Expanding-and-Shrinking Binary Neural Networks

Xulong Shi[1]   Caiyi Sun[2]   Zhi Qi[2]   Liu Hao[2]   Xiaodong Yang[1]

[1]QCraft   [2]Southeast University

Abstract

While binary neural networks (BNNs) offer significant benefits in terms of speed, memory and energy, they encounter substantial accuracy degradation in challenging tasks compared to their real-valued counterparts. Due to the binarization of weights and activations, the possible values of each entry in the feature maps generated by BNNs are strongly constrained. To tackle this limitation, we propose the expanding-and-shrinking operation, which enhances binary feature maps with negligible increase of computation complexity, thereby strengthening the representation capacity. Extensive experiments conducted on multiple benchmarks reveal that our approach generalizes well across diverse applications ranging from image classification, object detection to generative diffusion model, while also achieving remarkable improvement over various leading binarization algorithms based on different architectures including both CNNs and Transformers.

1. Introduction

Deep neural networks (DNNs) have achieved tremendous impact in a broad range of domains [1, 6, 19, 34, 49, 50]. However, the continuously increasing demand of computation and memory poses significant challenges for deployment on mobile or embedded platforms that support various real-time applications, e.g., robots, augmented reality, and autonomous driving [20, 25, 39].

In order to address this issue, numerous methods have been proposed, including quantization [7, 9, 53], pruning [21, 26, 28], knowledge distillation [11, 45, 51], and compact network design [13, 24, 52]. At the extreme end of quantization, binary neural networks (BNNs) emerge as one of the most promising techniques to fulfill the onboard deployment with constrained computation and memory resources. As demonstrated in [36], BNNs can yield $32\times$ memory compression and up to $58\times$ computation reduction on CPU, and they can be accelerated further on FPGA. In addition, BNNs perform only bit-wise operations using the arithmetic-logic unit, which is also more energy-efficient than the floating-point unit.
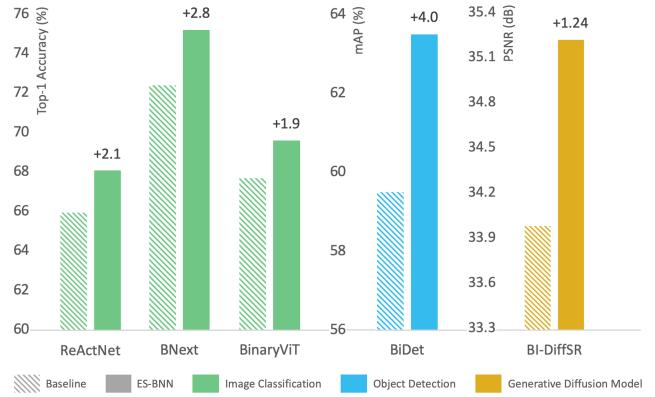

Figure 1. Overview of performance improvement for image classification (top-1 accuracy) on ImageNet, object detection (mAP) on PASCAL VOC, and generative diffusion model based image super-resolution (PSNR) on Manga. Enabled by the proposed ES-BNN, various binary neural networks based on both CNNs and Transformers obtain consistent and significant performance boost.

In spite of the aforementioned merits in memory, speed and energy, BNNs suffer from large accuracy degradation in challenging tasks. To mitigate the gap between BNNs and their real-valued counterparts, prior research has made substantial efforts in optimizing the forward inference and backward propagation algorithms. However, due to the binarization nature of weights and activations (i.e., only two possible values: -1 and 1), each entry of output feature maps produced in BNNs is strongly restricted in a limited number of possible values, and the number is determined by the specific architectural configuration of each layer. We define this number as the **representation capacity** of the corresponding layer, and show that such limited representation capacity is a fundamental issue leading to the accuracy degradation. Regardless of how binarization algorithms are improved, they are ultimately confined by the limited representation capacity.

In light of the above observations, we propose a simple yet effective expanding-and-shrinking operation, which expands input channels (i.e., output channels of a preceding layer) and shrinks convolution kernels. This operation leads to an enhanced representation capacity of output features in

each individual layer, while maintaining its original computation complexity and parameter number. Additionally, we combine the expanding-and-shrinking operation with binary group convolution, and their integration results in a notably advantageous synergy: the former inherently enables cross-group information flow, while the latter offers a more favorable balance between expanding and shrinking. We refer to a binary neural network enhanced by the proposed expanding-and-shrinking operation as **ES-BNN**. Thanks to the proposed generalizable operation, our approach is flexible to fit in diverse binarization algorithms, network architectures, as well as downstream tasks. As demonstrated in Figure 1, ES-BNN consistently and remarkably improves multiple representative BNNs.

Our main contributions are summarized as follows. First, we propose the expanding-and-shrinking operation, which boosts the representation capacity of each binary layer and improves the overall accuracy, at the cost of negligible increase in the computation complexity of an original BNN. Second, our approach is applicable to various leading binarization algorithms as well as network architectures from CNNs to Transformers. Third, our approach generalizes in diverse downstream applications ranging from image classification, object detection to generative diffusion model, while rendering superior results on multiple benchmarks. Our code and models are released at https://github.com/imfinethanks/ESBNN.

2. Related Work

Since the introduction of binary neural networks, a number of research works are dedicated to minimize the information loss incurred by the binarization of weights and activations. In this field, remarkable progress has been made in optimizing the training and binarization algorithms.

Binarization of a neural network originates from the pioneering works in [15, 40], which establish the end-to-end trainable framework of binary weights and activations through expectation backpropagation and straight-through estimator, respectively. A series of following methods are proposed to improve the training algorithms. IR-Net [35] introduces the error decay estimator to mitigate information loss of gradients by gradually approximating the sign function in backward propagation, which guarantees adequate updates at the beginning and accurate gradients at the end of training. RBNN [22] develops a training-aware approximation of the sign function to enable gradient propagation. ReCU [48] exploits the rectified clamp unit to revive the dead weights that are barely updated during training. AdamBNN [30] analyzes the impact of weight decay and Adam for training, based on which a simple training scheme is derived. BNext [10] proposes a diversified consecutive knowledge-distillation technique to alleviate the counter-intuitive overfitting problem.

In parallel, a large family of the binarization research attempts to compensate for the information loss in binarized features. XNOR-Net [36] first uses two types of real-valued scaling factors for both weights and activations in binarization so as to minimize the quantization error. Bi-Real [27] adds real-valued shortcuts to propagate real-valued features in order to complement the binarized main branch. Real-to-Bin [32] reduces the output discrepancy between the binary and the corresponding real-valued convolution based on the proposed real-to-binary attention matching and data-driven channel re-scaling. Group-Net [55] and BENN [54] both leverage the ensemble methods to combine multiple binary models to obtain performance gains. ReActNet [29] introduces RSign and RPRelu to explicitly shift and re-shape activation distribution, and uses a distributional loss to further align features between binary and real-valued networks. AdaBin [42] adaptively obtains an optimal binary set of weights and activations for each layer instead of a fixed set. INSTA-BNN [18] dynamically controls the quantization threshold in an instance-aware manner based on the representative higher-order statistics to estimate the characteristics of input distribution.

3. Method

In this section, we start from providing an overview, which includes the fundamental principles of binarization and the underlying motivation of our approach. We then present the proposed expanding-and-shrinking operation as well as the synergistic binary group convolution.

3.1. Problem Statement

In the following, we use CNNs to illustrate the formulation of our approach, which is also straightforward to be applied to Transformers. We denote by $W \in \mathbb{R}^{c_o \times c_i \times k \times k}$ and $A \in \mathbb{R}^{c_i \times w \times h}$ the weights and input features of a convolutional layer, where c_o and c_i represent the number of output and input channels, w and h indicate the width and height of the input features, and k is the spatial dimension of the kernel. In a BNN, both weights and features are binarized, and the convolution is performed as:

$$A * W \approx \mathcal{B}_A(A) \circledast \mathcal{B}_W(W), \qquad (1)$$

where $\mathcal{B}_A$ and $\mathcal{B}_W$ represent the binarization functions of weights and features, and $\circledast$ is the binary convolution that can be implemented by efficient bit-wise operations. While the specific algorithms of forward inference and backward propogation of $\mathcal{B}_A$ and $\mathcal{B}_W$ can be different in various methods, $\mathcal{B}_A(A)$ and $\mathcal{B}_W(W)$ have only two possible values (i.e., -1 and $+1$) represented by a single bit.

Thus, unlike the real-valued convolution, the output of binary convolution is constrained in a limited set of values,
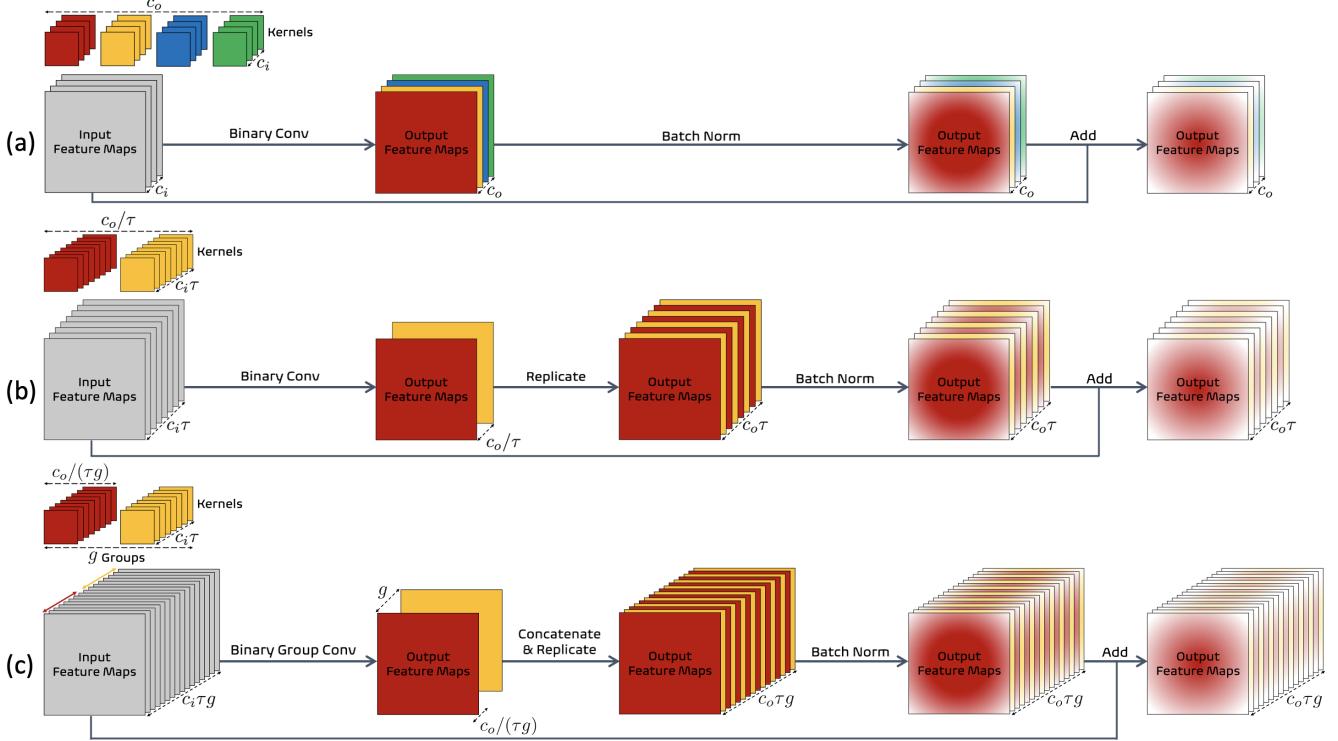
Figure 2. A schematic overview of the proposed approach ES-BNN utilized in each individual layer of a binary neural network. (a) shows a standard binary convolution layer. (b) illustrates the expanding-and-shrinking operation applied on top of (a). In (c) the binary group convolution is further integrated based on (b).

and the entry at $[i, x, y]$ of an output feature map is:

$$\sum_j \sum_{dx} \sum_{dy} \mathcal{B}_A(A[j, x+dx, y+dy])\mathcal{B}_W(W[i, j, x+dx, y+dy]),$$

where $j = 0, ..., c_i - 1$, and $dx, dy = -(k-1)/2, ..., (k-1)/2$. Therefore, the value of each entry of the output feature map is in the range of $[-c_i k^2, c_i k^2]$ with an interval of 2 (i.e., there are $c_i k^2 + 1$ possible values in total). We define this total number of possible values as the **representation capacity** of the corresponding layer in a BNN.

As aforementioned, research works in this field mostly focus on improving the forward inference and backward propogation algorithms to be better tailored for binarization. However, as long as A and W in Equation 1 undergo binarization, their output feature maps are restricted by the representation capacity, which strongly delimits the overall accuracy of a BNN.

In this paper, we seek to answer the research question: *How can we push the boundary of representation capacity, and meanwhile, maintain the original computation complexity of a BNN?*

Directly increasing c_i or k indeed escalates the representation capacity, but incurs dramatic growth in the computation complexity as well. In order to overcome this dilemma, we propose the expanding-and-shrinking operation, which

strengthens the representation capacity, and consequently, improves the final accuracy of various BNNs, with negligible extra computation overhead.

3.2. Expanding and Shrinking

Figure 2(a-b) illustrates the comparison between a standard binary convolution and our proposed approach. We first expand the input channels from c_i to $c_i\tau$, where $\tau \in \mathbb{Z}^+$ is a scaling factor to control the increase of representation capacity. We then shrink the convolution kernels from c_o to c_o/τ with the purpose of maintaining a constant computation complexity. This is equivalent to reshape A to $\widetilde{A} \in \mathbb{R}^{c_i\tau \times w \times h}$ and W to $\widetilde{W} \in \mathbb{R}^{c_o/\tau \times c_i\tau \times k \times k}$. However, such shrinked output feature maps undermine the representation capacity of next layer (i.e., the number of input channels for next layer is also reduced), and can be incompatible with the architectural unit (e.g., the numbers of feature channels are required to match in a residual connection). To compensate for the shrinking, we expand the output channels by making τ^2 copies of $\mathcal{B}_A(\widetilde{A}) \circledast \mathcal{B}_W(\widetilde{W})$ and concatenating them along the channel dimension, such that the final number of output channels of current layer become $c_o\tau$, which essentially makes the input channels of next layer expanded. In this manner, we can continu-
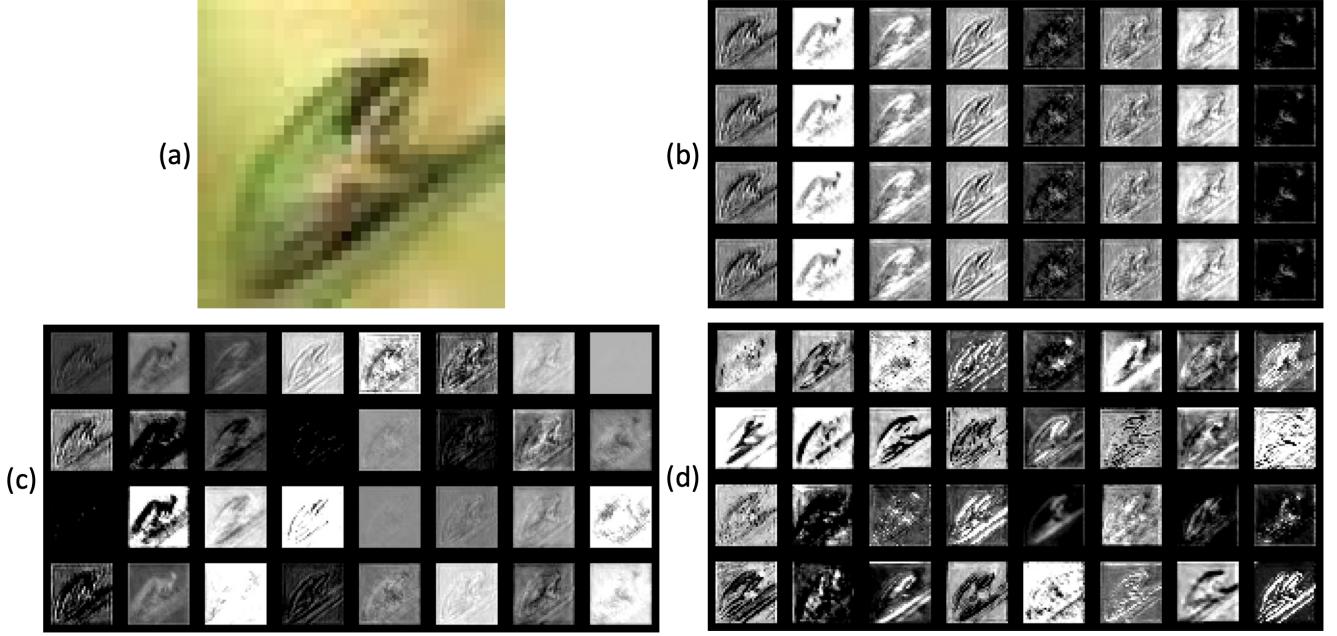
Figure 3. Illustration of the diversification process of the replicated feature maps in the proposed approach. (a) shows the input image. (b) presents the replication of 8 feature maps (in one row) from the 7th layer in ResNet-20. (c) demonstrates the feature maps in (b) after the batch normalization. (d) corresponds to the feature maps in (c) after the residual connection.

ously enhance the representation capacity across all layers through the expanding-and-shrinking operation, at the same time, preserve the original computation complexity.

Next we discuss how the replicated feature maps contribute to the final feature representation of each layer in the proposed approach. In a typical BNN with batch normalization and residual connection, the computation process of a single layer in our ES-BNN can be expressed as:

$$X = \text{Rep}\left(\tau^2, \mathcal{B}_A(\widetilde{A}) \circledast \mathcal{B}_W(\widetilde{W})\right),$$
$$Y = \gamma \odot \frac{X - \mu_X}{\sigma_X} + \beta + \widetilde{Y}, \tag{2}$$

where X is the expanded feature maps through replication, μ_X, σ_X, γ and β are respectively the mean, variance, scale and shift used to perform batch normalization, and $\widetilde{Y}$ denotes a residual connection. As we can see, the channel-wise parametric γ and β first make the replicated feature maps different. In addition, $\widetilde{Y}$ that accumulates the differences from previous layers diversifies these feature maps further. Therefore, a standard binary convolutional layer modified by our expanding-and-shrinking operation is still able to produce diverse output feature maps. In Figure 3, we illustrate this diversification process. It is apparent to observe that the replicated feature maps based on the expanding-and-shrinking operation turn into heterogeneous after batch normalization and residual connection.

3.3. Binary Group Convolution

Increasing τ expands the representation capacity, but it also shrinks the convolution kernels, although the overall numbers of learnable parameters in W and $\widetilde{W}$ remain the same. In the extreme case (i.e., $\tau = c_o$), there would be only one output channel generated directly by the binary convolution, undermining the diversity of output features. This trade-off between expanding and shrinking is indeed observed in our experiments (see Section 4.5 and Table 5).

In order to better balance expanding and shrinking, we introduce binary group convolution, which provides different input features and distinct batch normalization in each group, thus infusing greater diversity into the output features. As illustrated in Figure 2(c), given g groups, we start from expanding the input channels to $c_i \tau g$, and distribute them to the g-group binary convolutions, each of which takes the expanding-and-shrinking operation as described above. In a single group, to maintain computation complexity, $c_o/(\tau g)$ output channels are generated. After concatenating the output features from g groups, the c_o/τ output channels are in the end expanded to $c_o \tau g$, which readies the input channels for the next binary group convolution layer.

Group convolution is traditionally used together with the point-wise convolution [13] or the channel shuffle operation [52] due to the lack of interaction between channels in different groups. However, our binary group convolution is exempted from such a complication, and enables cross-

group information flow between multiple groups by nature. This is realized by the shrinking operation that condenses all channel information into c_o/τ output channels collected from all groups. And each group in next layer receives τ^2 copies of c_o/τ channels (as detailed in Equation 2), facilitating the exchange of information across groups.

3.4. Implementation

In practice, the scaling factor τ in expanding and shrinking, as well as the group number g in binary group convolution, are not necessarily the same across different layers. Following the general principle in network architecture design (i.e., deeper layers with increased channels for feature abstraction), we adopt larger τg in deeper layers to increase the representation capacity more. See more details in the appendix.

Apart from CNNs, our approach is also applicable to Transformers, where the feed-forward network (FFN) can be viewed as a point-wise convolution. This allows for the straightforward application of the proposed operation as described above. However, this results in substantially higher computation complexity in the self- or cross-attention layer due to the increased channels. We thus average the features of increased channels to restore to the original channel number before forwarding to an attention layer so as to preserve the initial complexity of attention. More details can be found in the appendix.

4. Experiments

In this section, we first describe the experimental setup, and then report extensive comparisons with the state-of-the-art methods on the popular benchmarks. A wide range of ablation study and related analysis are provided for in-depth understanding of the proposed approach.

4.1. Experimental Setup

Datasets. To comprehensively evaluate ES-BNN, we perform extensive experiments on a broad range of applications including image classification, object detection, as well as generative diffusion model based image super-resolution. These tasks are widely evaluated on seven benchmarks including CIFAR-10 [16], ImageNet [38], PASCAL VOC [8], Set5 [31], B100 [14], Urban100 [14] and Manga109 [33]. We follow the standard experimental protocols for fair comparisons with previous methods. We provide details of the multiple datasets in the appendix.

Implementation Details. Thanks to the flexibility of ES-BNN, all experiments are conducted by integrating the proposed operation into the official open-source codebase of original methods, with no changes to the original training settings. As for the real-valued parts in a network (e.g., the first convolution layer and the last fully-connected layer),

	Method	Baseline (Top-1)	ES-BNN (Top-1)
	Bi-Real [27]	89.7	**90.6 (+0.9)**
	IR-Net [35]	91.5	**92.9 (+1.4)**
ResNet-18	ReCU [48]	92.8	**93.9 (+1.1)**
	RBNN [22]	92.2	**93.4 (+1.2)**
	SiMaN [23]	92.5	**92.8 (+0.3)**
	AdaBin [42]	93.1	**93.5 (+0.4)**
	Bi-Real [27]	81.4	**85.9 (+4.5)**
	IR-Net [35]	86.5	**89.7 (+3.2)**
ResNet-20	ReCU [48]	87.4	**90.0 (+2.6)**
	RBNN [22]	87.8	**90.4 (+2.6)**
	SiMaN [23]	87.4	**89.7 (+2.3)**
	AdaBin [42]	88.2	**89.6 (+1.4)**

Table 1. Comparison of the top-1 accuracy (%) of baseline methods and our approach using the backbones of ResNet-18 and ResNet-20 on CIFAR-10.

we take different strategies: for the computationally intensive convolution layer, the expanding-and-shrinking operation is applied to maintain its computation complexity; while for the computationally lightweight fully-connected layer, the channels are modified directly with negligible increase in its computation complexity. More implementation details can be found in the appendix.

4.2. Resutls on Image Classification

CIFAR-10. To verify the generalizability of the proposed ES-BNN to different BNNs, we first experiment on CIFAR-10 and broadly evaluate with a series of methods, including Bi-Real [27], IR-Net [35], RBNN [22], ReCU [48], SiMaN [23], and AdaBin[42]. These methods are used as the baselines, and the proposed approach is universally applied on them. As shown in Table 1, ES-BNN consistently and considerably boosts the top-1 accuracy over all baselines. In particular, the most significant gains are obtained with 1.4% and 4.5% based on the backbones of ResNet-18 and ResNet-20, respectively.

ImageNet. To validate the proposed approach on the more challenging task, we perform the large-scale image classification on ImageNet. ES-BNN is applied upon a set of representative methods including ReCU [48], ReActNet [29], BNext [10], and BinaryViT [17]. As shown in Table 2, ES-BNN achieves superior results on both ResNet-18 and the customized CNN models. Our approach improves various baseline methods by a clear margin, while maintaining the computation complexity. Taking a closer look into the improvement, ES-BNN delivers a remarkable gain of 2.8% even on the high-performing method BNext, revealing that our enhancing effect is not diminishing with the strong baseline. Furthermore, we evaluate the proposed approach with BinaryViT [17], which is the recent binarization work on Transformers. It is found that ES-BNN is also able to render consistent and significant improvement, suggesting

	Method	BOPs (10^9)	FLOPs (10^8)	OPs (10^8)	Top-1 (%)
	BNNs [15]	1.70	1.33	1.60	42.2
	XNOR-Net [36]	1.70	1.33	1.60	51.2
	Bi-Real [27]	1.68	1.39	1.65	56.4
	XNOR-Net++ [4]	1.70	1.33	1.60	57.1
	IR-Net [35]	1.68	1.37	1.64	58.1
ResNet-18	RBNN [22]	1.68	1.37	1.64	59.9
	ReSTE [46]	1.68	1.37	1.64	60.9
	ReCU [48]	1.68	1.37	1.64	61.0
	ReCU (ES-BNN)	1.68	1.41	1.68	**64.4 (+3.4)**
	ReActNet [29]	1.68	1.37	1.64	65.9
	ReActNet (ES-BNN)	1.68	1.41	1.68	**68.0 (+2.1)**
	MeliusNet [2]	4.62	1.35	2.08	63.6
	Real2Binary [32]	1.68	1.56	1.83	65.4
	ReActNet [29]	4.82	0.12	0.87	69.4
Customized CNNs	AdamBNN [30]	4.82	0.12	0.87	70.5
	INSTA-BNN [18]	4.82	0.20	0.95	71.7
	BNext [10]	4.82	0.13	0.88	72.4
	BNext (ES-BNN)	4.82	0.14	0.89	**75.2 (+2.8)**
Transformers	BinaryViT [17]	3.83	0.19	0.79	67.7
	BinaryViT (ES-BNN)	3.83	0.20	0.80	**69.6 (+1.9)**

Table 2. Comparison of the top-1 accuracy and the computation complexity (BOPs, FLOPs and OPs) of the state-of-the-art methods and our approach using ResNet-18, customized CNNs and Transformers on ImageNet.

	Method	mAP (%)
	BNNs [15]	35.6
	XNOR-Net [36]	48.4
Faster R-CNN	Bi-Real [27]	58.2
(ResNet-18)	BiDet [43]	59.5
	AutoBiDet [44]	60.4
	BiDet (ES-BNN)	**63.5 (+4.0)**

Table 3. Comparison of the object detection results of various methods and our approach based on Faster R-CNN using ResNet-18 on PASCAL VOC.

that ES-BNN generalizes for heterogeneous architectures from CNNs to Transformers. Figure 4 quantitatively exemplifies how the representation capacity of each layer is enhanced by our approach in both standard (ResNet-18) and customized (BNext) network architectures.

4.3. Results on Object Detection

PASCAL VOC. Here we assess the generalization of ES-BNN on the task of object detection based on the dataset PASCAL VOC. BiDet [43] is used as the representative baseline method. As demonstrated in Table 3, our approach improves the baseline by 4.0% in mAP, and largely outperforms other competing algorithms. This experiment further verifies that ES-BNN is capable of adapting to different tasks from classification to detection, showcasing its efficacy in diverse downstream applications.
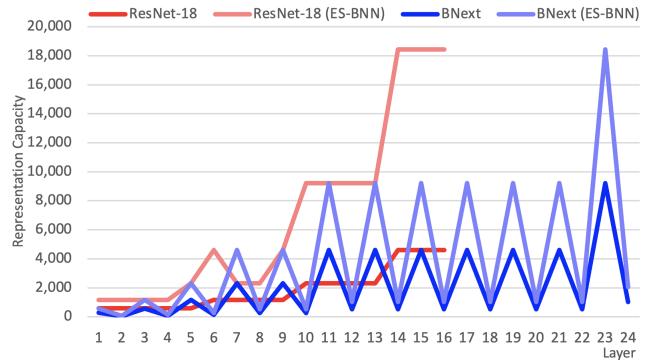

Figure 4. Comparison of the representation capacity of each individual layer before and after applying ES-BNN in a standard architecture ResNet-18 and a customized architecture BNext.

4.4. Results on Generative Diffusion Model

We adopt image super-resolution (SR) to verify the effectiveness of our approach on generative diffusion model. Due to the high-quality generation performance, the diffusion models [12, 37] have been widely used in conditional image generation tasks, including SR. However, achieving desirable results with diffusion models requires thousands of iterative steps, resulting in slow inference time. Binarization is therefore well-suited for this application and holds the potential to significantly accelerate diffusion models. We use the most recent method BI-DiffSR [5] as our baseline. Following the previous works, we employ DIV2K [41]

	Set5			B100			Urban100			Manga109		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
BNN [15]	13.97	0.5210	0.4529	13.73	0.4553	0.5784	12.75	0.4236	0.5575	9.290	0.3035	0.7489
DoReFa [53]	16.43	0.6553	0.2662	16.11	0.5912	0.3972	15.09	0.5495	0.4055	12.35	0.4609	0.5047
XNOR [36]	32.34	0.8661	0.0782	27.94	0.7548	0.1665	27.47	0.8225	0.1153	31.99	0.9428	0.0326
IRNet [35]	32.55	0.9340	0.0446	27.76	0.8199	0.1115	26.34	0.8452	0.0913	23.89	0.7621	0.1820
ReActNet [29]	34.30	0.9271	0.0351	28.36	0.8158	0.0943	27.43	0.8563	0.0731	32.16	0.9441	0.0379
BBCU [47]	34.31	0.9281	0.0393	28.39	0.8202	0.0905	28.05	0.8669	0.0620	32.88	0.9508	0.0272
BI-DiffSR [5]	35.68	0.9414	**0.0277**	29.73	0.8478	**0.0682**	28.97	0.8815	0.0522	33.99	0.9601	0.0172
BI-DiffSR (ES-BNN)	**36.35**	**0.9501**	0.0336	**30.17**	**0.8623**	0.0776	**29.48**	**0.8934**	**0.0501**	**35.23**	**0.9648**	**0.0161**

Table 4. Comparison of generative diffusion model based image super-resolution under three evaluation metrics (PSNR, SSIM and LPIPS) across four benchmark datasets (Set5, B100, Urban100 and Manga109).
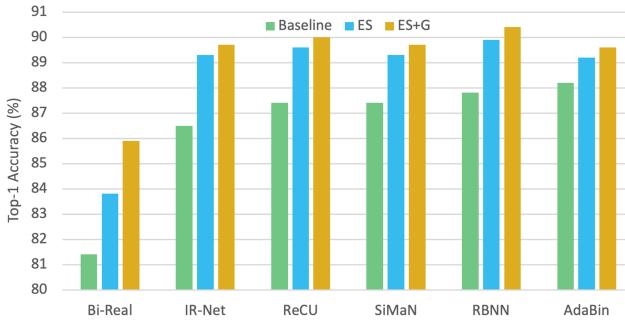

Figure 5. Comparison of our approach with different combinations of the proposed expanding-and-shrinking operation (ES) as well as the binary group convolution (G) using ResNet-20 on CIFAR-10.

	τ $(g=1)$			g $(\tau=4)$		
	Bi-Real	IR-Net	ReCU	Bi-Real	IR-Net	ReCU
1	81.4	86.5	87.5	83.8	89.4	89.6
2	83.4	88.7	89.0	85.2	89.7	89.9
4	83.8	89.4	89.6	85.9	89.6	90.0
8	82.8	89.3	89.3	-	-	-
16	80.2	88.0	88.6	-	-	-

Table 5. Analysis of the hyper-parameters τ (scaling factor) and g (group number) in ES-BNN. We report the top-1 accuracy (%) using ResNet-20 on CIFAR-10.

and Flickr2K [3] as the training set and evaluate on four benchmark datasets. We report the results of the $\times 2$ upscale factor in Table 4. As can be seen, our approach delivers superior results under most metrics across the four datasets. This application further demonstrates the versatility of ES-BNN, showing its applicability not only to high-level vision tasks like classification and detection but also to generative low-level vision tasks.

4.5. Ablation Study and Analysis

Contribution of Each Component. Here we investigate the contribution of each individual component in our approach through a set of combinatorial experiments. As shown in Figure 5, starting from the baseline methods, we incrementally integrate with the expanding-and-shrinking operation (i.e., ES) and the binary group convolution (i.e., ES+G). It can be found that the two components consistently make improvement over all baseline methods, which validate the effectiveness of the proposed design.

Hyper-Parameters. We study the two hyper-parameters in our approach: the scaling factor τ and the group number g. To first eliminate other impacts, a uniform value of τ is used throughout the whole network, and the binary group convolution is not utilized. In Table 5, we observe signif-

icant improvement in the accuracy when τ increases from 1 to 4, showing the effect of expanding representation capacity. However, the accuracy reaches a plateau or drops as τ increases further, indicating that the effect of shrinking convolution kernels becomes more prominent. Based on the optimal $\tau = 4$, combining binary group convolution can further improve the accuracy. For instance, $\tau = 4$ and $g = 4$ outperforms $\tau = 16$ and $g = 1$, although the overall expanded channels are the same in the two settings. Note the maximum value of g is 4, constrained by the limit of output channels in ResNet-20 (i.e., $c_o/(\tau g) \geq 1$).

Diversification of Replicated Features. In addition to the qualitative visualization of the diversification process in Figure 3, we provide quantitative evaluation of the diversification of replicated features. We make use of the correlation coefficient to measure the relationship strength between the replicated features. Specifically, we compute the absolute values of correlation coefficients between pairwise feature maps, which include the feature map directly generated by binary convolution and the one replicated from this feature map (see Figure 2(b)). We then take the average of all pairs as the correlation coefficient of the corresponding layer. Figure 7 plots the coefficients of all layers (apart from the first and last layers that are not binarized) in ResNet-20. Initially, the coefficients of all layers are 1. As can be seen from this figure, the coefficients are largely

Figure 6. Comparison of the loss landscape (i.e., the contour line view) of baseline methods and our approach using ResNet-20 on CIFAR-10. A real-valued counterpart is demonstrated here for reference.
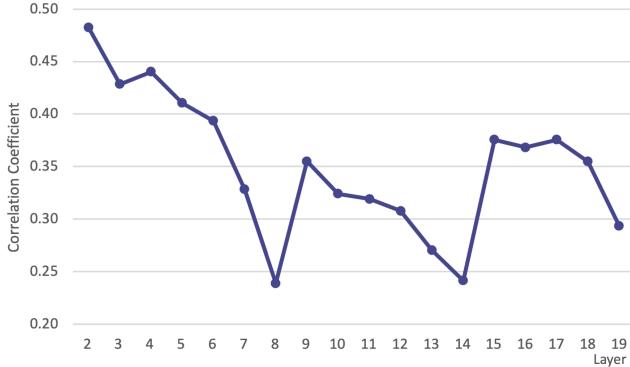

Figure 7. Quantization of the diversification of replicated features by computing the correlation coefficients of binarized layers. We report the result based on ReCU with ResNet-20 on CIFAR-10.

	Baseline	ES-BNN	Δ
Conv FLOPs (10^8)	1.373	1.373	0.000
FC FLOPs (10^8)	0.005	0.041	0.036
BOPs (10^9)	1.676	1.676	0.000
OPs (10^8)	1.640	1.676	0.036 (2.2%)
BN FLOPs (10^8)	0.025	0.082	0.057
OPs+ (10^8)	1.665	1.758	0.093 (5.6%)
Top-1	61.0	64.4	3.4

Table 6. Comparison of the computation complexity measured in different metrics (FLOPs, BOPs, OPs and OPs+). We break down the computation into convolution (Conv), fully-connected (FC), and batch normalization (BN) layers. We report the computation complexity and top-1 accuracy (%) using ResNet-18 on ImageNet.

reduced from the initial values, clearly indicating the replicated features are diversified. It can be also observed that the coefficients generally decrease with an increase in layers. Notably, abrupt shifts occur at the 9th and the 15th layers, aligning with the architectural variations of ResNet-20 (i.e., zero-padding in feature channels for residual connections, see more details in the appendix).

Loss Landscape. For the in-depth understanding of the improvement achieved through our approach, we further visualize the loss landscape (i.e., the contour line view) of multiple binarization methods before and after implementing ES-BNN. As shown in Figure 6, ES-BNN noticeably smoothes the rugged and discontinuous surfaces, making them more closely resemble that of the real-valued network. This eventually makes it easier to optimize and escape from sub-optimal convergence, suggesting the advantage of the enhanced representation capacity by our approach.

Computation Complexity. Finally, we elaborate on the computation complexity and conduct a thorough analysis of the computational impact introduced by ES-BNN. The majority of binarization research commonly employs a unified metric, referred to as OPs. It can be calculated by $\text{OPs} = \text{FLOPs} + \text{BOPs}/64$, where FLOPs and BOPs denote the number of floating point operations and binary operations. In line with the previous works, we also report the computation complexity of our approach in OPs. However,

it is noteworthy that the previous works regard the computation cost of batch normalization as insignificant and omit it from OPs. Yet the computation cost of batch normalization in our approach is increased due to the expanding operation. To better elucidate the computation complexity of our approach, we define a more comprehensive metric OPs+, which takes the computation cost of batch normalization into the overall measurement. Table 6 shows the breakdown of computation complexity in multiple metrics. As we can see, ES-BNN incurs negligible increase in the computation complexity in terms of both OPs (2.2%) and OPs+ (5.6%), while achieving significant accuracy improvement (3.4%).

5. Conclusion

We have presented ES-BNN, which is a simple yet effective approach to improve various BNNs at negligible increase of computation complexity. ES-BNN achieves performance boosts by the proposed expanding-and-shrinking operation to enhance the representation capacity of each binary layer. Extensive experiments on multiple benchmarks reveal that ES-BNN obtains considerable gains over a wide range of binarization algorithms with the architectures of both CNNs and Transformers. Moreover, our approach shows strong generalizability in different applications from image classification, object detection to generative diffusion model.

References

[1] Ishak Ayad, Nicolas Larue1, and Mai Nguyen. Qn-mixer: A quasi-newton mlp-mixer model for sparse-view ct reconstruction. In *CVPR*, 2024. 1

[2] Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, and Christoph Meinel. Meliusnet: An improved network architecture for binary neural networks. In *WACV*, 2021. 6

[3] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012. 7, 11

[4] Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv preprint arXiv:1909.13863*, 2019. 6

[5] Zheng Chen, Haotong Qin, Yong Guo, Xiongfei Su, Xin Yuan, Linghe Kong, and Yulun Zhang. Binarized diffusion model for image super-resolution. In *NeurIPS*, 2024. 6, 7, 11

[6] Devikalyan Das, Christopher Wewer, Raza Yunus, Eddy Ilg, and Jan Eric Lenssen. Neural parametric gaussians for monocular non-rigid object reconstruction. In *CVPR*, 2024. 1

[7] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. In *ICLR*, 2020. 1

[8] Mark Everingham, Luc van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 5, 11

[9] 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 *ICCV*, 2019. 1

[10] Nianhui Guo, Joseph Bethge, Christoph Meinel, and Haojin Yang. Join the high accuracy club on imagenet with a binary neural network ticket. *arXiv preprint arXiv:2211.12933*, 2022. 2, 5, 6, 11

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 6

[13] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1, 4

[14] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 5, 11

[15] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in neural information processing systems*, 29, 2016. 2, 6, 7

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5, 11

[17] Phuoc-Hoan Charles Le and Xinlin Li. Binaryvit: Pushing binary vision transformers towards convolutional models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4664–4673, 2023. 5, 6, 11

[18] Changhun Lee, Hyungjun Kim, Eunhyeok Park, and Jae-Joon Kim. Insta-bnn: Binary neural network with instance-aware threshold. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17325–17334, 2023. 2, 6

[19] Jinyu Li, Chenxu Luo, and Xiaodong Yang. Pillarnext: Rethinking network designs for 3d object detection in lidar point clouds. In *CVPR*, 2023. 1

[20] Weixin Li and Xiaodong Yang. Transcendental idealism of planner: Evaluating perception from planning perspective for autonomous driving. In *ICML*, 2023. 1

[21] Yawei Li, Kamil Adamczewski, Wen Li, Shuhang Gu, Radu Timofte, and Luc Van Gool. Revisiting random channel pruning for neural network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 191–201, 2022. 1

[22] 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. 2, 5, 6, 11, 12

[23] Mingbao Lin, Rongrong Ji, Zihan Xu, Baochang Zhang, Fei Chao, Chia-Wen Lin, and Ling Shao. Siman: Sign-to-magnitude network binarization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):6277–6288, 2022. 5, 11, 12

[24] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2018. 1

[25] Yongfan Liu and Hyoukjun Kwon. Efficient stereo depth estimation model for wearable augmented reality devices. In *CVPR*, 2025. 1

[26] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017. 1

[27] 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. 2, 5, 6, 11, 12

[28] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3296–3305, 2019. 1

[29] 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. 2, 5, 6, 7, 11

[30] Zechun Liu, Zhiqiang Shen, Shichao Li, Koen Helwegen, Dong Huang, and Kwang-Ting Cheng. How do adam and training strategies help bnns optimization. In *International conference on machine learning*, pages 6936–6946. PMLR, 2021. 2, 6

[31] 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*, pages 416–423. IEEE, 2001. 5, 11

[32] Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. In *International Conference on Learning Representations*, 2020. 2, 6

[33] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia tools and applications*, 76:21811–21838, 2017. 5, 11

[34] Trong-Thuan Nguyen, Pha Nguyen, and Khoa Luu. Hig: Hierarchical interlacement graph approach to scene graph generation in video understanding. In *CVPR*, 2024. 1

[35] 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. 2, 5, 6, 7, 11, 12

[36] 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. 1, 2, 6, 7

[37] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 6

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 5, 11

[39] Laura Smith, Yunhao Cao, and Sergey Levine. Grow your limits: Continuous improvement with real-world rl for robotic locomotion. In *ICRA*, 2024. 1

[40] Daniel Soudry, Itay Hubara, and Ron Meir. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *NeurIPS*, 2014. 2

[41] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 114–125, 2017. 6, 11

[42] Zhijun Tu, Xinghao Chen, Pengju Ren, and Yunhe Wang. Adabin: Improving binary neural networks with adaptive binary sets. In *European conference on computer vision*, pages 379–395. Springer, 2022. 2, 5, 11, 12

[43] 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. 6, 11

[44] Ziwei Wang, Jiwen Lu, Ziyi Wu, and Jie Zhou. Learning efficient binarized object detectors with information compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):3082–3095, 2021. 6

[45] Zeyu Wang, Dingwen Li, Chenxu Luo, Cihang Xie, and Xiaodong Yang. Distillbev: Boosting multi-camera 3d object detection with cross-modal knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8637–8646, 2023. 1

[46] Xiao-Ming Wu, Dian Zheng, Zuhao Liu, and Wei-Shi Zheng. Estimator meets equilibrium perspective: A rectified straight through estimator for binary neural networks training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17055–17064, 2023. 6

[47] Bin Xia, Yulun Zhang, Yitong Wang, Yapeng Tian, Wenming Yang, Radu Timofte, and Luc Van Gool. Basic binary convolution unit for binarized image restoration network. *ICLR*, 2023. 7

[48] Zihan Xu, Mingbao Lin, Jianzhuang Liu, Jie Chen, Ling Shao, Yue Gao, Yonghong Tian, and Rongrong Ji. Recu: Reviving the dead weights in binary neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5198–5208, 2021. 2, 5, 6, 11, 12

[49] Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Making convolutional networks recurrent for visual sequence learning. In *CVPR*, 2018. 1

[50] Xiaodong Yang, Zhuang Ma, Zhiyu Ji, and Zhe Ren. Gedepth: Ground embedding for monocular depth estimation. In *ICCV*, 2023. 1

[51] Zhendong Yang, Zhe Li, Xiaohu Jiang, Yuan Gong, Zehuan Yuan, Danpei Zhao, and Chun Yuan. Focal and global knowledge distillation for detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4643–4652, 2022. 1

[52] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018. 1, 4

[53] 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. 1, 7

[54] Shilin Zhu, Xin Dong, and Hao Su. Binary ensemble neural network: More bits per network or more networks per bit? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4923–4932, 2019. 2

[55] 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. 2

Appendix

Section A presents more details regarding our implementation. Section B introduces more details about the multiple benchmark datasets and experimental results. Section C elucidates the application of ES-BNN on BinaryViT. Section D describes more context about the zero padding.

A. More Implementation Details

ES-BNN can be easily implemented in a few lines of code in PyTorch. Following illustrates the python code of integrating the proposed expanding-and-shrinking operation into a standard binary convolution block including binary convolution, batch normalization and residual connection.

```
# ichn (ochn):  number of input (output) channels
# ctau (ntau):  scaling factor (τ) of current (next) layers
# cgrp (ngrp):  groups of current (next) layers
# ifeat:        input feature maps

class ESBNN_Block(nn.Module):
    def __init__(self, ichn, ochn, ctau, ntau, cgrp, ngrp):
        super(ESBNN_Block, self).__init__()
        self.conv = BConv2d(ichn * ctau * cgrp, int(ochn /
 ctau), groups = cgrp)
        self.replication = ctau * ntau * ngrp
        self.bn = nn.BatchNorm2d(ochn * ntau * ngrp)
    def forward(self, ifeat):
        ofeat = self.conv(ifeat)
        ofeat = ofeat.repeat(1, self.replication, 1, 1)
        return self.bn(ofeat) + ifeat
```

As described in the main paper, all of our experiments are conducted by integrating the proposed expanding-and-shrinking operation into the official open-source codebase of various binarization methods, without changing their original training settings. Table 7 lists the sources that are used to perform our experiments.

Method	Source
Bi-Real [27]	github.com/liuzechun/Bi-Real-net
IR-Net [35]	github.com/htqin/IR-Net
ReCU [48]	github.com/z-hXu/ReCU
RBNN [22]	github.com/lmbxmu/RBNN
AdaBin [42]	github.com/huawei-noah/Efficient-Computing
SiMaN [23]	github.com/lmbxmu/SiMaN
ReActNet [29]	github.com/liuzechun/ReActNet
BNext [10]	github.com/hpi-xnor/BNext
BinaryViT [17]	github.com/Phuoc-Hoan-Le/BinaryViT
BiDet [43]	github.com/ZiweiWangTHU/BiDet
BI-DiffSR [5]	github.com/zhengchen1999/BI-DiffSR

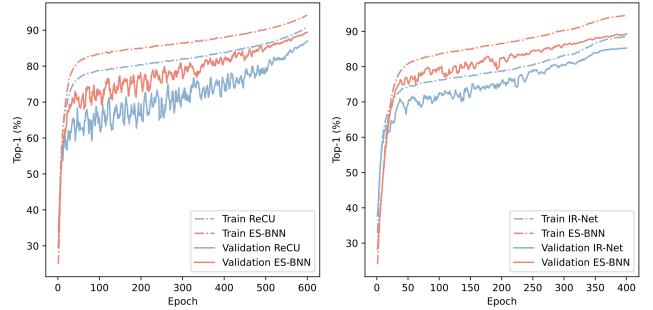Table 7. Sources of different methods used in our experiments.

Figure 8. Comparison of the learning behaviors between the two baseline methods (ReCU and IR-Net) and the proposed approach using ResNet-20 on CIFAR-10.

B. More Experimental Details

Datasets. We conduct extensive experiments on a broad range of datasets, including CIFAR-10 [16], ImageNet [38], PASCAL VOC [8], DIV2K [41], Flickr2K [3], Set5 [31], B100 [14], Urban100 [14] and Manga109 [33]. CIFAR-10 is widely used in the binarization research, and consists of 60,000 images in 10 categories. It comprises 50,000 images for training and 10,000 images for testing. ImageNet is a large-scale dataset of 1,000 categories including 1.2M training images and 50K validation images. PASCAL VOC covers 20 categories and owns two versions, i.e., VOC-2007 and VOC-2012. To be consistent with the previous methods [43], we combine the training sets from both versions, totaling 16,000 images for training, and the test set from VOC 2007 is used. DIV2K and Flickr2K consist of 1,000 and 2,650 images, respectively, and are commonly utilized as training datasets for image super-resolution. For evaluation, Set5, B100, Urban100 and Manga109 serve as widely adopted benchmarks. Set5 is a compact dataset of 5 images, often used for quick assessments. B100, derived from the Berkeley segmentation dataset [14], includes 100 images featuring diverse content. Urban100 contains 100 images of urban scenes, capturing the intricate details of urban environments. Manga109 is distinct, comprising 109 manga images, offering a unique challenge due to the presence of line art and text characteristic of manga.

Learning Behaviors. To gain the insight into how ES-BNN impacts on the learning behavior of a binary network, we provide the training process of the baseline methods and our approach. As demonstrated in Figure 8, it is observed that ES-BNN yields steady improvement in both training and validation accuracy throughout the learning process. Moreover, this improving trend is consistent across different binarization methods used as the baselines.

Feature Visualization. We further analyze the feature distribution for a deeper comprehension of the enhancement achieved by ES-BNN. Figure 9 visualizes the features extracted before the last fully-connected layer of the baselines

	τ																g															
Layer	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Param.	2	2	2	2	4	4	2	2	4	4	4	4	4	4	4	4	1	1	2	2	1	1	2	2	1	1	1	1	1	1	2	2

Table 8. Illustration of the setting of hyper-parameters τ and g.

Method	Baseline (n)	ES-BNN		
		$n\tau$	n	n/τ
Bi-Real [27]	81.4	83.4 (+2.0)	83.5 (+2.1)	83.4 (+2.0)
IR-Net [35]	86.5	88.7 (+2.2)	88.5 (+2.0)	88.4 (+1.9)
ReCU [48]	87.5	89.0 (+1.5)	89.1 (+1.6)	88.8 (+1.3)
RBNN [22]	87.8	89.4 (+1.6)	89.3 (+1.5)	89.1 (+1.3)
SiMaN [23]	87.4	89.2 (+1.8)	89.1 (+1.7)	88.9 (+1.5)
AdaBin [42]	88.2	89.5 (+1.3)	89.4 (+1.2)	89.3 (+1.1)

Table 9. Comparison of the top-1 accuracy (%) of the baseline methods and our approach using different channel numbers in the last fully connected layer.
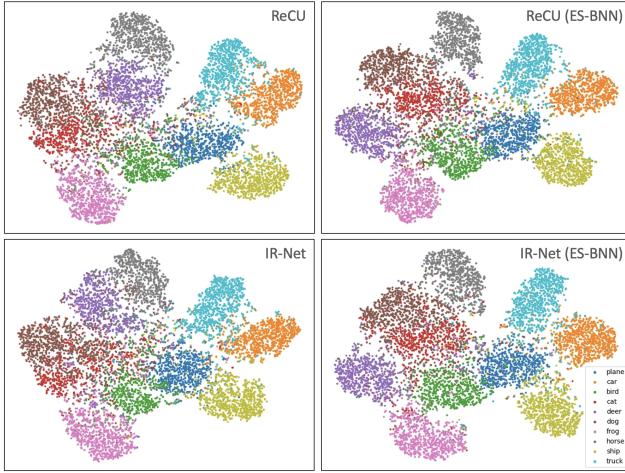

Figure 9. t-SNE visualization of the features extracted before the last fully-connected layer of baseline methods and our approach using ResNet-20 on CIFAR-10.

and our approach using ResNet-20 on CIFAR-10. It can be found that the inter-class features of our approach are overall more scattered. For instance, distinguishing truck (cyan) and car (orange) in ReCU, as well as between deer (purple) and horse (gray) in IR-Net, which is relatively challenging in the baselines, is improved in ES-BNN.

Fully Connected Layer. In the main paper, we keep using the replication for the last fully connected layer due to its negligible computation overhead. In practice, the accuracy can be still improved by keeping the number of channels in this layer unchanged or reduced. Table 9 shows the three cases in the last fully connected layer: 1) the same replication as used in convolution layers, i.e., $n\tau$ channels, 2) the number of channels is simply replicated to match the original one, i.e., n channels, 3) no replication, i.e., n/τ

channels. We experiment on multiple baseline methods under the setting of $\tau = 2$ and $g = 1$ with ResNet-20 on CIFAR-10. As can be seen in the table, ES-BNN is overall insensitive to the number of channels in the last fully connected layer. This implies that our approach can still bring significant improvement even with lower OPs than the baseline methods.

Hyper-Parameters As introduced in Section 3.4 of the main paper, we follow the principle of general network design to use larger τg for the deeper layers so as to enhance the representation capacity more in the deeper layers for feature abstraction. Table 8 shows the specific τ and g in each layer of ResNet-18.

C. More Details on BinaryViT

To provide more details of how ES-BNN is applied to BinaryViT, we present a schematic diagram in Figure 10. For the feed-forward network (FFN), we directly apply our approach to increase its representation capacity. For the attention layers (both self-attention and cross-attention), we take average of the features to revert to the original feature dimension, thereby maintaining the initial complexity of the attention layers.
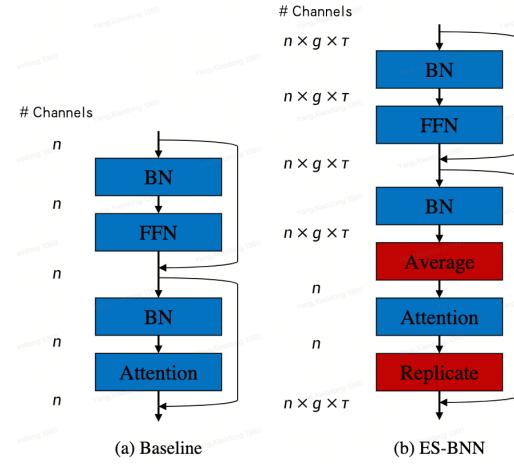

Figure 10. Illustration of how ES-BNN is applied to BinaryViT.

D. Zero Padding

As shown in Figure 7 of the main paper, the abrupt shifts at the 9th and 15th layers are due to the zero-padding in feature channels of residual connections at the corresponding layers
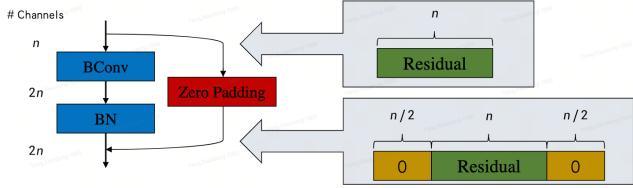
Figure 11. Illustration of zero padding of residuals in ResNet-20.

in ResNet-20. Here we provide more context. Traditionally, zero-padding is used in the spatial dimensions to preserve the input feature map size. However, in ResNet-20, zero-padding is used in the feature dimension to compensate for the feature dimension mismatch of residual connections in the 9th and 15th layers, as illustrated in Figure 11. As a result of the padded zeros, the correlation coefficients of the two layers shift dramatically.