Efficient Super Resolution Using Binarized Neural Network

Yinglan Ma*
Adobe Inc. Research
yingma@adobe.com

Hongyu Xiong*
Facebook Research
richardxiong@fb.com

Zhe Hu
Hikvision Research
zhe.hu@hikvision.com

Lizhuang Ma
East China Normal University
lzma@sei.ecnu.edu.cn

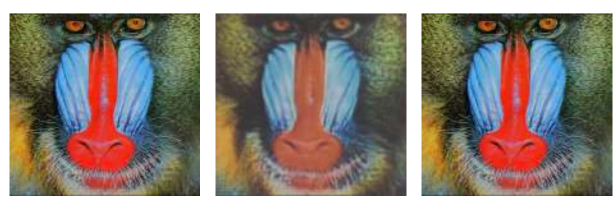
Abstract

Deep convolutional neural networks (DCNNs) have recently demonstrated high-quality results in single-image super-resolution (SR). DCNNs often suffer from over-parametrization and large amounts of redundancy, which results in inefficient inference and high memory usage, preventing massive applications on mobile devices. As a way to significantly reduce model size and computation time, binarized neural network has only been shown to excel on semantic-level tasks such as image classification and recognition. However, little effort of network quantization has been spent on image enhancement tasks like SR, as network quantization is usually assumed to sacrifice pixel-level accuracy. In this work, we explore an network-binarization approach for SR tasks without sacrificing much reconstruction accuracy. To achieve this, we binarize the convolutional filters in only residual blocks, and adopt a learnable weight for each binary filter. We evaluate this idea on several state-of-the-art DCNN-based architectures, and show that binarized SR networks achieve comparable qualitative and quantitative results as their real-weight counterparts. Moreover, the proposed binarized strategy could help reduce model size by 80% *when applying on SRResNet [18], and could potentially speed up inference by* $5\times$.

1. Introduction

The challenging task of estimating a high-resolution (HR) image from its low-resolution (LR) counterpart is referred to as super-resolution (SR). SR, particularly single image SR, has received substantial attention within the computer vision research community, and has been widely used in applications ranging from HDTV, surveillance imaging to medical imaging. The difficulty of SR is to reconstruct high-frequency details from the low-frequency information in the input image. In other words, it is to revert the non-revertible process of low-pass filter and downsampling that produces LR images.

Recent literatures have witnessed promising progress of SR using convolutional neural networks [6, 13]. However, inefficiency and large model size raise big issues for practical application of deep neural networks, due to over-parametrization. To address these issues, neural network quantizations, e.g., using binary weights and operations [5, 23], are proposed for semantic-level tasks like classification and recognition. Binarized values have huge advantages from an efficiency perspective, since multiplications can be eliminated, and bit-wise operations can be used to further reduce computational cost. Nevertheless, little effort of neural network quantization has been spent on image enhancement tasks like SR, as it was assumed to sacrifice the desired pixel-level accuracy for those tasks.



(a) (b) (c)

Figure 1. Example of binarized networks based on SRResNet [18], with a factor of $4\times$. (a) Real-weight network; (b) Fully-binarized network using binarization strategy [23]; (c) Our binarized network.

In this work, we explore a network binarization approach for SR methods. To our best knowledge, it is the first work to explore neural network binarization for image SR task. We show that simply binarizing the whole SR network does not generate satisfactory results. Therefore, we propose a binarization strategy for SR task, by (1) applying binariza-

*These two authors contributed equally.

tion only to residual blocks, and (2) using learnable weights to binarize convolutional filters.

We apply this strategy to a few state-of-the-art SR neural networks to verify its effectiveness. The experimental results show that the binarized SR network perform similarly to the real-weight network without sacrificing much image quality, but with significant model size and computational load saving, making it possible to apply state-of-the-art SR algorithms in mobile device applications, video streaming, and Internet-of-Things (IoT) edge-device processing.

2. Related Work

Single-image super-resolution has achieved significant progress in recent years. Pioneer methods on single-image super-resolution have been well discussed in the survey work [21, 29]. In this work we focus our discussion on the most updated SR paradigms based on convolutional neural network (CNN), and the progress in neural network quantization.

CNN-based super resolution. As one of the first proposed SR method based on convolutional structure, SRCNN [6] trained an end-to-end network using the bicubic upsampling of the LR image as the input. The VDSR network [13] demonstrates significant improvement over SRCNN by increasing the network depth. To facilitate training a deeper model with a fast convergence speed, VDSR aims on predicting the residuals rather than the actual pixel values. In [14], the authors provide a deep convolutional structure that allows a recursive forward propagation of a certain layer, giving decent performance and reducing the number of parameters. ESPCN [24] manages to improve the network's performance in both accuracy and speed by directly learning the upsampling filter, and using the LR images as input instead of their bicubic upsampled images. The work [7] adopts a similar idea as ESPCN but with more layers and fewer parameters. DCSCN [28] proposes a shallower CNN than VDSR, by introducing skip-connections at different levels; directly using LR image as the input, DCSCN investigates and divides the network's function to feature extraction module and reconstruction module, giving one of the highest super-resolution performance. Johnson et al. [12] investigates a perceptual loss related to human perception of high-resolution image, and incorporates the loss with the difference between high-level features from a pretrained VGG network of the predicted and target images. SRGAN [18] introduces generative adversarial network (GAN) [8, 22] into SR technique, with a combined content loss and adversarial loss for the perceptual loss function; its generator model, also described as SRResNet, adopts a deep residual framework for feature extraction and uses subpixel-convolutional layer for upscaling reconstruction. EDSR [20] moves beyond SRResNet by simplifying the the residual blocks and proposes a multi-scale SR systems, achieving even higher performances. LapSRN [16] adopts a Laplacian pyramid to predict HR image; given a fixed upscaling factor for a single level, multiple levels of pyramid could be stacked for larger upscaling factor, and the convolutional filters are shared between different pyramid levels, significantly reducing the number of parameters.

Neural network with low-precision weights. A great amount of efforts have been made to the speed-up and compression on CNNs during training, feed-forward inference or both stages. Among existing methods, the attempt to restrict CNNs weights into low-precision versions (like binary value or bit-quantized value) attracts great attention from researchers and developers. Soudry et al. [25] propose expectation back-propagation (EBP) to estimate the posterior distribution of the weights of the network, which are constrained to $+1$ and -1 during feed-forward inference in a probabilistic way. BinaryConnect [4] extends the idea of EBP, to binarize network weights during training phase directly and updating the real-value weights during the backward pass based on the gradients of the binarized weights. BinaryConnect achieves state-of-the-art classification performance for small datasets such as MNIST [17] and CIFAR-10 [15], showing the possibility that binarized CNNs can have a performance extremely close to real-value network. BinaryNet [5] moves beyond BinaryConnect, whose weights and activations are both binarized. XNOR-net [23] extends further beyond BinaryNet and BinaryConnect, by incorporating binarized convolution operation and binarized input during feed-forward inference, showing a significant reduction of memory usage and huge boost of computation speed, despite at certain level compromise of the accuracy. Later on, other than binarized networks, a series of efforts have been invested to train CNNs with low-precision weights, low-precision activations and even low-precision gradients, including but not limited to ternary weight network (TWN) [19], DoReFa-Net [32], quantized neural network (QNN) [11], and incremental network quantization (INQ) [31], but we are focusing on binary network in this paper. CNNs with low-precision weights have been shown to exhibit extremely closed performance as their real-value counterparts, on semantic-level tasks such as image classification [4] (like TWN, XNOR, QNN, INQ, etc.); however, it is widely presumed that CNNs with low-precision weights would fail on pixel-level tasks such as image reconstruction and super resolution, due to the reduced model complexity.

In this work, we are proposing an efficient network binarization strategy for image super-resolution tasks, based on BinaryNet and XnorNet [5, 23], to speed up inference and simultaneously achieve similar image quality as the full-precision network. To verify the effectiveness of the proposed strategy for general CNN-based SR methods, we evaluate two state-of-the-art SR models SRGAN [18] and LapSRN [16], and compare them with their real-value counterparts, on three benchmark metrics the peak-signal-to-noise-ratio (PSNR), structure similarity (SSIM) [27], and information fidelity criterion (IFC) calculated on y-channel to evaluate the performance of super resolution.

3. SR Network Binarization

The original binary neural networks [5, 23] are designed to binarize the parameters of convolution layers, and are shown to be effective for semantic-level tasks, without sacrificing much accuracy. Unlike semantic-level tasks, SR is an image reconstruction task with a focus on the pixel-level accuracy. And simply applying the binarization strategy [5, 23] to the SR networks does not work well. Figure 1(b) shows an example from a fully binarized (binarize all the parameters of convolution layers) SRResNet network [18] with a factor of $4\times$, using the strategy [23]. The results show that the fully binarized network does not generate satisfying results. In fact, this network does not converge well, due to the limited network capacity and the variation of the pixel-level reconstruction. Therefore, network binarization needs to be specially designed for the SR task, as well as other image reconstruction tasks.

3.1. Motivation

For a neural network, deeper network structure usually means more representation capacity and therefore better performance, but higher difficulty of convergence. Residual blocks [9] are designed to facilitate the training process of very deep neural networks by a skip connection of the original feature. Recent SR algorithms utilize residual structures in their deep neural networks to effectively forward color and structure information from the input image, and achieve decent results.

The residual structure has some similarities to the multi-scale image pyramids that are commonly used for image reconstruction tasks [1, 3]. Image pyramids represent an image with a series of band-pass filtered images. The base layer contains the low-frequency information and the gradient layers capture sub-band information and most high-frequency details. Similarly, SR tasks can be considered as reconstructing high-frequency information based on a low-pass filtered image (LR image). And the recent residual-structure-based SR networks function in a similar way as the image-pyramid reconstruction process, where the residual structure and the skip connection in neural network resemble to the gradient layers and the upsampled images from coarser scales in image pyramid.

In image pyramid, the gradient layers are known to hold good sparsity property. We represent images using the Laplacian pyramid [3], and collect the statistics of the intensity values in the gradient layers (See Figure 2). As shown

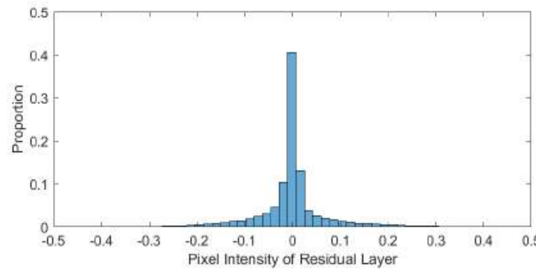


Figure 2. Statistics on histograms of the gradient layers in the Laplacian pyramid. The range of the image pixel intensity is $[0, 1]$.

in the figure, value distribution is highly centralized around 0 and most of the pixels ($\approx 40\%$) has a value of 0. Those facts indicate that the gradient layer of a Laplacian pyramid can be approximated by a layer with a small number of values, and therefore motivate us to represent it via binary filters with scaling factors in a neural network.

3.2. Binarization Method

Binary weights, from its quantization property in nature, would cause information loss comparing to real-value weights. Since SR tasks aim at enhancing image details from its original image, binary weights should be carefully used in the network. Based on the analysis in Section 3.1, we propose a SR network binarization strategy that binarizes the residual blocks and couples each binary filter with a learnable weight in the network. The convolutional filters outside the residual blocks will be kept in real-values, and the propagation and parameter update steps are the same as usual.

Algorithm 1 Binary Forward Propagation

for $l = 1$ to L layer **do**
  for $k = 1$ to K output channel **do**
    $\alpha_{lk} \leftarrow \text{UpdateAlpha}()$ //using (4)
    $B_{lk} \leftarrow \text{sign}(W_{lk})$
  end for
end for
$I^{SR} \leftarrow \text{BinaryForward}(I^{LR}, \alpha B)$

To binarize residual blocks in the network, we first define our binarization function as in [5] to transform real values to $+1$ or -1:

$$x^b = \text{sign}(x) \begin{cases} +1, \text{if } x \geqslant 0, \\ -1, \text{otherwise}, \end{cases} \tag{1}$$

where x is the real-value variable, and x^b is the corresponding binarized weight. For the practical applications, booleans can be used to represent the binarized weights. In the binarized network, a scaling factor α is assigned to each

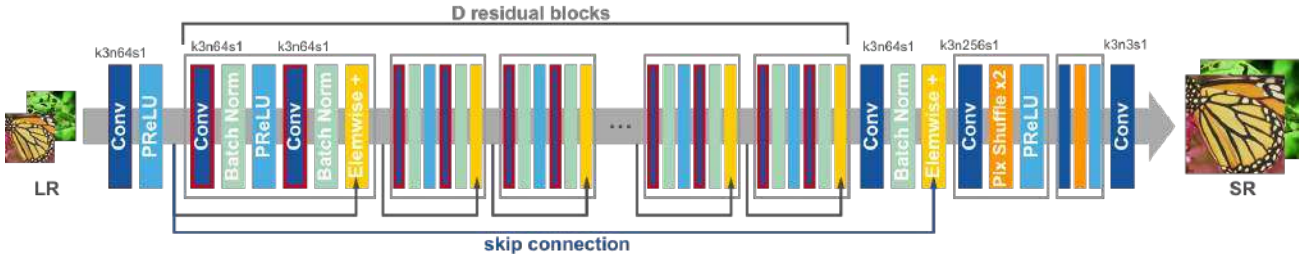


Figure 3. System architecture of the generator network in SRResNet/SRGAN [18] that transforms the low-resolution input image I^{LR} to high-resolution image I^{SR}. The red boxes indicate the binarized layers in our network. k is the filter size, n denotes the number of filters and s represents the stride number in convolution layer.

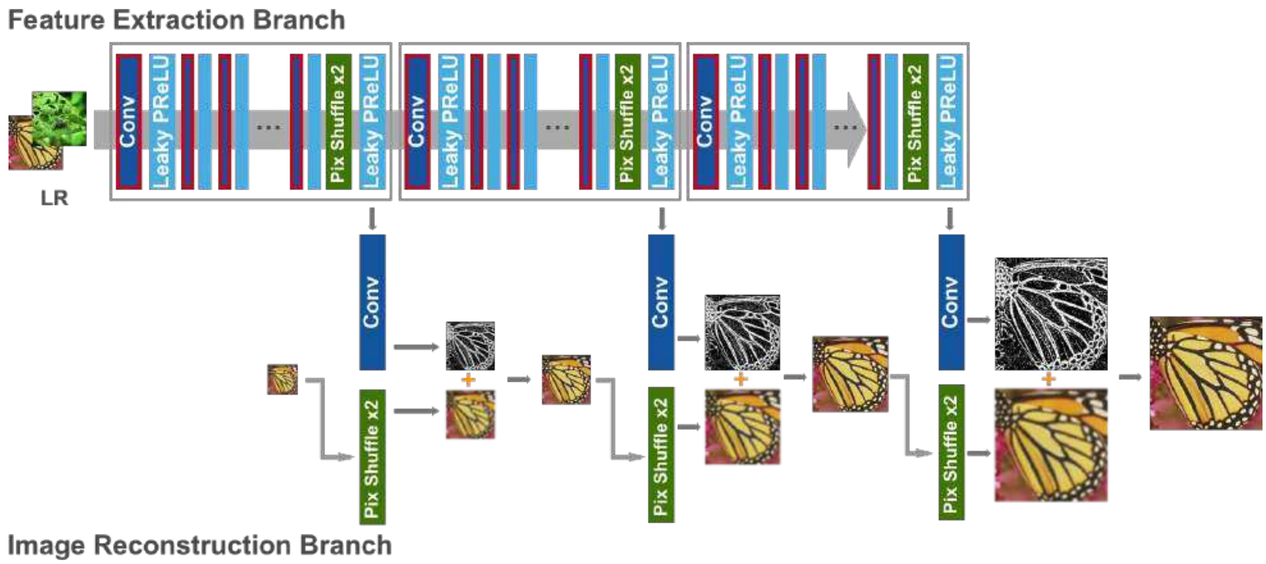


Figure 4. System architecture of the Laplacian Pyramid network [16] for SR. The red boxes indicate binarized layers in our network.

binarized filter, so a real-value weight filter W in the convolutional neural network is replaced with $\alpha \cdot \text{sign}(W)$. The factor α is a vector of the same length as the output channel size of the corresponding convolutional layer.

At each iteration of the training phase, it takes LR image patches I^{LR}, corresponding HR image patches I^{HR}, the cost function $C(I^{HR}, I^{SR})$ and the learning rate η as inputs, and outputs updated weights W^t. For the forward propagation, we first update the factor α and binarize the real-value weight filter $B = \text{sign}(W)$, then αB is used for the following computation. Algorithm 1 demonstrates the procedure of forward propagation, and we will describe α updating strategy in Section 3.3. For the backward propagation, the gradients of the weights are kept in real values. Similar to [23], we take derivatives of l-th layer cost C_l with respect to the binarized weight B_{lk} as $\frac{\partial C_l}{\partial B_{lk}}$. The gradients are clipped to range $(-5, 5)$ for stability, and are used to update the real weight by $\frac{1}{\alpha_{lk}} \frac{\partial C_l}{\partial B_{lk}}$. The real-value parameters and binary parameters are then updated by accumulating gradients as shown in Algorithm 3.

Algorithm 2 Binary Backward Propagation

for $l = L$ to 1 layer **do**
  for $k = 1$ to K output channel **do**
    $g_{W_{lk}} \leftarrow \text{Clip}(\frac{1}{\alpha_{lk}} \frac{\partial C_l}{\partial B_{lk}}, -5, 5)$
    $W_{lk} \leftarrow \text{UpdateBinaryParameter}(W_{lk}, \eta, g_{W_{lk}})$
  end for
end for

3.3. Updating Scaling Factors for Binary Filters

In [23], the scaling vector α is determined in a deterministic way, by solving the following optimization:

$$J(B, \alpha) = ||W - \alpha B||^2, \tag{2}$$

where the binary filter $B = \text{sign}(W)$ is obtained by the binarization function. The optimal value of the scaling factor $\alpha*$ is calculated as:

$$\alpha^* = \frac{1}{n} ||W||_{l1}. \tag{3}$$

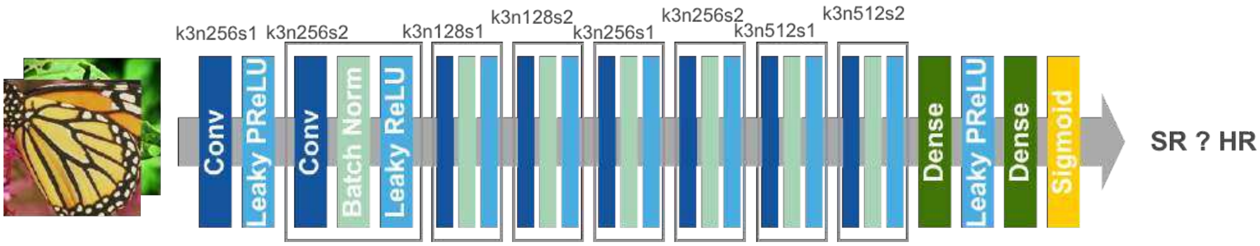


Figure 5. System architecture of the discriminator network in SRGAN [18] that is trained to distinguish super-resolution results I^{SR} from high-resolution image I^{HR}. k is the filter size, n denotes the number of filters and s represents the stride number in convolution layer.

However, this α^* is optimized for approximating W with αB, instead of optimized for closing the gap between the prediction and ground truth in pixel-level.

Algorithm 3 Accumulating Parameter Gradients

for $t = 0$ to T **do**
  $\theta^{t+1} \leftarrow$ UpdateNonBinaryParameter$(\theta^t, \eta^t, g_\theta)$
  $W^{t+1} \leftarrow$ UpdateBinaryParameter(W^t, η^t, g_W)
  $\eta^{t+1} \leftarrow$ UpdateLearningRate(η^t)
end for

To better approximate the gradient information, we propose to train α as a parameter in the network,

$$\alpha_c^t \leftarrow \text{UpdateAlpha}(\alpha_c^{t-1}, \eta, g_{\alpha_c}), \tag{4}$$

instead of

$$\alpha_c \leftarrow \text{avg}(\text{abs}|W_c|_1). \tag{5}$$

Experimental comparison between these two methods will be provided in Section 4.

4. Experiment

To verify the proposed binarization strategy for SR tasks, we evaluate this strategy on several state-of-the-art SR architectures: SRResNet/SRGAN [18] and LapSRN [16]. We evaluate the methods using NTIRE 2017 dataset [26], which includes 800 HR training images and 100 HR validation images as our training and test datasets. To train/test $2\times$ and $4\times$ models, the corresponding LR images are generated using the bicubic downsampling, with a scale factor of $2\times$ and $4\times$, respectively.

4.1. Training Details

We test our binarization strategy on three different models, SRResNet, SRGAN, and LapSRN, all of which employ residual blocks in their networks. We will describe the models and training details for each method below. For all the three models, we set larger learning rates for the binary versions, with a factor of $3\times$-$4\times$ comparing to those for the real-value network. The reason is that it requires larger momentum to change the binary weights/ switch signs. Within this learning-rate setting, the binary network converges at a similar pace as the real-value network. We show the learning curves for both real-value networks and their binary weight counterparts in the supplementary material. We use a batch size of 16 for all the training. For data augmentation, we adopt the following approaches: 1) randomly cropping patches; 2) randomly flipping horizontally or vertically, rotation and noise; 3) randomly rotating images by $\{0, 90, 180, 270\}$ degrees; 4) adding Gaussian noise to HR training patches.

SRResNet. We provide the binarized SRResNet structure in Figure 3 and show the binarized layers using red boxes. Basically, we binarize all the convolutional layers in the residual blocks, using the algorithms 1, 2 and 3. We train real-weight and binary-weight SRResNets for 500 epochs, with each epoch containing 50 iterations. We use 1×10^{-4} and 3×10^{-4} respectively, as our initial learning rate, and a decay of 0.9 in every 20 epochs.

SRGAN. We apply the same binarization method used for SRResNet to the generator of SRGAN. We keep the discriminator in real weights, as the purpose of binarization is to improve inference performance in terms of speed and model storage size and the discriminator does not affect inference performance. We pretrain real-weight and binary-weight generators of SRGAN for 100 epochs with learning rates of 1×10^{-4} and 3×10^{-4} respectively. Then we jointly train generator and discriminator for 500 epochs with a learning rate of 1×10^{-4} as our initial learning rate, and a decay of 0.9 in every 20 epochs.

LapSRN. The Laplacian Pyramid network proposed by Lai et al. [16] consists of D residual blocks in the Feature Extraction Branch. We binarized all the weights of convolutional layers in residual blocks, as shown in Figure 4. We train real-weight and binary-weight LapSRNs for 300 epochs, with each epoch containing 800 iterations. We use 3×10^{-5} and 1×10^{-4} respectively, as the initial learning rates, and a decay of 0.8 in every 30 epochs.

4.2. Evaluation Results

We evaluate our models on three widely used benchmark datasets **Set5** [2], **Set14** [30], and **Urban100** [10]. To eval-

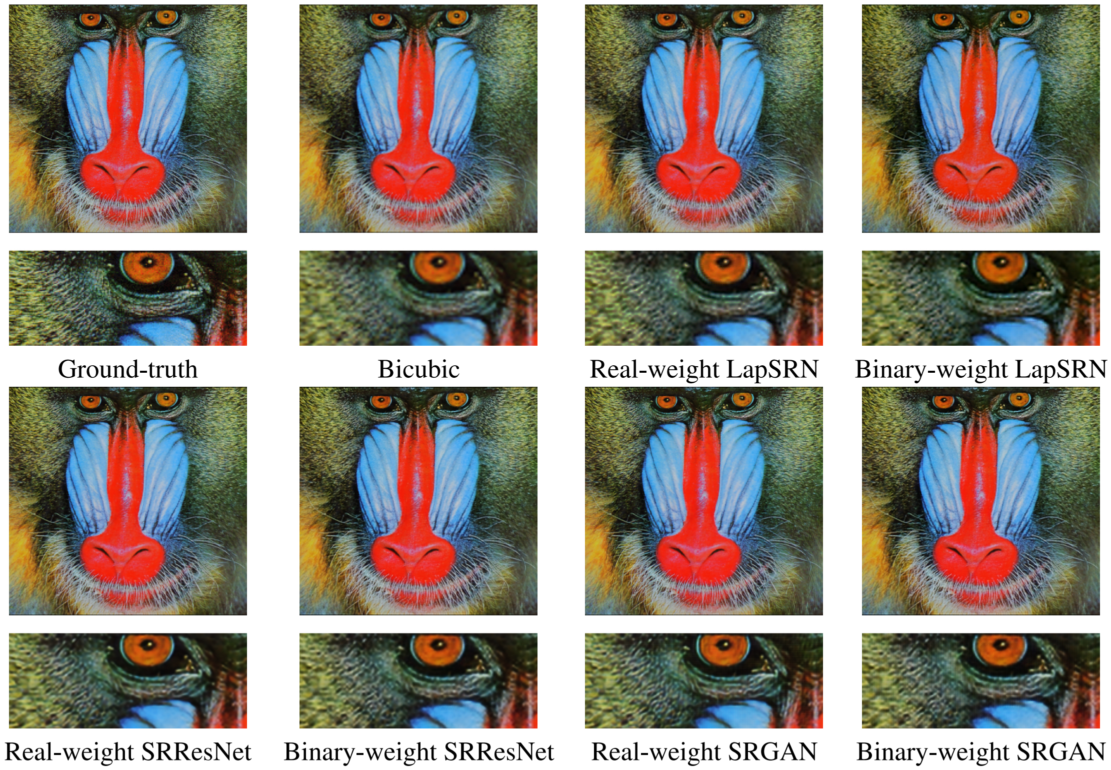


Figure 6. Comparisons between real-weight networks and their binarized versions with an upsampling factor of 2.

Figure 7. Comparisons between real-weight networks and their binarized versions with an upsampling factor of 4.

Algorithm	Scale	**Set5** PSNR/SSIM/IFC	**Set14** PSNR/SSIM/IFC	**Urban100** PSNR/SSIM/IFC
Bicubic	4	28.43 / 0.811 / 2.337	25.90 / 0.704 / 2.246	23.15 / 0.660 / 2.386
SRResNet [18]	4	31.10 / 0.875 / 3.091	27.59 / 0.764 / 2.743	25.09 / 0.750 / 3.040
SRResNet Binary (ours)	4	30.34 / 0.864 / 3.052	27.16 / 0.756 / 2.749	24.48 / 0.728 / 2.913
SRGAN [18]	4	30.43 / 0.855 / 2.862	27.00 / 0.749 / 2.493	24.75 / 0.734 / 2.865
SRGAN Binary (ours)	4	30.13 / 0.853 / 2.547	26.98 / 0.746 / 2.336	24.31 / 0.716 / 2.499
LapSRN (4x) [16]	4	30.28 / 0.858 / 2.851	27.15 / 0.748 / 2.566	24.37 / 0.722 / 2.755
LapSRN Binary (4x) (ours)	4	30.21 / 0.857 / 2.865	27.13 / 0.751 / 2.616	24.31 / 0.720 / 2.735
Bicubic	2	33.69 / 0.931 / 6.166	30.25 / 0.870 / 6.126	26.89 / 0.841 / 6.319
SRResNet [18]	2	36.36 / 0.952 / 6.714	32.16 / 0.904 / 6.577	29.96 / 0.901 / 7.016
SRResNet Binary (ours)	2	35.66 / 0.946 / 5.936	31.56 / 0.897 / 5.893	28.76 / 0.882 / 6.112
SRGAN [18]	2	35.31 / 0.941 / 6.332	31.81 / 0.901 / 6.280	29.63 / 0.897 / 6.646
SRGAN Binary (ours)	2	34.91 / 0.938 / 5.840	30.92 / 0.892 / 6.354	28.55 / 0.878 / 6.490

Table 1. Quantitative evaluation of state-of-the-art SR algorithms and their binarized versions.

uate our method, we provide comparisons of the SR images from the following models and their corresponding binarized versions: SRResNet [18], SRGAN [18], and LapSRN [16]. The results of the bicubic upsampling method is provided as a baseline. More experiment results can be found in the supplementary material.

As shown in Figure 6 and 7, the binarized networks using the proposed method perform similarly to their real-weight counterparts. Although achieving comparable results, the binarized networks generate results with slightly more aliasing artifacts compared with their real-weight versions. That is, the binarization strategy still compromises performance in terms of high-frequency details, due to the limited network capacity.

Model (4x)	deterministic	learnable
SRResNet	27.06/0.754/2.712	**27.16/0.756/2.749**
LapSRN	27.02/0.747/2.604	**27.13/0.751/2.616**

Table 2. Comparison on networks using deterministic and learnable scaling factors. The results are evaluated on SRResNet [18] and LapSRN [16] structures in PSNR/SSIM/IFC.

For quantitative evaluation, we calculate PSNR, SSIM and IFC metrics [27] between generated images and the ground truths, and all the reported values in Table 1 are calculated on the Y-channel of the YUV color space. Both the real-weight and the binary-weight networks achieve better results than the bicubic baseline in terms of the metrics. In most of the cases, the binary-weight networks perform similarly with their real-weight counterparts, with a difference less than 0.5 in PSNR or 0.005 in SSIM. And the margin becomes even smaller for the upsampling factor of 4.

# of Res-Blocks	PSNR	SSIM	IFC
8	26.74	0.738	2.136
16	27.16	**0.756**	2.749
24	**27.19**	0.754	**2.774**

Table 3. Performance of binarized SRRestNet (4x) with 8, 16 and 24 residual blocks on Set14 [30] dataset.

4.3. Model Analysis

To verify the effectiveness of the proposed learnable scaling factor for binary filters, we conduct experimental comparisons between networks using the learnable scaling factors and the deterministic ones in Table 2. In general, using learnable scaling factors has faster convergence and achieve better results comparing to the deterministic ones.

The binarized networks reduce the computational load and enable efficient inference. As deeper networks have better representation abilities and usually generate better results, we also evaluate the binarized networks in terms of the network depth. We train SRResNet-Binary (4x) with 8, 16 and 24 residual blocks, and show the quantitative results in Table 3. We note that we use 16 residual blocks for the SRResNet in all other experiments, same as in [18]. From the results, we can see that deeper networks perform better than shallower ones in general. However, marginal improvement is achieved after using more residual blocks over 16, and it suggests that binarized layers in 16 residual blocks are sufficient to represent the high-frequency details of images.

4.4. Model Size and Computational Complexity

In Table 4, we list the number of parameters and model size of the networks that we test. We use 32-/64-bit floating point precision to represents real parameters, and the parameter binarization would reduce it to a single bit, which

Algorithm	# of Res-Blocks / Res-Conv-Layer	# of Binary Params Binary/Real Network	# of Real Params Binary/Real Network	Model Size Binary/Real Network
SRResNet (2x)	16 / 32	1,179,648 / 0	197,059 / 1,374,659	0.928 / 5.499MB
SRResNet (4x)	8 / 16	589,824 / 0	339,651 / 928,451	1.428 / 3.714MB
SRResNet (4x)	16 / 32	1,179,648 / 0	344,771 / 1,522,371	1.518 / 6.089MB
SRResNet (4x)	24 / 48	1,769,472 / 0	349,801 / 2,116,201	1.608 / 8.465MB
LapSRN (4x)	10 / 10	368,640 / 0	152,656 / 520,656	1.494 / 2.083MB

Table 4. Number of parameters and model size of binary and real-weight networks (32-bit floating point). The parameters of binary networks include binary parameters of filters and float parameters assigned to filters.

Vote Percentage	$2\times$	$4\times$
SRResNet	0.50	0.45
SRResNet-Bin	0.50	0.55
SRGAN	0.60	0.62
SRGAN-Bin	0.40	0.38
LapSRN	0.51	0.55
LapSRN-Bin	0.49	0.45

Table 5. User study for $2\times$ and $4\times$ SR results from binary and real-weight networks. The test images are randomly selected from **Set5** [2] and **Set14** [30].

is a factor of 32 or 64 in terms of memory used for model storage. Even though this binarization strategy only applies to residual blocks in the network, it still yields a $50\%-80\%$ model compression depending on the network structure. Higher portion of residual components would yield higher compression rate. This also provides design guideline for applications with memory limitation, like on-device processing in IoT edge devices and mobile platforms.

Another benefit of network binarization is to reduce computational complexity during inference, as binarized filter weights enables bit operations instead of float multiplication (flops) when computing convolution. For a single convolution layer with input tensor size (W, H, C) and convolutional filters of size (K, K, C, F), where W, H, C represent the width, height, and number of input channels of the intermediate state, and F represent the number of output channels; there are in total $O(WHCFK^2)$ multiplication floating-operations (flops), which will be replaced with bit operations after binarization. Modern computers can compute 4 flops or 512 bit operations per clock cycle, and the operation of addition is $\sim 3\times$ faster than that of multiplication, For a SRResNet 4x model with 16 residual blocks on an image of size $1200 \times 800 \times 3$, there are in total $\sim 10^{14}$ flops, with about the same number of flops of multiplication and addition. In this model, more than 75% of multiplication flops are replaced with bit operations after binarization. Thus, there is a potential speedup of $\sim 2\times$ for this SRResNet(4x) model. Applying same calculation to SRResNet(2x) model results in $\sim 5\times$ computational gain.

4.5. User Study

To better evaluate the visual quality of the results from real-weight networks and their binarized versions, we conducted a user study on the SR results. We develop a web-based system to display and collect study results. The system provides two side-by-side images at a time, one from the real-weight network with scaling factor $2\times$ or $4\times$, and another from its binary counterpart. Each pair of images is randomly selected and placed from the dataset **Set5** [2] and **Set14** [30]. We have collected results of their preferred images from 24 users, and each user is asked to rate 20 pairs of images. The results are shown in Table 5: our binarized SR models perform similarly as real-weight SR models.

5. Limitation and Future Work

The proposed binarization strategy are designed for SR tasks, but it is possible to apply this strategy to other pixel-level tasks like denoising and deblurring. However, specific network designs are needed when applying binarization to other tasks, e.g., special handling may be needed for the kernel estimation process in deblurring. And how to apply the binarization strategy to other pixel-level image reconstruction tasks could be a good future research direction.

Although we are able to replace the float multiplications with bit-wise operation when computing convolution, there are still rooms of efficiency improvement using binary network. How to binarize input images for further bit-wise computation remains a difficult and open question.

6. Conclusions

In this paper, we introduce a network binarization strategy for super resolution, without losing much per-pixel accuracy. Inspired by the structure and statistics on the gradient histogram of the Laplacian pyramid, we argue that it is appropriate to pose binarization components in residual architectures, and assign a learnable parameter to each binary convolutional filter. Qualitative and quantitative experiments have shown that residual-based SR networks with binarized components can generate comparable results to their real-weight counterparts, and obtain significant improvement in model size and computational complexity.

References

[1] Edward H Adelson, Charles H Anderson, James R Bergen, Peter J Burt, and Joan M Ogden. Pyramid methods in image processing. *RCA engineer*, 29(6):33–41, 1984. 3

[2] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012. 5, 8

[3] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. In *Readings in Computer Vision*, pages 671–679. Elsevier, 1987. 3

[4] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems 28*, pages 3123–3131. 2015. 2

[5] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016. 1, 2, 3

[6] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016. 1, 2

[7] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407. Springer, 2016. 2

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 3

[10] 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] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061*, 2016. 2

[12] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016. 2

[13] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1646–1654, 2016. 1, 2

[14] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1645, 2016. 2

[15] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009. 2

[16] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. *CoRR*, abs/1704.03915, 2017. 2, 3, 4, 5, 7

[17] Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010. 2

[18] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016. 1, 2, 3, 4, 5, 7

[19] Fengfu Li, Bo Zhang, and Bin Liu. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016. 2

[20] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1132–1140. IEEE, 2017. 2

[21] Kamal Nasrollahi and Thomas B. Moeslund. Super-resolution: A comprehensive survey. *Mach. Vision Appl.*, 25(6):1423–1468, Aug. 2014. 2

[22] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015. 2

[23] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. *CoRR*, abs/1603.05279, 2016. 1, 2, 3, 4

[24] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 2

[25] Daniel Soudry, Itay Hubara, and Ron Meir. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 963–971. 2014. 2

[26] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, Kyoung Mu Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1110–1121. IEEE, 2017. 5

[27] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004. 3, 7

[28] Jin Yamanaka, Shigesumi Kuwashima, and Takio Kurita. Fast and accurate image super resolution by deep cnn with

skip connection and network in network. In *International Conference on Neural Information Processing*, pages 217–225. Springer, 2017. 2

[29] Chih-Yuan Yang, Chao Ma, and Ming-Hsuan Yang. Single-image super-resolution: A benchmark. In *Proceedings of European Conference on Computer Vision*, 2014. 2

[30] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010. 5, 7, 8

[31] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017. 2

[32] 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. 2