


Explicit Model Size Control and Relaxation via Smooth Regularization for Mixed-Precision Quantization Supplementary Material

Vladimir Chikin^{1*}, Kirill Solodskikh^{1*}, and Irina Zhelavskaya² 

¹ Huawei Noah's Ark Lab

{vladimir.chikin,solodskikh.kirill1}@huawei.com

² Skolkovo Institute of Science and Technology (Skoltech)
irina.zhelavskaya@skolkovotech.ru

A Quantization regularizers as an estimate of the quantization error

Let $\mathbf{x} \in \mathbb{R}^k$ be a vector that we want to quantize, s – the scale parameter (that is scalar), b – the quantization bit-width, and $t = 2^{b-1}$. The mean squared quantization error is defined as follows:

$$\begin{aligned} \text{MSQE}(\mathbf{x}, s, t) &= \frac{1}{k} \left\| \mathbf{x} - \frac{s}{t} \mathcal{Q}_U^b \left(\frac{t \cdot \mathbf{x}}{s} \right) \right\|_2^2 = \\ &= \frac{s^2}{t^2} \frac{1}{k} \left\| \frac{t \cdot \mathbf{x}}{s} - \mathcal{Q}_U^b \left(\frac{t \cdot \mathbf{x}}{s} \right) \right\|_2^2. \end{aligned} \tag{A.1}$$

In this case, uniform quantization function \mathcal{Q}_U^b is applied element-wise. We will denote $\text{MSQE}(\mathbf{x}, 1, 1)$ by $\text{MSQE}(\mathbf{x})$. Note that the following equality holds: $\text{MSQE}(\mathbf{x}, s, t) = \frac{s^2}{t^2} \text{MSQE}(\frac{t \cdot \mathbf{x}}{s})$.

The proposed regularizers have the following property: for small quantization errors, the values of regularizers can be used as an estimate of the quantization error. Based on this property, we propose to use the proposed regularizer ϕ multiplied by $\frac{s^2}{t^2}$ to train quantized models. We assume that the second derivative $\phi''(x, t)$ is continuous in the neighborhood of integer points from the range $[-\lfloor t \rfloor, \lfloor t \rfloor - 1]$. It is easy to verify that this requirement is satisfied for all integer points of this range for the regularizer we use, except for the extreme integer points when t is integer.

Proposition 1. *For each regularizer ϕ from the proposed class of regularizers, in the neighborhood of each integer point from the range $[-\lfloor t \rfloor, \lfloor t \rfloor - 1]$, there*

* These authors contributed equally to this work.

exists a constant C , such that

$$\frac{s^2}{t^2} \phi\left(\frac{t \cdot \mathbf{x}}{s}, t\right) = C \cdot \text{MSQE}(\mathbf{x}, s, t) + o(\text{MSQE}(\mathbf{x}, s, t)),$$

for $\text{MSQE}(\mathbf{x}, s, t) \rightarrow 0$. (A.2)

Proof. The value of $\text{MSQE}(\mathbf{x}, s, t)$ is zero only when the value of $\frac{t \cdot \mathbf{x}}{s}$ falls into the integer points from $[-\lfloor t \rfloor, \lfloor t \rfloor - 1]$. Let n be some integer from $[-\lfloor t \rfloor, \lfloor t \rfloor - 1]$. By definition of the proposed regularizers, all integers from this segment are the roots and minima of function $\phi(\cdot, t)$, in particular, $\phi(n, t) = 0$ and $\phi'(n, t) = 0$. Consider the Taylor series of function $\phi(x, t)$ at the point n for a fixed value of t . Since $\phi(x, t)$ is twice differentiable in the neighborhood of n , we have

$$\phi(x, t) = \frac{1}{2} \phi''(n, t) (x - n)^2 + o((x - n)^2), \quad x \rightarrow n. \quad (\text{A.3})$$

But $\text{MSQE}(x) = (x - n)^2$ for any $x \in [n - \frac{1}{2}, n + \frac{1}{2}]$, and therefore we get that

$$\phi(x, t) = \frac{1}{2} \phi''(n, t) \text{MSQE}(x) + o(\text{MSQE}(x)),$$

$\text{MSQE}(x) \rightarrow 0$. (A.4)

Let us denote $\frac{1}{2} \phi''(n, t)$ by C . In the last equality, replace \mathbf{x} with $\frac{t \cdot \mathbf{x}}{s}$, and multiply this equality by $\frac{s^2}{t^2}$. Considering that function $\phi(\mathbf{x})$ of the vector \mathbf{x} is equal to the average of the outputs of such functions applied to the components of vector \mathbf{x} , as well as that $\text{MSQE}(\mathbf{x}, s, t) = \frac{s^2}{t^2} \text{MSQE}(\frac{t \cdot \mathbf{x}}{s})$, we obtain the required statement. \square

B Pseudocode of the proposed algorithm and quantization of continuous distributions

Algorithm 1 summarizes our training approach for mixed-precision quantization. Note that during minimization of \mathcal{L}_Q relative to its variables, we compute gradients of activation regularizers relative only to scale parameters \mathbf{s}_a . This is possible due to the properties of function $\mathcal{L}_a(\mathbf{s}_a)$ with fixed \mathbf{W} and \mathbf{s}_w and allows for faster convergence of \mathbf{s}_a to the global minimum of $\mathcal{L}_a(\mathbf{s}_a)$. We discuss this in detail below. Also, we compute gradients of the weight regularizers \mathcal{L}_w relative only to scales \mathbf{s}_w and bit-widths θ during some time at the beginning of training. We do this in order to weaken the effect of tuning the weights to bit-widths, which will change during training.

Let ξ be a random variable with density p_ξ , the first and second moments of which are finite. Let us consider the mean squared quantization error for a random variable ξ as a function of the scale parameter s for fixed bit-width and parameter t :

$$\text{MSQE}[\xi](s) = \frac{s^2}{t^2} \int_{\mathbb{R}} \left(\frac{t \cdot x}{s} - \mathcal{Q}_U\left(\frac{t \cdot x}{s}\right) \right)^2 p_\xi(x) dx. \quad (\text{B.5})$$

Algorithm 1

Require: \mathbf{W} , \mathbf{s}_w , \mathbf{s}_a – trainable parameters (model and scale parameters).
Require: θ – trainable bit-width parameters.
Require: λ_w , λ_a – regularization coefficients.
Require: b_{init} – initial bit-width of the quantized layers.
Require: lr , N_{train} , N_{init} – learning rate, epoch size, the number of initialization batches.

- 1: Initialize bit-width parameters θ .
- 2: Initialize \mathbf{s}_w using the current weight tensors.
- 3: Initialize \mathbf{s}_a by sampling statistics evaluating F on N_{init} batches from \mathbf{X} .
- 4: **for** N_{train} times **do**
- 5: Sample a random batch from the training dataset.
- 6: Evaluate quantization loss function \mathcal{L}_Q .
- 7: Evaluate gradient parameters:

$$G_w = \frac{\partial \mathcal{L}_Q}{\partial \mathbf{W}} + \lambda_w \frac{\partial \mathcal{L}_w}{\partial \mathbf{W}}, G_{s_w} = \frac{\partial (\mathcal{L}_Q + \lambda_w \mathcal{L}_w)}{\partial \mathbf{s}_w},$$

$$G_{s_a} = \frac{\partial (\mathcal{L}_Q + \lambda_a \mathcal{L}_a)}{\partial \mathbf{s}_a}, G_\theta = \frac{\partial (\mathcal{L}_Q + \lambda_w \mathcal{L}_w)}{\partial \theta}.$$
- 8: Update parameters by calculating gradients and learning rate lr .
- 9: **end for**
- 10: Validate quantized model \mathcal{F}^q .
- 11: **return** $\mathbf{W}, \mathbf{s}_w, \mathbf{s}_a, \theta$.

Consider the problem of minimizing $\text{MSQE}[\xi](s)$ w.r.t the scale parameter s . We compare $\text{MSQE}[\xi](s)$ to the following function:

$$\phi[\xi](s) = \frac{s^2}{t^2} \int_{\mathbb{R}} \phi\left(\frac{t \cdot x}{s}\right) p_\xi(x) dx, \quad (\text{B.6})$$

where ϕ is the regularizer that we use. We investigated the behavior of these functions for various distributions of ξ . In our investigation, we used distributions that suit well for modeling the weights and activations of neural networks. In particular, we used normal and Laplace distributions, as well as normal and Laplace distributions with the subsequent use of ReLU. Figure B.1 shows the plots of $\phi[\xi](s)$ and $\text{MSQE}[\xi](s)$ for different bit-widths and distributions. Note that we scale $\text{MSQE}[\xi](s)$ using a special scalar coefficient for the correct representation of functions $\text{MSQE}[\xi](s)$ and $\phi[\xi](s)$ on the same plot. It can be seen from Figure B.1 that the optimal value of s for $\text{MSQE}[\xi](s) \rightarrow \min$ is quite close to the optimal value of s for $\phi[\xi](s) \rightarrow \min$. Therefore, we can conclude that $\phi[\xi](s)$ is a good estimate of $\text{MSQE}[\xi](s)$. Thus, for a more thorough tuning of the activation scale parameters, we propose to compute the gradients of the activation regularizers only with respect to scale factors s_a during minimization of the loss with the proposed quantization regularizers.

C Training protocols for the experiments

We have performed all experiments using PyTorch 1.9.1, Torchvision 0.10.1, and GPU: NVIDIA Tesla V100 32GB.

Image classification: ResNet-20 on CIFAR-10 For the ablation study in section 5.1 of the paper, we quantize the weights of all layers and do not quantize activations. Quantized models are trained for 100 epochs, using SGD with momentum 0.9 and a learning rate schedule starting with 0.01 and reducing it by a factor of 10 every 25 epochs.

In the experiments in section 5.2, the quantized models are trained for 200 epochs, using SGD with momentum 0.9 and a learning rate schedule starting with 0.01 and reducing it by a factor of 10 after 120 and 160 epochs.

In all these experiments, we use $\lambda_w = 0.01$ and $\lambda_a = 0.01$. We collect statistics for the initialization of activation scale parameters during 50 batches of size 100 and start computing the gradients of regularizers w.r.t. weights after 20 epoch. We quantize the weights of the first and last layers to 4 bits, and the weights of the remaining layers to mixed-precision. In the case of quantization without using the proposed regularizers, parameter b_{init} is equal to 1.7. In the case of quantization using the proposed regularizers, parameter b_{init} is equal to 1.9 for the weights-only quantization, and to 1.8 when activations are quantized to 4 bits. Weight decay is 10^{-8} .

Image classification: ResNet-18, ResNet-50, and MobileNet-V2 on Imagenet In these experiments, the models quantized to near W8A8 are trained for 15 epochs, using SGD with momentum 0.9 and a learning rate schedule starting with 10^{-4} and reducing it to 10^{-5} after 1500 batches. The models quantized to near W4A4 are trained for 32 and 99 epochs for ResNet-18 and ResNet-50 respectively. Learning rate is not reduced for ResNet-18 in this case and reduced after 20000 batches for ResNet-50.

For near W8A8, we quantize activations of all layers of the models to 8 bits and use the proposed quantization regularizers. Initially, we set $\lambda_w = 0$ and $\lambda_a = 10^{-4}$. We collect statistics for the initialization of activation scale parameters during 100 batches (the batch size differs depending on the model and is noted below). After 20000 batches, we start computing the gradients of regularizers w.r.t. weights. We increase λ_w to 1 after 40000 batches, and after 80000 batches we increase it to 10. Weight decay is 10^{-8} . Parameter b_{init} is equal to 7.2. We quantize weights of the first and last layers of all models to 8 bits, and weights of the remaining layers are quantized to mixed-precision. In the case of ResNet-18, the batch size is 64, for ResNet-50 the batch size is 26, and for MobileNet-V2 the batch size is 46.

For near W4A4, we quantize activations of all layers of the models to 4 bits (except for the first and last layers, which are quantized to 8 bits) and use the proposed quantization regularizers. Initially, we set $\lambda_w = 0$ and $\lambda_a = 0.1$. We collect statistics for the initialization of activation scale parameters during 100 batches (the batch size differs depending on the model and is noted below). After 2000 batches, we start computing the gradients of regularizers w.r.t. weights. We increase λ_w to 1 after 40000 batches, and after 80000 batches we increase it to 10. Weight decay is 10^{-8} . Parameter b_{init} is equal to 3.5. We quantize weights of the first and last layers of all models to 8 bits, and weights of the remaining

layers are quantized to mixed-precision. In the case of ResNet-18, the batch size is 32 and for ResNet-50 the batch size is 16.

Image super-resolution: ESPCNN on Vimeo-90K All super-resolution networks are trained on the Vimeo-90K dataset by randomly picking the 126×126 crops from a frame. As a downsampling operator we use a Gaussian blur followed by a bicubic downsampling. We use Adam optimizer with batches of size 32. Single epoch is 128 batches. We use L_1 loss for training the models.

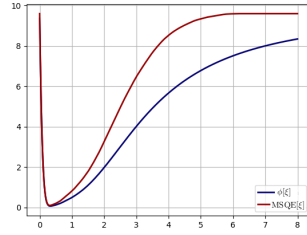
In the case of MixPr1 model, parameter b_{init} is equal to 7.31, and in the case of MixPr2 model, it is equal to 5.94. We start regularizing the bit-widths using a special regularizer for bit-widths values for a set of 4 and 8 bits from the beginning of training. We collect statistics for the initialization of activation scale parameters over 64 batches. We train these models during 5 epochs using regularization parameters $\lambda_w = 10^3$, $\lambda_a = 0.1$ and learning rate 10^{-3} without computing the gradients of regularizers w.r.t. weights. After that, we start computing the gradients of regularizers w.r.t. weights and train these models during 10 epochs with $\lambda_w = 10$, $\lambda_a = 1$ and learning rate 10^{-4} , and then we train them during 200 epochs using $\lambda_w = 10^3$, $\lambda_a = 1$ and learning rate 10^{-5} . Weight decay is 10^{-6} .

D Additional examples for image super-resolution task

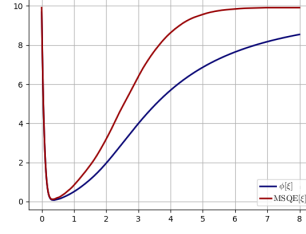
We provide additional examples of images produced by the quantized ESPCNN model in Figures D.1, D.2 and D.3.

E Obtained bit-width distributions

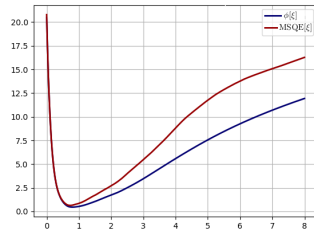
We provide the bit-width distributions obtained by the proposed quantization method for ResNet-20 on CIFAR-10, ResNet-18, ResNet-50, and MobileNet-V2 on ImageNet in Figures E.1-E.8.



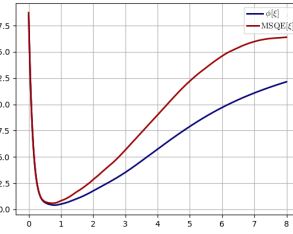
$N(0,1)$, 4-bit scheme.



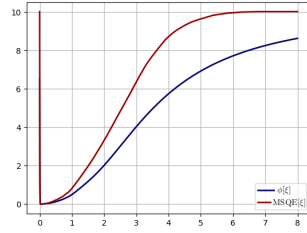
$N(0,1)$ followed by ReLU, 4-bit scheme.



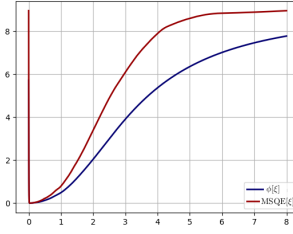
$\text{Laplace}(0,1)$, 4-bit scheme.



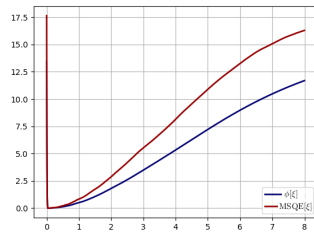
$\text{Laplace}(0,1)$ followed by ReLU, 4-bit scheme.



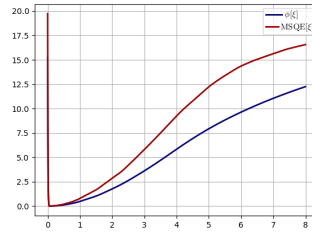
$N(0,1)$, 8-bit scheme.



$N(0,1)$ followed by ReLU, 8-bit scheme.



$\text{Laplace}(0,1)$, 8-bit scheme.



$\text{Laplace}(0,1)$ followed by ReLU, 8-bit scheme.

Fig. B.1: Functions $\phi[\xi](s)$ and $\text{MSQE}[\xi](s)$ for different bit-widths and for normal and Laplace distributions ξ , as well as for normal and Laplace distributions with subsequent use of ReLU.



Fig.D.1: Mixed-precision quantization of the image super resolution task.

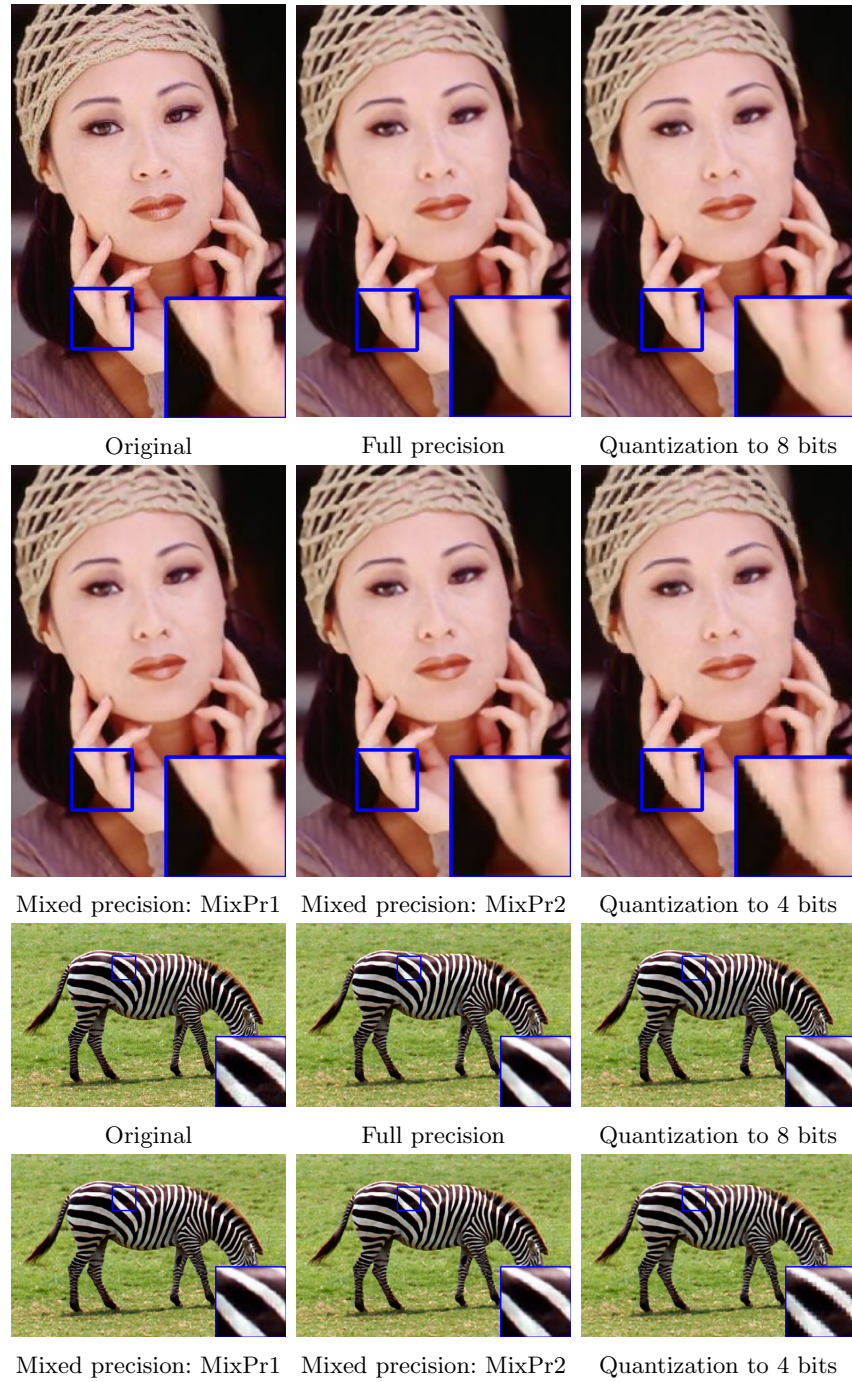


Fig. D.2: Mixed-precision quantization of the image super resolution task.

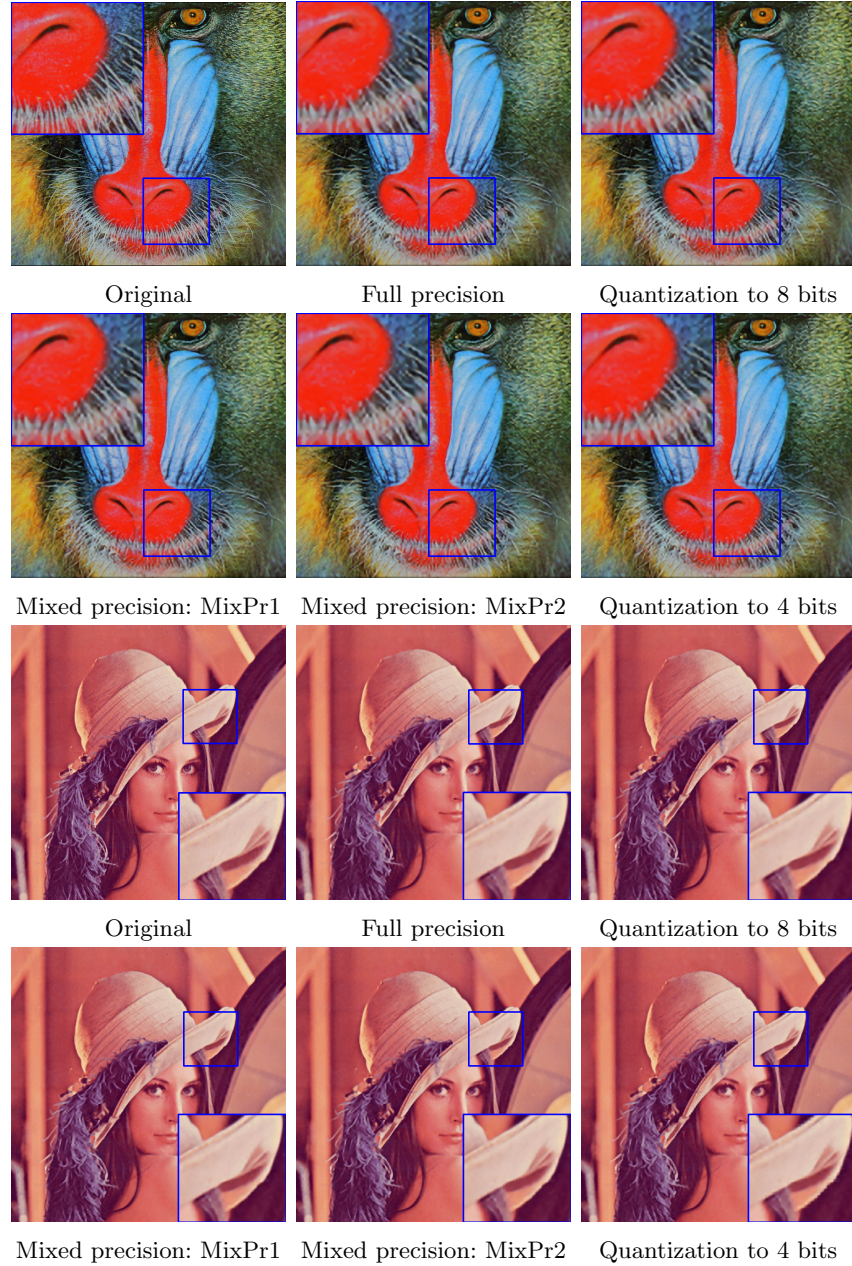


Fig. D.3: Mixed-precision quantization of the image super resolution task.

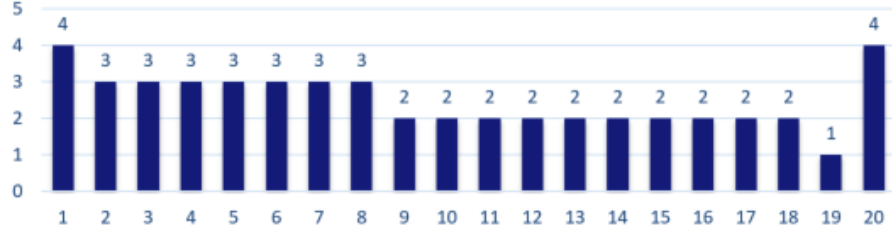


Fig. E.1: Bit-width distribution of quantized ResNet-20 on CIFAR-10 obtained using regularizers without quantization of activations. Accuracy of the obtained model is equal to 91.97%.

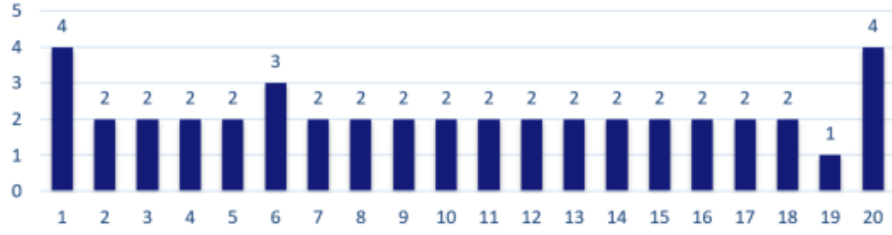


Fig. E.2: Bit-width distribution of quantized ResNet-20 on CIFAR-10 obtained using regularizers with quantization of activations to 4 bits. Accuracy of the obtained model is equal to 91.62%.

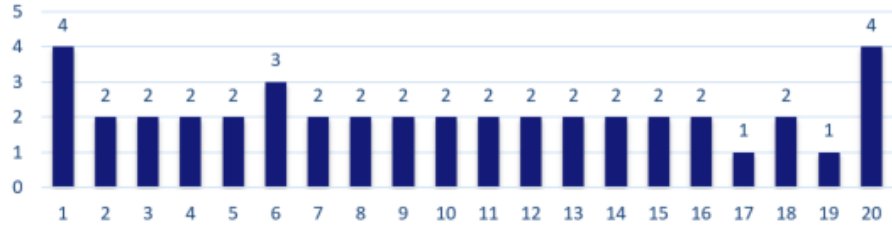


Fig. E.3: Bit-width distribution of quantized ResNet-20 on CIFAR-10 obtained without using regularizers with quantization of activations to 4 bits and without quantization of activations. Accuracy of the obtained model with quantization of activations to 4 bits is equal to 91.55%, and accuracy of the obtained model without quantization of activations is equal to 91.75%.

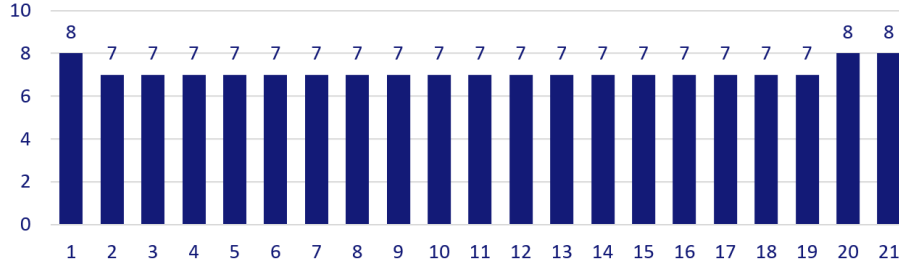


Fig. E.4: Bit-width distribution for quantized ResNet-18 on Imagenet model obtained using regularizers with quantization of activations to 8 bits. Accuracy of the obtained model is equal to 71.81%.

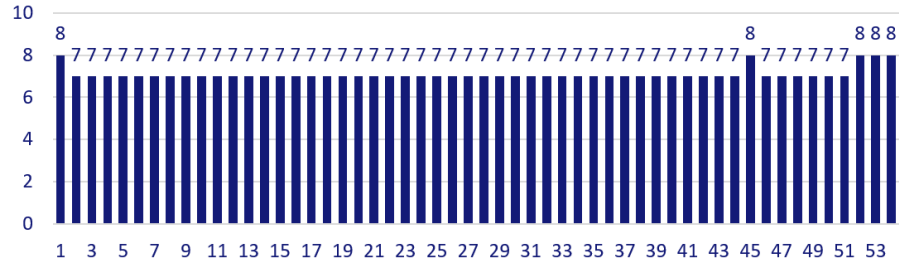


Fig. E.5: Bit-width distribution for quantized ResNet-50 on Imagenet model obtained using regularizers with quantization of activations to 8 bits. Accuracy of the obtained model is equal to 79.45%.

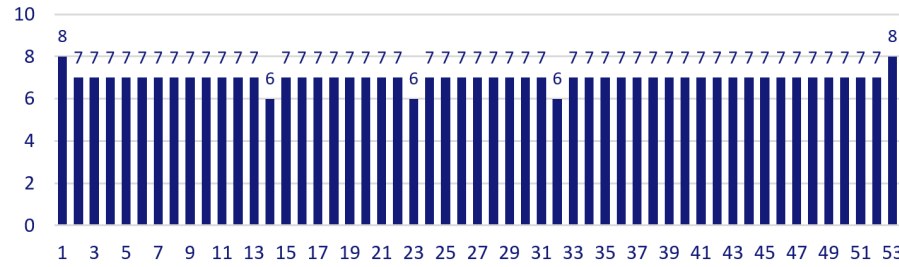


Fig. E.6: Bit-width distribution for quantized MobileNet-v2 on Imagenet model obtained using regularizers with quantization of activations to 8 bits. Accuracy of the obtained model is equal to 71.90%.

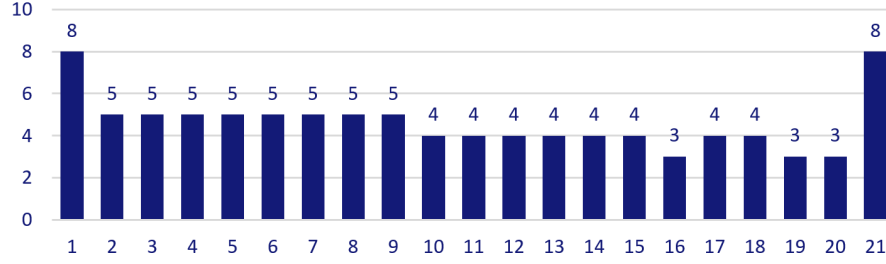


Fig. E.7: Bit-width distribution for quantized ResNet-18 on Imagenet model obtained using regularizers with quantization of activations to 4 bits. Accuracy of the obtained model is equal to 70.68%.

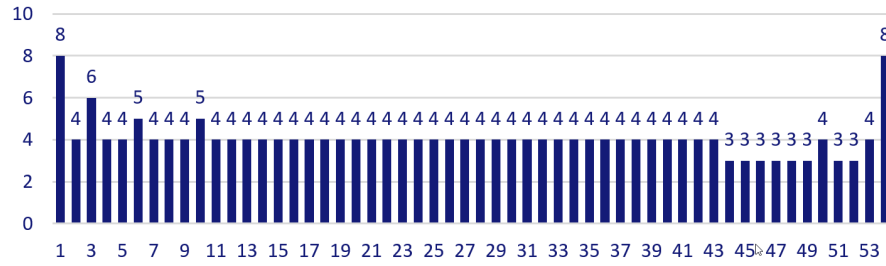


Fig. E.8: Bit-width distribution for quantized ResNet-50 on Imagenet model obtained using regularizers with quantization of activations to 4 bits. Accuracy of the obtained model is equal to 77.45%.