.

Moreover, FINNUmuroglu et al. (2017) demonstrates that by combining BN with the Sign function, the MAdd operation can be replaced with a simple comparison, as illustrated below:

$$\text{Sign}(\text{BN}(x)) = \text{Sign}(\gamma\frac{x-\mu}{\sqrt{\sigma^2+\epsilon}}+\beta) \quad = \begin{cases} -1, & \text{if } \tau < 0 \\ +1, & \text{otherwise} \end{cases} \tag{4}$$

where $\tau = \mu - \beta\sqrt{\sigma^2+\epsilon}/\gamma$, represents the threshold used in `comparison`. This finding indicates that BN is hardware-friendly for 1-bit CNNs. However, in certain IR tasks, alternative normalization techniques may be more suitable, such as Instance Normalization (IN)Ulyanov et al. (2016). While IN cannot be simplified during inference, the computational overhead it incurs is acceptable (see Sec. 4.2).

Applying normalization to complex-valued activations can be challengingTrabelsi et al. (2017). Fortunately, BCNNLi et al. (2021) has demonstrated that Complex Gaussian Batch Normalization(CGBN) is more effective in stabilizing the training process and reducing computation. Therefore, we adopt a similar method to normalize complex activations, treating the real and imaginary parts of the complex numbers separately, as illustrated below:

$$\widehat{x}_c = \gamma_c\left(\frac{x_r-\mu_r}{\sqrt{2\sigma_r^2+\epsilon}} + i\frac{x_i-\mu_i}{\sqrt{2\sigma_i^2+\epsilon}}\right)+\beta_c \tag{5}$$

where the subscript c, r, i indicate complex value, real value and imaginary value respectively.

Close Loop. The architectures of IR neural networks exhibit rich diversity, but a significant majority of them feature a closed-loop structure. In our method, we introduce a closed loop to those networks that lack it. The closed loop plays a crucial role in binarizing IR neural networks by mitigating information degradation. Let a_n represent a pixel in the feature map of the n-th layer with c_n channels. The value of a_n is obtained through the

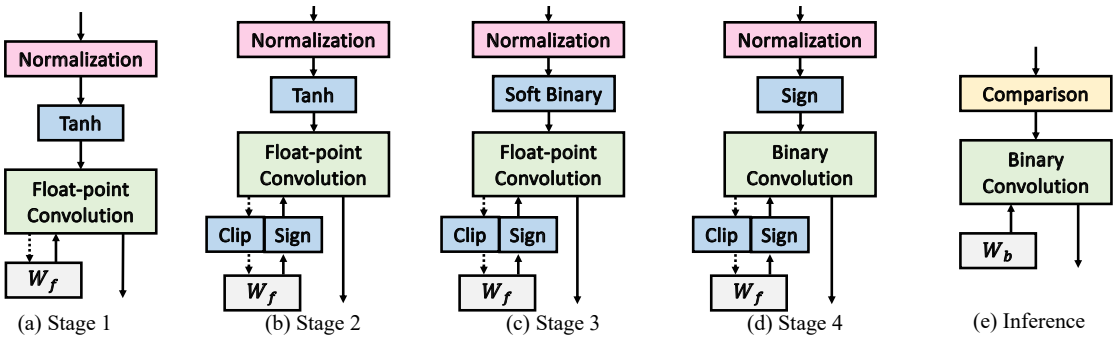


Figure 3: A graphical illustration of the network in different stages using GAM. Clip is short for $\text{Clip}(-1, x, 1)$. W_f indicates the float-point weights and W_b indicates the bianry weights. The solid lines indicate the flow of forward propagation and the dotted lines indicate the gradient flow to weight of convolution during backward propagation. The close loop is omitted for simplicity.

following process:

$$a_n = \text{BN}(\text{Conv}(a'_{n-1}; w'_{n-1})) \tag{6}$$

where a'_{n-1} and w'_{n-1} represent binarized activations and weights in $(n-1)$-th layer respectively. BN can be regarded as an affine transformation that does not alter the number of possible values for a_n. Thus, the number of possible values is determined by the convolution operation, as expressed by:

$$Num(a_n^j) = c_{n-1}k^2 + 1 \tag{7}$$

where $Num()$ represents the number of possible values for a_n^j, which corresponds to the j-th channel, and k denotes the kernel size. It is important to note that this representation capacity is significantly lower than that of floating-point numbers. However, as demonstrated in Liu et al. (2018), a simple shortcut can enhance the representational capability.

Hence, in the case of certain IR networks lacking a closed loop structure, additional architectural modifications are required. In Section 4.4, we will demonstrate the effectiveness of incorporating a closed loop when binarizing a plain CNN for IR tasks.

3.2. Gentle Approximation Method

The training of 1-bit CNNs presents two challenges: the non-differentiability of the Sign function and the requirement for larger gradients to update the weights' signs. These challenges are traditionally addressed using the method proposed in Courbariaux et al. (2016), which involves approximating the gradient of the Sign function using the Straight Through Estimator (STE) and preserving the real values of weights during training, as illustrated in Eq. 8.

$$\frac{\partial l}{\partial x_r} = \frac{\partial l}{\partial x_b} 1_{|x_r| \leq 1} \tag{8}$$

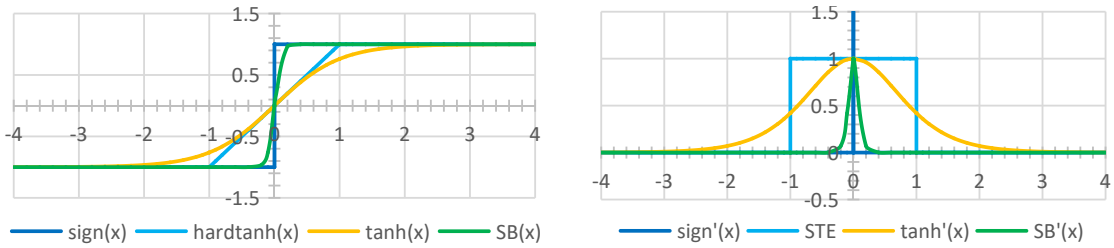


Figure 4: (a) different activation functions; (b) derivatives of functions in (a) respectively.

However, the application of traditional training methods to train binarized IR neural networks often results in poor performance or even non-convergence, particularly for BC-NNs. In addition, considering the finer information contained in the activations and outputs of IR tasks, we propose a new strategy called the Gentle Approximation Method (GAM) to train binarized IR neural networks. Our approach aims to closely approximate the distribution of the full precision activation to that of the 1-bit CNN. The entire training process is divided into four stages, where each stage utilizes the model obtained from the previous stage as a pre-trained model. Within each stage, once the trained model is able to generate the desired results, it can be employed as the pretrained model for the subsequent stage. This method can be easily extended to train any 1-bit CNN without introducing additional loss functions or parameters.

Stage 1: Training Full-precision Network. In Bi-RealLiu et al. (2018), the network is initialized with activation functions replaced by Hardtanh(x). As the architecture of the binarized IR network undergoes modifications, a similar initialization approach is employed for training the network, but Tanh(x) is used instead, as illustrated in Fig. 3(a). It can be observed that compared to Hardtanh(x), Tanh(x) exhibits a gentler slope and a wider range of non-zero derivatives(Fig. 4). This characteristic facilitates a looser condition and a more accurate direction for network training, thereby laying the foundation for stage 2.

Stage 2: Binarizing Weight. The only difference between stage 1 and stage 2 lies in the binarization of weights (Fig. 3(b)). To compute derivatives, Clip($-1, x, 1$) is utilized. The weights are binarized first because they are less sensitive to quantizationZhou et al. (2016); Gong et al. (2019). During inference, multiplication becomes dispensable with binary weights, allowing the model obtained in this stage to be used for inference in certain cases.

Stage 3: Gently Binarizing Activation. Since activations carry more information and are highly sensitive to binarizationGong et al. (2019), we have observed that directly binarizing activations after stage 2 results in a decrease in network performance or even non-convergence, based on our empirical study. Therefore, we propose a new activation function called Soft Binary (SB), defined as follows:

$$\mathrm{SB}(x) = \mathrm{Tanh}(10x) \tag{9}$$

This function closely resembles Sign (Fig. 4(a)), while exhibiting a significantly wider range of non-zero derivatives(Fig. 4(b)), approximately ranging from $[-1.6, 1.6]$. This ex-

Table 1: The comparison between float-point model, BNN and BCNN for Zero-DCE++.

	PSNR/SSIM	#params	model size	FLOPs	BOPs
Float-Point	15.68/0.512	10,561	41.25KB	1.23G	-
BNN	14.28/0.499	10,625	6.75KB	111.38M	1.12G
BCNN	14.31/0.501	7,037	6.30KB	150.50M	1.36G

tended range allows for finer adjustments during the training process. The model obtained in this stage can even be used for inference in 1-bit CNN with minimal performance decay, when compared to the model obtained in stage 4.

Stage 4: Training Binarized Network. This stage serves as a fine-tuning step for the network. The training method follows the approach proposed in Courbariaux et al. (2016), where both the weights and activations are binarized.

4. Experiments

To demonstrate the versatility of our method, this section employs four different CNNs for various IR tasks. Among them, two networks have not been binarized previously. Both plain binarization and complex binarization techniques are applied to these networks, utilizing the GAM for training. Following convention, the first and last convolution layers or blocks remain in floating-point format. For BCNN, a complex input generation block is added as described in Section 3.1. To ensure a fair comparison, during complex binarization, half of the channels in the floating-point network are maintained as real-valued, while the remaining channels are represented as imaginary binary numbers. Unlike some other binarization strategiesLi et al. (2021); Nie et al. (2022), we keep all hyperparameters, weight initialization, and optimization algorithms consistent with the original network. The weight initialization in complex binarization follows the approach proposed in Li et al. (2021), and the number of training epochs may vary, allowing ample room for further improvement.

In terms of efficiency of our methods, both performance and resource saved will be considered. For performance evaluation, we select at least one objective benchmark and compare the results obtained from full-precision networks, BNNs, and BCNNs, using the same training and test sets. Regarding resource utilization, we calculate memory usage and computational complexity. Since hardware implementations can vary in real-world scenarios, we separately calculate the floating-point operations (FLOPs) and binary operations (BOPs).All the networks are implemented by Pytorch and trained on a NVIDIA 2080Ti GPU.

4.1. Low-Light Image Enhancement

We first binarize the Zero-DCE++ networkGuo et al. (2020), which is specifically designed for low-light enhancement. Unlike other IR networks, Zero-DCE++ produces a parameter map as its output. This lightweight network comprises 7 depthwise separable convolutions with a U-Net-like structure. We utilize the training set provided by the original official code for training. During our experimentation, we observed that applying the traditional

(a) low-light input (b) float-point (c) BNN (d) BCNN (e) ground truth

Figure 5: A comparison between original Zero-DCE++ and binarized ones.

training strategy for BNNs may result in non-convergence when training BCNNs for Zero-DCE++. However, when trained with GAM, the BNN version requires 25 epochs for each stage, while the BCNN version requires 70 epochs for the first stage and 10 epochs for each subsequent stage.

Table 1 presents a comparison of performance and efficiency between the floating-point network and 1-bit CNNs. FLOPs and BOPs are computed based on a 256×256 input image. BOPs encompass the operations of `xnor`, `popcnt`, and `comparison`. The evaluation is conducted on the Part2 Subset of the SICE datasetCai et al. (2018), ensuring a fair comparison with the original work. Although the original network's performance differs slightly from the paper, it is evident that the gap between the floating-point and binary versions is small, and the details of the input images are preserved(see Fig.5). Furthermore, the floating-point calculation is only approximately one-tenth of the original network, resulting in a significant reduction in memory usage, making it suitable for edge devices. BCNN outperforms BNN with slightly increased computation but with better memory efficiency, indicating its ability to make more effective use of limited information.

4.2. Deraining

We select PeReNetRen et al. (2019) for binarizing the deraining network. PeReNet consists of 6 different structures, and for simplicity and lightweight implementation, we choose the smallest network, PRN_r. If higher performance is desired, larger networks can be selected accordingly. PRN_r is composed of 6 repeated blocks, with the first and last blocks kept in full precision and the middle blocks binarized. Both the first and last blocks have identical structures, as do the remaining blocks. It should be noted that the addition of the new binary block increases the model memory overhead compared to the original network. The number of blocks in the intermediate section can be adjusted flexibly based on specific requirements.

Table 2: The comparison between float-point model, BNN and BCNN for PRN_r.

	PSNR/SSIM (Rain100H)	PSNR/SSIM (Rain100L)	#params	model size	FLOPs	BOPs
Float-Point	27.43/0.874	36.11/0.973	21,123	82.51KB	69.46G	-
BNN	25.00/0.852	35.63/0.971	42,147	83.15KB	23.54G	46.27G
BCNN	25.03/0.857	35.91/0.972	44,007	127.22KB	23.76G	46.49G

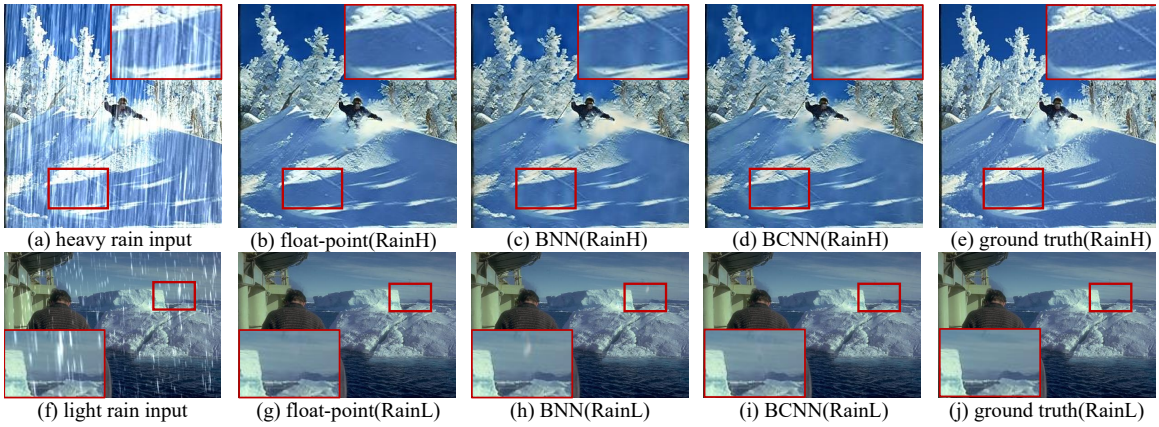


Figure 6: A comparison between original PRN_r and binarized ones in both Rain100H and Rain100L dataset.

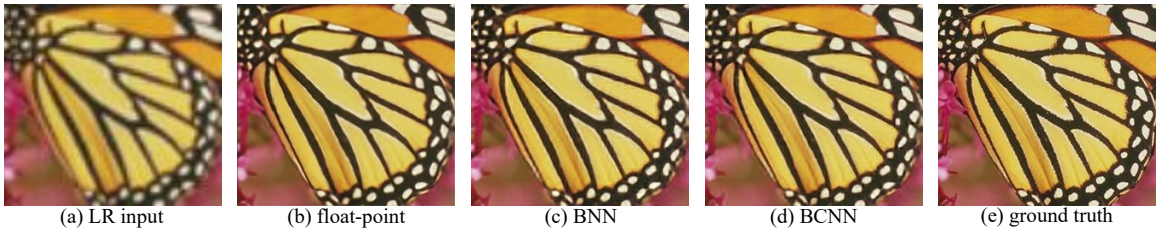


Figure 7: A comparison between original SRResNet and binarized ones.

The comparison is made in Table 2. Two representative datasets are chosen to test binarized networks. Rain100H is harder to be derained than Rain100LYang et al. (2017). From the table we can tell while maintaining most of the deraining function, 1-bit CNNs reduce the FLOPs to about one third of the original. Also, the performance of 1-bit CNN is closer to float-point in easier task. We empirically found using IN leads to better performance than using BN, whose PSNR is 22.05dB for BNN on Rain100H dataset. BCNN still remain superiority in this task. From fig. 6, we can tell that BCNN is able to handle the heavy rain better in both heavy and light rain condition. Its performance is very close to original network for Rain100L dataset.

4.3. Super-resolution

To compare with previous works, we choose SRResNetLedig et al. (2017), which is also binarized in Ma et al. (2019), Xin et al. (2020), Huang et al. (2021). We keep the first convolution layer and the upscale block float-point as is done in previous works. The super-resolution scale is $\times 4$.

Because the upscale block takes nearly half of the calculation complexity, the reduction is not as much as other networks, but still 1-bit CNN only need less than a half computation

Table 3: The comparison between different binarized SRResNet.

	PSNR(Set5)	PSNR(Set14)	#params	model size	FLOPs	BOPs
Float-Point	31.80	28.25	1,546,752	5.90MB	273.66G	-
BNN	30.53	27.59	1,546,752	1.53MB	129.47G	144.00G
BCNN	30.64	27.60	956,958	1.46MB	131.37G	144.06G

Table 4: The comparison between float-point model, BNN and BCNN for SRResNet.

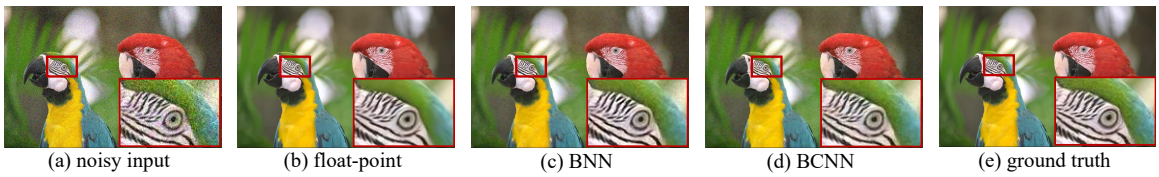
	FP	BNN	Dorefa	BNN (this work)	BCNN (this work)	Ma et al.	BNN (BWN)	BCNN (BWN)
Set5	31.80	29.33	30.38	30.53	**30.64**	30.34	30.64	**30.66**
Set14	28.25	26.72	27.48	27.59	**27.60**	27.16	**27.63**	27.62

resource of the original work, and about a quarter of model size, as is shown in Table 3. We also compare our work with previous work in Table 4. Our plain BNN version has exceeded the BNN version trained by traditional methods, implying the superiority of GAM. Since binary weight network (BWN) is a byproduct of our training strategy, we compare these models with Ma et al.Ma et al. (2019), too. Fig. 7 shows an example from Set5.

4.4. Denoising

For denoising, we choose a typical CNN, FFDNetZhang et al. (2018), to be binarized. This is a simple network consisted of vanilla convolutional layer, BN and ReLU, making it the only network added a ResNetHe et al. (2016) connection in this work, in order to form the close loop. ResNet is a simple yet effective structure for IR tasks, for it is easier to optimize residual blocks when its input and output are alike. Limited by the experimental resources, a subset of the original training set, BSD500Martin et al. (2001), is used, as the change in training set don't do too much harm to the performance of original network based on our experiments. The test set is Kodak24Franzen (1999).

Table 5 and table 6 are the performance and resources cost by full-precision network and 1-bit CNNs. Although gaps exist between the performance of float-point and binary FFDNet, the cost reduction is huge. The FLOPs and model size is about 3% and 5% of



(a) noisy input (b) float-point (c) BNN (d) BCNN (e) ground truth

Figure 8: A comparison between original SRResNet and binarized ones. σ=25.

Table 5: The performance comparison between float-point model, BNN and BCNN for FFDNet. σ indicates the level of noise.

		Kodak24				
	model	σ=15	σ=25	σ=35	σ=50	σ=75
PSNR	Float-Point	34.72	32.28	30.75	29.18	27.48
	BNN	31.94	29.93	28.66	27.13	24.76
	BCNN	32.41	30.23	30.23	27.17	25.06

Table 6: The cost comparison between float-point model, BNN and BCNN for FFDNet.

	#params	model size	FLOPs	BOPs
Float-Point	856,608	3.27MB	104.27G	-
BNN	856,608	192.82KB	2.84G	101.25G
BCNN	442,458	144.49KB	4.46G	101.28G

the original network, respectively, pushing the compression to the limit and leaving a big margin for further improvement. Fig. 8 is an example of outputs from different networks.

5. Conclusion

In this paper, we have proposed a universal method for transforming IR neural networks into BNNs or BCNNs. We have also introduced an efficient and easy-to-implement training strategy, called GAM, which enables the float point network to mimic the data distribution in BNNs, without introducing any new parameters or loss functions. Our experiments have been conducted on various IR tasks, including low-light image enhancement, deraining, super-resolution, and denoising. The results demonstrate the effectiveness and universality of our method.

As a work that applies 1-bit CNN to a wide range of IR tasks, our primary focus has been on demonstrating the feasibility of our method. Therefore, there is ample room for further improving the performance of these networks. We also advocate for the development of binary networks specifically designed for IR tasks, which can be dedicated to the training process while maintaining a simple model and inference process.

References

Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, and Christoph Meinel. Meliusnet: An improved network architecture for binary neural networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1439–1448, 2021.

Jianrui Cai, Shuhang Gu, and Lei Zhang. Learning a deep single image contrast enhancer from multi-exposure images. *IEEE Transactions on Image Processing*, 27(4):2049–2062, 2018.

Liangyu Chen, Xin Lu, Jie Zhang, Xiaojie Chu, and Chengpeng Chen. Hinet: Half instance normalization network for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 182–192, 2021.

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.

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. doi: 10.1109/JPROC.2020.2976475.

R. Franzen. Kodak lossless true color image suite. http://r0k.us/graphics/kodak, 1999.

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/CVF International Conference on Computer Vision*, pages 4852–4861, 2019.

Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. Zero-reference deep curve estimation for low-light image enhancement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1780–1789, 2020.

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.

Qiu Huang, Yuxin Zhang, Haoji Hu, Yongdong Zhu, and Zhifeng Zhao. Binarizing super-resolution networks by pixel-correlation knowledge distillation. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 1814–1818. IEEE, 2021.

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.

Xinrui Jiang, Nannan Wang, Jingwei Xin, Keyu Li, Xi Yang, and Xinbo Gao. Training binary neural network without batch normalization for image super-resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1700–1707, 2021.

Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8878–8887, 2019.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

Yanfei Li, Tong Geng, Ang Li, and Huimin Yu. Bcnn: Binary complex neural network. *Microprocessors and Microsystems*, 87:104359, 2021.

Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. *Advances in neural information processing systems*, 30, 2017.

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.

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.

Yinglan Ma, Hongyu Xiong, Zhe Hu, and Lizhuang Ma. Efficient super resolution using binarized neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 2437–2445, 2020.

Guhong Nie, Lirui Xiao, Menglong Zhu, Dongliang Chu, Yue Shen, Peng Li, Kang Yang, Li Du, and Bo Chen. Binary neural networks as a general-propose compute paradigm for on-device computer vision. *arXiv preprint arXiv:2202.03716*, 2022.

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.

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.

Dongwei Ren, Wangmeng Zuo, Qinghua Hu, Pengfei Zhu, and Deyu Meng. Progressive image deraining networks: A better and simpler baseline. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3937–3946, 2019.

Yuda Song, Yang Zhou, Hui Qian, and Xin Du. Rethinking performance gains in image dehazing networks. *arXiv preprint arXiv:2209.11448*, 2022.

Jingwen Su, Boyan Xu, and Hujun Yin. A survey of deep learning approaches to image restoration. *Neurocomputing*, 487:46–65, 2022.

Siyang Sun, Yingjie Yin, Xingang Wang, De Xu, Wenqi Wu, and Qingyi Gu. Fast object detection based on binary deep convolution neural networks. *CAAI transactions on intelligence technology*, 3(4):191–197, 2018.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, Joao Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2017.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Yaman Umuroglu, Nicholas J Fraser, Giulio Gambardella, Michaela Blott, Philip Leong, Magnus Jahre, and Kees Vissers. Finn: A framework for fast, scalable binarized neural network inference. In *Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays*, pages 65–74, 2017.

Ziwei Wang, Ziyi Wu, Jiwen Lu, and Jie Zhou. Bidet: An efficient binarized object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2049–2058, 2020.

Wang Y Xia B, Zhang Y. Basic binary convolution unit for binarized image restoration network. *arXiv preprint arXiv:2210.00405*, 2022.

Jingwei Xin, Nannan Wang, Xinrui Jiang, Jie Li, Heng Huang, and Xinbo Gao. Binarized neural network for single image super resolution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 91–107. Springer, 2020.

Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1357–1366, 2017.

Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

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.

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/CVF Conference on Computer Vision and Pattern Recognition*, pages 413–422, 2019.