Towards Robust Full Low-bit Quantization of Super Resolution Networks

Denis Makhov[1⋆], Ruslan Ostapets[1], Irina Zhelavskaya[1], Dehua Song[1], and Kirill Solodskikh[1⋆⋆]

Noah's Ark Lab, Huawei Technologies, makhovds@gmail.com

Abstract. Quantization is among the most common strategies to accelerate neural networks (NNs) on terminal devices. We are interested in increasing the robustness of Super Resolution (SR) networks to low-bit quantization considering mathematical model of natural images. Natural images contain partially smooth areas with edges between them. The number of pixels corresponding to edges is significantly smaller than the overall number of pixels. As SR task could be considered as ill-posed restoration of edges and texture, we propose to manually focus quantized CNNs on high-frequency part of the input image thus hiding quantization error in edges and texture providing visually appealing results. We extract edges and texture using well-known edge detectors based on finite-difference approximations of differential operators. To perform inverse transformation we propose to use solver for partial differential equations with regularization term that significantly increase solution robustness to errors in operator domain. The proposed approach significantly outperforms regular quantization counterpart in the case of full 4-bit quantization, for example, we achieved +3.75 dB for EDSR x2 and +3.67 dB for RFDN x2 on test part of DIV2K.

Keywords: quantization error redistribution, neural network quantization, low-bit quantization, super resolution, differential operator, partial differential equations

1 Introduction

Neural networks (NNs) achieve state-of-the-art results in different low-level vision tasks including super resolution (SR). The size of such networks typically varies from several convolutional layers [9,27] to hundreds [18,20,33]. To achieve new state-of-the-art results in SR, researchers usually either increase the capacity of the network or design new types of NN architectures. In the recent decade, with the evolution of mobile chips, the SR task is encountered not only on desktops or servers, but also on edge devices with strict requirements on the latency and power consumption.

There are different approaches to increase the efficiency of SR CNNs for deployment on edge devices. These include pruning and knowledge distillation for

⋆ Currently affiliated with Samsung AI Center
⋆⋆ Currently affiliated with theStage.ai

architecture optimization, and quantization for improving both performance and power consumption on edge devices. Quantization of neural networks is a general approach for different architectures and is achieved by decreasing the bitwidth of floating point (FP) weights and activations to integer ones. Integer computations are faster and moreover require less memory read-write operations [6,25,31,32].

Integer representation of neural network weights and activations leads to a smaller model capacity which makes task execution more challenging. Depending on the task, low-bit quantization leads to different levels of quality drop. For example, for high-level vision tasks such as image classification there exist binary networks that can match the performance of FP models [8,21,34]. This implies that the input-output relation does not necessarily require high bitwidth for successful completion of the task. This can be attributed to the fact that essential image features for classification may be potentially represented even with binary features, *i.e.*, such features describe whether a particular pattern exists in the image or not. In contrast to high-level vision tasks, efficiently dealing with the SR task becomes challenging when the intermediate activations are represented with bitwidth lower than that of the input image, as pixel-level accuracy is required. Internal feature maps of SR CNNs usually contain both smooth areas, edges and textures (see Fig. 1). Step-like artifacts typically appear on smooth areas after quantization, which can be hard for NN to distinguish from real edges.

Fig. 1: FP and 4-bit quantized representation of input image (a) and (d), one channel of the 1-st convolution (b) and (e) and one channel of the third convolution (c) and (f), in trained SRCNN respectively.

Most recent advances in SR network quantization [13,14,26] consider the influence of quantization on the internal quantized feature maps without considering the essential properties of natural images and corresponding internal feature maps. According to compressed sensing theory, natural images can be considered as partially smooth areas divided by the first type discontinuities on edges. One of the key assumptions in the compressed sensing is that the number of edges is much smaller than the overall number of pixels. This assumption inspired us to make an attempt to redistribute quantization error to a small number of edge pixels as this should provide visually appealing results. Considering finite difference approximation kernels of the first and higher order derivatives, we expect that we will achieve extremely sparse image representations for natural images. These kernels are high-pass filters (e.g., Laplacian) and thus the quantization error in such domains will only influence edges in the original image domain. Based on several papers [28,30], we finally assume that such a representation can be sufficient to solve the SR task.

The main drawback of current solutions for low-bit quantization of SR networks is that the first and last convolutions usually remain FP. As first and last

convolutions work with a spatial size comparable with the size of the original image in general, they are extremely computationally expensive. Thus, making SR network fully quantized will significantly reduce its inference time. In this paper, we address the problem of preserving the quality of SR CNNs, while being fully quantized in bitwidth lower than the original input image using a partially-smooth image prior.

Overall, the contribution of this paper is three-fold:

1. We propose to redistribute quantization error using partially-smooth image prior into minority of edge pixels using finite-difference approximations of differential operators;
2. We achieve robust restoration of the image to the original domain from its operator image by regularized partial differential equation solver;
3. The proposed pipeline is capable of full 4-bit quantization and achieve close to FP performance for various SR networks.

2 Preliminaries

In this section, we provide a brief overview of uniform quantization and peculiarities in CNN quantization.

2.1 Uniform Quantization

Quantization is a process of converting FP values into integer ones performed in the following way:

$$x_{int} = clip(\lfloor \frac{x}{s} \rceil + z; 0, 2^b - 1), \tag{1}$$

where x_{int} is an integer representation of FP value x, s is a scale factor, z is an offset, b is the bitwidth of the integer representation. Operator $\lfloor \rceil$ denotes rounding-to-nearest operator and $clip()$ refers to the following function:

$$clip(x, a, b) = \begin{cases} x, a \leq x \leq b \\ a, x < a \\ b, x > b \end{cases}. \tag{2}$$

To perform equivalent calculations with integer representations, we need to move back to FP values using the so-called dequantization step:

$$x \approx \hat{x} = s(x_{int} - z), \tag{3}$$

where $\hat{x}$ is an FP representation of the quantized value. So even the number becomes FP it can take only 2^b unique values. For more details on neural network quantization we highly recommend the white paper [25].

2.2 Quantization Errors

Quantization may be viewed as projecting FP values onto a uniform grid of integer values. There are two types of errors that follow from Eq. 3: rounding

and clipping errors (see Fig. 2). The former occurs due to the round-to-nearest operation and the latter due to the clip operation. Clipping errors occur since projecting a non-uniformly distributed tensor onto a uniform integer grid using its minimum and maximum values leads to a sub-optimal scale. Therefore, usually we can sacrifice some outlier values to be clipped to increase the accuracy for the majority of values. This usually happens as the distribution of weights and activations in DNNs are non-uniform and sometimes skewed. For optimal quantization of a neural network, we need to find an optimal trade-off between the rounding and clipping errors so that the final quality is the best.

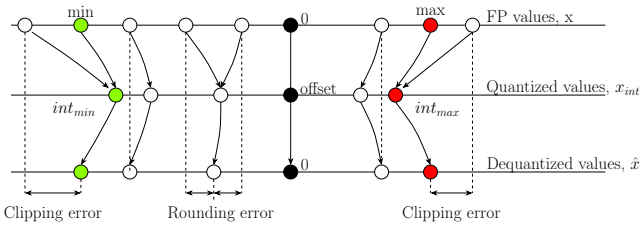

Fig. 2: Different types of quantization errors after quantization and dequantization steps.

To lower quantization errors for non-uniformly distributed data, methods such as nonuniform quantization can be used. Nonuniform quantization tries to make quantization error lower by projecting FP values onto a nonuniform grid, thus making more accurate conversion for the majority of elements. Nonuniform quantization can be implemented in two ways: using a nonuniform grid for input data and a uniform grid for the output data [22], or using both nonuniform representations for input and output [6,10,12,16,19]. The first one can be efficiently implemented on the device using look-up-tables but it is still not supported by the modern mass-market mobile chips. The latter cannot be implemented for efficient inference on edge devices as it requires the decoding step before calculation, e.g., matrix multiplication, that leads to a significant computational overhead.

3 Related Works

Super-Resolution Neural Networks Super Resolution task takes its roots from the compressed sensing theory that proves that it is possible to restore the signal with sampling under the Nyquist frequency under several conditions. In general, SR means that using a low resolution (LR) image we need to produce a high resolution (HR) image restoring the details that were lost after some degradation (downsampling, blurring, motion, etc.). First application of convolutional NN to SR task was made by C. Dong and his colleagues in [9] outperforming other methods at that moment. Further improvements in SR were made with more and more computationally expensive CNNs. [18, 33].

Neural Network Quantization Quantization of neural networks can be done in two ways: using post-training quantization (PTQ) or quantization aware training (QAT). The former considers that we have only an FP model and some

unlabeled data. This method is fast and provides close-to-FP results with several techniques like AdaRound [23], Cross Layer Equalization [24], knowledge distillation [5], etc. QAT method uses straight-through estimator [3] for approximating the gradients of the round-to-nearest operator. This method requires labeled data and training pipeline for application and achieves better results than PTQ generally, but it is also usually time-consuming to make the quantized model converge. CNN quantization unavoidably leads to a quality drop of different amounts: the smaller bitwidth is used, the more significant is the quality drop. Also, the quality drop depends on the task complexity, e.g. for low-level vision tasks ternary and binary networks were proposed that reached performance close to FP models [8, 21, 34]. In contrast, low-level vision tasks are still challenging for low-bit quantization [13].

Quantized Super-Resolution Models Lightweight and efficient solutions evolved in parallel to reduce the computational overhead while preserving the performance. Mobile chips development also pushed researchers to find lightweight and effective solutions starting from the simplest one [27] to the modern state-of-the-art real time solutions [1, 11, 29]. Although these solutions are effective, they still cannot be deployed on terminal devices for real applications without quantization. Numerous competitions in mobile SR proves great importance of porting SR solutions to edge devices [7, 15, 17]. In these competitions, quantization is a general way for reducing inference time but it also adds more challenges. There are several papers dedicated to SR CNN quantization. In [14], authors proposed to consider the channel-wise distribution of intermediate feature maps to perform quantization more precisely according to the probability density functions. In [13], authors proposed to allocate appropriate bitwidth for different image regions to reduce quantization error for internal feature maps. These methods are beneficial for SR CNN quantization but the former requires significant overhead and memory usage, while mixed precision computations in the latter are less effective than equal bitwidth counterparts due to casting operations. Moreover, in all aforementioned papers the first and last convolutions are not quantized, which in turn does not allow to achieve the lower bound of inference time. In this work, we focus on full quantization of SR networks including the first and last layers into 4 bit with 8-bit input images to achieve significant acceleration while preserving almost the same performance as FP models.

4 Motivation

Natural images comprise a class of 2D discrete signals with certain properties. There are some empirical observations on natural images that we employ in this paper. First, an image is a partially smooth 2D signal with a finite number of discontinuities of the first kind. Second, the number of discontinuities is much smaller than the total number of pixels. Using these facts we want to hide quantization error into this small number of edge pixels, while lowering it for the majority of other pixels. Overall, quantization error will stay the same but we expect that "hiding" it in edges and texture will provide more appealing results.

To get edge pixels from the image, we can employ well-known filters for edge detection, such as gradient and Laplacian. In the worst case, errors of SR

networks in such a domain will affect only a small number of pixels on the output image. As we consider application of finite-difference approximations for differential operators (gradient and Laplacian are approximations of the first and second order derivatives, respectively), we utilize solvers for partial differential equations (PDEs) to restore the output of the network back to the original domain. We investigated several types of solvers: based only on the boundary pixels and its regularized version. With the latter we achieved impressive results for full 4-bit quantization of different SR networks, which we describe in the next sections.

To show the efficacy of the proposed approach let us consider a toy example of smooth 8-bit image Fig. 3a that we will quantize into 4 bit. The 4 bit result (Fig. 3b) suffers from rounding-to-nearest error, which causes step-like artifacts. But if we quantize the Laplacian of the original image and then restore it back using PDE solver, we will obtain the result shown in Fig. 3c. This toy example proves that the Laplacian domain, in particular, with consequent reconstruction is more robust to quantization errors.

(a) (b) (c)

Fig. 3: Quantization of original 8bit smooth image *(a)* to 4 bit *(b)* and restored image from 4-bit Laplacian *(c)*.

5 Proposed Approach

In this section, we present the proposed approach for application of differential operators to the quantized SR NNs. First, we give the pipeline overview and then the detailed description of its parts.

5.1 Pipeline Overview

In a general case, the application of quantized SR NNs is straightforward: an input low-resolution (LR) image is passed through the quantized network, which outputs the high-resolution (HR) image. The pipeline of the proposed method is presented in Fig. 4 and consists of the following parts:

1. **Differential Operator:** An input image first passes through a differential operator that extracts edges and texture from it.
2. **Quantized SR NN**: The obtained LR image in the operator domain passes through the quantized SR NN to obtain an enhanced version of the image **in the operator domain**.
3. **PDE solver**: A PDE solver is used to restore the obtained enhanced image from the DO to the original domain. Dirichlet boundary conditions from the initial LR image are used to guarantee a unique solution. The boundaries are upsampled using a bilinear upsample since the input LR image has a smaller size than the output image. As will be shown later, we utilize regularization

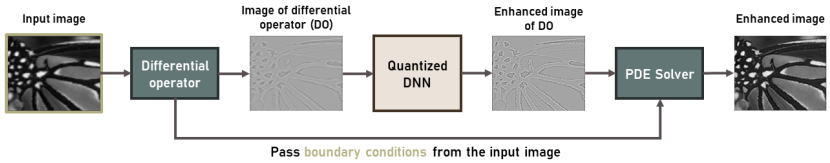

Fig. 4: The proposed pipeline for quantized SR CNNs.

term to make the solver more robust to quantization errors in the operator domain.

The key idea of the proposed approach is that such a sparse input representation allows the capacity of the network to be utilized only for a small amount of information extracted from the whole image, which makes it easier to quantize. However, the neural network is not capable to restore the image from its operator image by itself. Therefore, we utilize an additional module, a PDE solver, which allows to reconstruct the enhanced image in the original domain from the enhanced operator image (the network output) and initial conditions from an input LR image. It is worth noting that the backbone of the original CNN is not modified, all layers (including the first and the last) are uniformly quantized, and only the differential operator and PDE solver blocks are added to it. In the next sections, we describe each of these blocks in more detail.

5.2 Differential Operator

The first and higher order derivatives can be represented using finite difference approximations for discrete signals (such as images) differentiation. Without loss of the generality, let us consider a simple 1D case for the well-known Laplace equation:

$$\nabla^2 f(x) = 0, \tag{4}$$

where $f(x)$ is a 1D twice differentiable real-valued function. In the Cartesian coordinate system, this can be written in the following form:

$$\nabla^2 f(x) = \frac{\partial^2 f}{\partial x^2} = 0. \tag{5}$$

To find a unique solution to Laplace's equation, we need to specify boundary conditions. Considering central finite difference approximation of the second order we can rewrite second order derivative in the following matrix form:

$$\frac{\partial^2 f}{\partial x^2} = \begin{bmatrix} -1 & 2 & -1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & -1 & 2 & -1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \tag{6}$$

where y_i are discrete values of function f.

In order to achieve efficient implementation on edge devices, instead of matrix multiplication we utilize convolution with a 2D finite difference operator kernel approximation in order to transform the input image into the operator domain. Operator application can also be implemented via filtering in the frequency domain, but ordinary convolution requires less time for small kernel sizes. It is also worth noting that the operator kernel can be easily made trainable since the operator is applied to the input image via convolution.

5.3 PDE Solver

To perform the inverse transformation from the operator domain we utilize a PDE solver. There are several ways how it can be implemented:

1. Using linear algebra to solve a system of linear equations;
2. Using Green's function as a solution to a linear differential equation (similar to inverse filtering);
3. Using a PDE solver with regularization.

We will not consider the first approach in this paper, as solving matrix equation 6 is not computationally efficient on terminal devices since this process is iterative. The second approach is described in detail in the supplemental information. The significant disadvantages of this approach, as we show in ablation study, are its low robustness to errors occurring in the operator domain and zeros in the denominator (zeros in Fourier coefficients of the operator). To overcome the first problem, we use patch-wise processing with a large overlap relative to the patch size, while for the second problem, we use a special case of Tikhonov regularization by adding a small constant in the denominator. However, the patch-wise processing with such a large overlap leads to a significant computational overhead which decreases the benefits of full 4-bit quantization.

To overcome the aforementioned issues, we regularize the PDE solver with the points from the initial domain. This problem can be formulated as follows:

$$\underset{y}{argmin}\Big[\|\mathcal{L}y - \hat{\mathcal{L}}y\|_2^2 + \lambda \cdot \|y - \hat{y}\|_2^2\Big], \tag{7}$$

where $\mathcal{L}y$ is the operator image of image y we want to restore. $\hat{\mathcal{L}}y$ is an estimate of the operator image from the quantized SR network, $\hat{y}$ is the estimate in the initial domain. Parameter λ is a hyper-parameter that is responsible for the trade-off between estimates in the initial and operator domains. By using the Parseval's theorem on preserving the L_2-norm, we can obtain the following analytical solution of this minimization problem:

$$\mathcal{F}(f) = \frac{\mathcal{F}(\hat{y}) + \lambda \cdot \overline{\mathcal{F}(K)} \cdot \mathcal{F}(\hat{\mathcal{L}}y)}{1 + \lambda \cdot |\mathcal{F}(K)|^2}, \tag{8}$$

where $\overline{(\cdot)}$ denotes the complex conjugate.

The regularized PDE solver is more computationally expensive than inverse filtering but it is robust to errors in the boundary conditions and errors in the operator domain. Moreover, the denominator is always positive if non-negative λ is used, and therefore this solver is not ill-posed in contrast with the inverse filtering formulation.

6 Experiment Results

In this section, we provide results for different SR networks to show the benefits of the proposed approach for low-bit quantization. We employ SRCNN, ESPCN, EDSR, and RFDN to demonstrate the overarching benefit of our approach for SR tasks under ***full*** low-bit quantization. In the ablation subsection, we compare our approach with different solvers and check its robustness to different bitwidth. In the main results subsection, we present results of the proposed method on different SR architectures in comparison with regular LSQ+ [4] quantization. Finally, we provide comparison with DAQ [14] and CADYQ [13] SOTA methods, and show that our approach achieves current state-of-the-art results for full 4-bit quantization on EDSR network. Generally, the suggested approach is orthogonal to these methods and can be combined together.

6.1 Experimental Setup

For our experiments, we used the open-source code for SRCNN [1], ESPCN [2], EDSR [3], RFDN [4] and used our method as a wrapper for each of these models. We also found that using PyTorch bilinear upsample results in shifts in the resulting images that do not allow to achieve optimal performance. To overcome this issue, we used the ResizeRight package [5]. For EDSR, we used mean subtraction and addition outside the main pipeline, as mean RGB values were calculated for DIV2K images in the original domain. All other training details (such as learning rate, number of epochs, etc.) were used as is, please refer to the repositories mentioned above. For our experiments we initialized kernels with: Dirac delta (kernel that does not change the input image), sum of gradients, Laplacian, sum of Sobel horizontal and vertical filters. We tried these filters both fixed and trainable and due to space limitations only the best results are presented.

Finally, we measured theoretical complexity of the proposed approach and other methods using BitOps [2]. BitOps for an arbitrary layer is calculated in the following way:

$$BitOps = FLOPS \cdot \frac{w\_bit}{32} \cdot \frac{a\_bit}{32},$$

where $FLOPS$ is floating point operations, w_bit and a_bit are bitwidths of weights and activations, respectively.

6.2 Ablation Study

Inverse Filtering vs. Regularized PDE Solver Our first attempts were made using Inverse Filtering as a PDE solver, but this approach has the following drawbacks. First, using operator kernels makes inverse filtering an ill-posed

[1] https://github.com/yjn870/SRCNN-pytorch
[2] https://github.com/yjn870/ESPCN-pytorch
[3] https://github.com/sanghyun-son/EDSR-PyTorch
[4] https://github.com/njulj/RFDN
[5] https://github.com/assafshocher/ResizeRight

problem, so we added a small constant to the denominator to overcome this (special case of Tikhonov regularization). Additionally, we faced with high instability during the inference stage, as errors in the operator domain produced errors in the original domain that were propagated throughout the whole image. Therefore, for inverse filtering solver, we used a small trick by dividing the image into small patches of size 64-by-64 with overlap of 16.

To overcome these drawbacks, we developed a regularized PDE solver (Eq. (8)) that has additional computational overhead but is no longer ill-posed allowing to use trainable operator kernels and images of different size without patch-wise processing. In Tab. 1, we present results on different NNs with inverse filtering and regularized PDE solver. From the obtained results we can conclude that the regularized PDE solver outperforms inverse filtering and does not have the disadvantages mentioned above. The drawback of the regularized approach is additional computational overhead, which is discussed in Sec. 6.5.

Table 1: Comparison between inverse filtering and regularized PDE solvers for SRCNN x3 on different test sets (PSNR, dB / SSIM / LPIPS). **Bold** values denotes the best results.

Test set	IF PDE solver	Reg. PDE Solver
Set5	32.27 / 0.952 / 0.157	**32.65** / **0.956** / **0.143**
Set14	28.62 / 0.893 / 0.267	**28.86** / **0.897** / **0.249**
Urban100	26.66 / 0.880 / 0.240	**26.96** / **0.886** / **0.213**
BSDS300	28.59 / 0.878 / 0.304	**28.81** / **0.881** / **0.281**

Robustness of the Proposed Approach to Different Bitwidth We checked our approach on EDSR with scale x2 using different bitwidths ranging from 2 to 8 for both weights and activations. The PSNR curves for the proposed approach and regular LSQ+ quantization are shown in Fig. 5. For reference, we also provide results for the FP model and bicubic upsampling. From this figure, we can conclude that the proposed approach significantly outperforms regular quantization for all bitwidths. Moreover, our method is still better than bicubic upsampling even for quantization into full 2-bit quantization, while regular quantization produces results worse than bicubic upsampling at this bitwidth.

6.3 Main Results

In Tab. 2 we present the best results achieved with the proposed method in FP and quantization scenarios. We present results for fully quantized networks with weights and activations quantized in 4-bits (denoted as W4A4). Visual effect of the proposed approach is presented in Fig. 6. For more examples please refer to supplemental materials. From visual and quantitative results we observe that the proposed approach significantly improves SR network performance. We observe that most problems with regular quantization pipeline occur on smooth areas, while the proposed approach does not suffer from it. Moreover, details with our approach are also improved as the network does not spend its capacity to overcome false contours from quantization.

Table 2: Results for different SR networks with full 4-bit quantization using regular quantization pipeline and our method (PSNR, dB / SSIM).

CNN			FP	W4A4	
			Regular	Regular	Our method
SRCNN	x2	Set5	37.36/0.9842	34.23/0.9653	**37.18/0.9836**
		Set14	32.34/0.9491	30.54/0.9243	**32.12/0.9482**
		Urban100	29.45/0.9347	27.82/0.9074	**29.14/0.9317**
		B100	33.09/0.9514	30.87/0.9225	**32.94/0.9501**
	x3	Set5	31.33/0.9382	32.63/0.9550	**32.76/0.9573**
		Set14	28.13/0.8759	28.91/0.8973	**28.99/0.8989**
		Urban100	26.23/0.8652	27.02/0.8867	**27.12/0.8882**
		B100	27.95/0.8568	28.82/0.8811	**28.90/0.8833**
	x4	Set5	30.74/0.9575	29.73/0.9183	**30.63/0.9348**
		Set14	27.53/0.8985	26.87/0.8421	**27.36/0.8601**
		Urban100	24.41/0.8881	23.87/0.7919	**24.27/0.8086**
		B100	27.68/0.8823	26.96/0.8221	**27.60/0.8420**
ESPCN	x3	Set5	32.27/0.9519	30.05/0.9203	**31.88/0.9462**
		Set14	28.72/0.8942	27.35/0.8607	**28.54/0.8883**
		Urban100	26.58/0.8766	25.54/0.8480	**26.53/0.8747**
		B100	28.63/0.8782	27.29/0.8423	**28.48/0.8733**
EDSR	x2	Set5	38.58/0.9879	34.75/0.9698	**38.08/0.9852**
		Set14	33.77/0.9550	31.34/0.9372	**33.37/0.9536**
		Urban100	31.99/0.9557	29.17/0.9309	**31.14/0.9495**
		B100	34.01/0.9573	32.06/0.9411	**33.74/0.9557**
		DIV2K	34.57/0.9711	30.23/0.9502	**33.98/0.9687**
	x3	Set5	34.41/0.9668	32.18/0.9489	**34.00/0.9641**
		Set14	30.37/0.9112	28.92/0.8906	**30.05/0.9075**
		Urban100	29.31/0.9218	27.43/0.8931	**28.73/0.9140**
		B100	29.63/0.8943	28.64/0.8752	**29.43/0.8904**
		DIV2K	30.91/0.9389	28.39/0.9155	**30.49/0.9332**
	x4	Set5	32.71/0.9554	31.00/0.9373	**32.22/0.9519**
		Set14	29.05/0.8811	27.83/0.8618	**28.75/0.8772**
		Urban100	26.04/0.8582	24.81/0.8261	**25.62/0.8483**
		B100	28.48/0.8615	27.63/0.8418	**28.27/0.8562**
		DIV2K	28.94/0.9071	27.10/0.8877	**28.57/0.9018**
RFDN	x2	Set5	38.35/0.9861	34.72/0.9721	**37.85/0.9846**
		Set14	33.54/0.9543	31.06/0.9380	**33.15/0.9519**
		Urban100	31.55/0.9515	28.92/0.9302	**30.83/0.9469**
		B100	33.84/0.9563	31.83/0.9411	**33.57/0.9545**
		DIV2K	34.33/0.9699	30.09/0.9526	**33.76/0.9674**

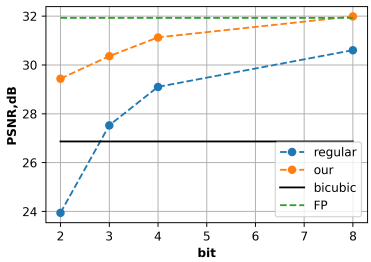

Fig. 5: PSNR for the proposed approach and regular quantization of EDSR x2 using different bitwidths. Results for the FP model and bicubic upsampling are also shown for reference.

Table 3: Comparison between DAQ, CADYQ, regular quantization and the proposed method in case of full 4-bit quantization for EDSR x2 on Urban100 benchmark. *Please note that the average feature quantization rate (FQR, see [13] for details) was 7.33 and this value was also used for BOps estimation.

Method	BOps, G	Latency, ms	PSNR	SSIM
Regular	5.63	12.69	29.17	0.9309
DAQ	12.56	N/A	27.13	0.8103
CADYQ*	24.76	N/A	30.84	0.9132
Our	6.41 (6.01)	13.52	**31.14**	**0.9495**

6.4 Comparison with Other Methods

Additionally, we compare the proposed method with DAQ [14] and CADYQ [13] (Tab. 3) for the case of full 4-bit quantization. It is worth noting that CADYQ uses a set of bitwidths to allocate the appropriate number of bits for different regions of the input image and internal feature maps. In case when we limit this set to 4 bit only, the method becomes LSQ+ (denoted as Regular W4A4). From these results, it can be seen that the proposed method outperforms the regular approach and the considered SOTA methods. We adopted BitOps [2] as a measure of complexity as it is not hardware specific and allows to estimate the most expensive computations (multiply-accumulate) with respect to the bitwidth of corresponding operations. In our calculation we made BOps calculations as is, not considering ways to accelerate them. For optimization, we can reuse $\overline{\mathcal{F}(K)}$ and use the symmetry property for real-valued signals. Moreover, terms $\lambda \cdot \overline{\mathcal{F}(K)}$ and $1 + \lambda \cdot |\mathcal{F}(K)|^2$ can be pre-computed once without further impact on the inference time (please see updated BOPps with optimizations above in Tab. 3, this value is provided in brackets). With such an optimization, BOps are significantly reduced. Additionally we performed latency measurements on Nvidia A10G GPU. We used the following setup: input image 512x512, float16 evaluation of pre- and post-processing, w8a8 quantization for all convolutions with TensorRT compilation. We have approximated w4a4 inference time based on

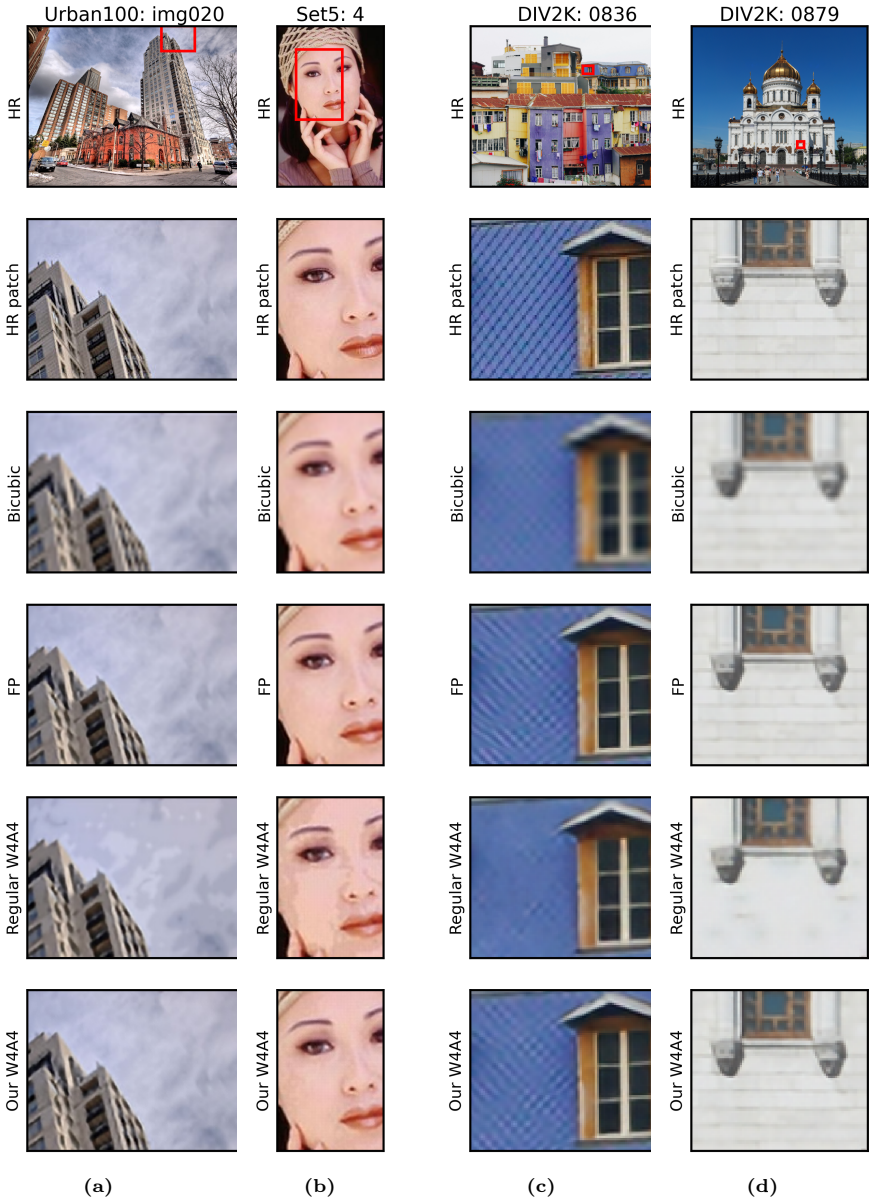

Fig. 6: Image examples for (a) SRCNN x3, (b) ESPCN x3, (c) EDSR x3, (d) RFDN x2.

Nvidia report on cutlass experimental w4a4 API, as it is currently not officially supported in TensorRT [6].

6.5 Computational Complexity

Introducing additional parts into the pipeline unavoidably leads to the additional computational overhead. Thus, we checked the BitOps for different networks with and without proposed blocks (Fig. 4) for input image of size 512 by 512.

From theoretical calculations (Tab. 4) we can conclude that the proposed method adds substantial computational overhead as operator and solver are calculated in FP using Fast Fourier Transform. In order to neglect this overhead on terminal device, it is possible to apply both operator and solver on CPU, DSP or GPU while the quantized network could be inferenced on the NPU.

Table 4: Computational overhead in BitOps (G) for different SR networks using the proposed approach.

Model	Scale	FP Regular	W4A4 Regular	W4A4 Our
SRCNN	x2, x3, x4	14.99	0.23	0.26
ESPCN	x2	5.55	0.08	0.16
	x3	5.93	0.09	0.26
	x4	6.46	0.10	0.40
EDSR	x2	360.12	5.63	6.41
	x3	410.71	6.42	7.50
	x4	520.20	8.13	9.62
RFDN	x2	140.63	2.20	2.77
	x3	142.33	2.22	3.09
	x4	144.71	2.26	3.54

7 Conclusions

This paper is dedicated to the problem of full 4-bit quantization of SR neural networks. We utilize essential properties of natural images to overcome significant performance degradation of low-bit quantization. We propose to redistribute the quantization error by hiding it in edges and texture of the original image. To extract this information we apply finite-difference approximation of differential operators and for restoration from the operator domain we utilize the regularized PDE solver. Our result on different SR networks show a significant improvement in performance in comparison with regular quantization pipeline in the case of full 4-bit quantization. According to the BitOps measurements we can conclude that computational overhead is insignificant in comparison with full 4-bit quantization acceleration.

[6] https://developer.nvidia.com/blog/int4-for-ai-inference/

References

1. Ayazoglu, M.: Extremely lightweight quantization robust real-time single-image super resolution for mobile devices. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. pp. 2472–2479 (June 2021) 5
2. Baskin, C., Liss, N., Schwartz, E., Zheltonozhskii, E., Giryes, R., Bronstein, A.M., Mendelson, A.: Uniq: Uniform noise injection for non-uniform quantization of neural networks. ACM Transactions on Computer Systems **37**(1–4), 1–15 (Nov 2019). https://doi.org/10.1145/3444943, http://dx.doi.org/10.1145/3444943 9, 12
3. Bengio, Y., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation (2013) 5
4. Bhalgat, Y., Lee, J., Nagel, M., Blankevoort, T., Kwak, N.: Lsq+: Improving low-bit quantization through learnable offsets and better initialization (2020), https://arxiv.org/abs/2004.09576 9
5. Cho, J.H., Hariharan, B.: On the efficacy of knowledge distillation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4794–4802 (2019) 5
6. Choi, Y., El-Khamy, M., Lee, J.: Towards the limit of network quantization (2017) 2, 4
7. Conde, M.V., Zamfir, E., Timofte, R., Motilla, D., Liu, C., Zhang, Z., Peng, Y., Lin, Y., Guo, J., Zou, X., et al.: Efficient deep models for real-time 4k image super-resolution. ntire 2023 benchmark and report. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1495–1521 (2023) 5
8. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1 (2016) 2, 5
9. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks (2015) 1, 4
10. Faraone, J., Fraser, N., Blott, M., Leong, P.H.W.: Syq: Learning symmetric quantization for efficient deep neural networks (2018) 4
11. Gao, D., Zhou, D.: A very lightweight and efficient image super-resolution network. Expert Systems with Applications **213**, 118898 (2023) 5
12. Gong, Y., Liu, L., Yang, M., Bourdev, L.: Compressing deep convolutional networks using vector quantization (2014) 4
13. Hong, C., Baik, S., Kim, H., Nah, S., Lee, K.M.: Cadyq: Content-aware dynamic quantization for image super-resolution. In: European Conference on Computer Vision. pp. 367–383. Springer (2022) 2, 5, 9, 12
14. Hong, C., Kim, H., Baik, S., Oh, J., Lee, K.M.: Daq: Channel-wise distribution-aware quantization for deep image super-resolution networks. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2675–2684 (2022) 2, 5, 9, 12
15. Ignatov, A., Romero, A., Kim, H., Timofte, R.: Real-time video super-resolution on smartphones with deep learning, mobile ai 2021 challenge: Report. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. pp. 2535–2544 (June 2021) 5
16. Jung, S., Son, C., Lee, S., Son, J., Kwak, Y., Han, J.J., Hwang, S.J., Choi, C.: Learning to quantize deep networks by optimizing quantization intervals with task loss (2018) 4

17. Khan, F.S., Khan, S.: Ntire 2022 challenge on efficient super-resolution: Methods and results (2022) 5
18. Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W.: Photo-realistic single image super-resolution using a generative adversarial network (2017) 1, 4
19. Liao, Z., Couillet, R., Mahoney, M.W.: Sparse quantized spectral clustering (2020) 4
20. Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution (2017) 1
21. Lin, X., Zhao, C., Pan, W.: Towards accurate binary convolutional neural network (2017) 2, 5
22. Liu, Z., Cheng, K.T., Huang, D., Xing, E., Shen, Z.: Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation (2022) 4
23. Nagel, M., Amjad, R.A., Van Baalen, M., Louizos, C., Blankevoort, T.: Up or down? adaptive rounding for post-training quantization. In: International Conference on Machine Learning. pp. 7197–7206. PMLR (2020) 5
24. Nagel, M., Baalen, M.v., Blankevoort, T., Welling, M.: Data-free quantization through weight equalization and bias correction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1325–1334 (2019) 5
25. Nagel, M., Fournarakis, M., Amjad, R.A., Bondarenko, Y., van Baalen, M., Blankevoort, T.: A white paper on neural network quantization (2021) 2, 3
26. Qin, H., Zhang, Y., Ding, Y., liu, Y., Liu, X., Danelljan, M., Yu, F.: Quantsr: Accurate low-bit quantization for efficient image super-resolution. In: Conference on Neural Information Processing Systems (NeurIPS) (2023) 2
27. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network (2016) 1, 5
28. Singh, V., Sharma, A., Devanathan, S., Mittal, A.: High-frequency refinement for sharper video super-resolution. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3299–3308 (2020) 2
29. Song, D., Wang, Y., Chen, H., Xu, C., Xu, C., Tao, D.: Addersr: Towards energy efficient image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 15648–15657 (June 2021) 5
30. Weng, X., Chen, Y., Zheng, Z., Gu, Y., Zhou, J., Zhang, Y.: A high-frequency focused network for lightweight single image super-resolution (2023) 2
31. Xu, Y., Wang, Y., Zhou, A., Lin, W., Xiong, H.: Deep neural network compression with single and multiple level quantization (2018) 2
32. Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., Hua, X.: Quantization networks (2019) 2
33. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks (2018) 1, 4
34. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients (2018) 2, 5