Supplementary Materials for One-Shot Model for Mixed-Precision Quantization

S1. The approximation of the Boltzmann distribution

Let us simplify the hardware term used in evidence lower bound (ELBO) (8). We begin by writing the prior probability (3) given the hardware constraints (12) in terms of bit operations (BOPs),

$$\log p_{k,b} = \log \frac{e^{-\eta b_k^\omega b_k^x \,\mathrm{MACs}(k)}}{\sum_{b\in\mathcal{B}} e^{-\eta b_k^\omega b_k^x \,\mathrm{MACs}(k)}}$$
$$= -\eta b_k^\omega b_k^x \,\mathrm{MACs}(k) - \log \sum_{b\in\mathcal{B}} e^{-\eta b_k^\omega b_k^x \,\mathrm{MACs}(k)}, \quad \text{(S1)}$$

where $\mathcal{B} = \mathcal{B}^\omega \times \mathcal{B}^x$ is a Cartesian product of the available bit width options for weights and activations, and the $*$ symbol is used as a wildcard for $\{\omega, x\}$.

Let us rewrite (S1) in terms of $v = \eta\,\mathrm{MACs}(k)$. Such a notation is convenient because, in practice, v takes small values, usually in the range of $[10^{-5}, 10^{-2}]$. This allows us to express (S1) as a Taylor series around $v = 0$,

$$\log p_{k,b} = -v b_k^\omega b_k^x - \log \sum_{b\in\mathcal{B}} e^{-v b_k^\omega b_k^x} \quad \text{(S2)}$$

$$\log p_{k,b}|_{v=0} = -\log |\mathcal{B}^\omega||\mathcal{B}^x| \quad \text{(S3)}$$

$$\frac{d}{dv}\log p_{k,b}\Big|_{v=0} = -b_k^\omega b_k^x + \frac{\sum_{b\in\mathcal{B}} b_k^\omega b_k^x e^{-v b_k^\omega b_k^x}}{\sum_{b\in\mathcal{B}} e^{-v b_k^\omega b_k^x}}\Big|_{v=0}$$
$$= -b_k^\omega b_k^x + \bar{b}^\omega \bar{b}^x, \quad \text{(S4)}$$

where $\bar{b}^* = |\mathcal{B}^*|^{-1} \sum_{b^*\in\mathcal{B}^*} b^*$ is the average bit widths of weights and activations.

Using (S3) and (S4) in a Taylor expansion of (S2), we get

$$\log p_{k,b} \approx -\log |\mathcal{B}^\omega||\mathcal{B}^x| + (\bar{b}^\omega \bar{b}^x - b_k^\omega b_k^x)\eta\,\mathrm{MACs}(k)$$
$$= -\eta\,\mathrm{MACs}(k) b_k^\omega b_k^x + \eta\,\mathrm{MACs}(k)\bar{b}^\omega \bar{b}^x - \log |\mathcal{B}^\omega||\mathcal{B}^x|. \quad \text{(S5)}$$

The last two terms do not depend on the bit width allocation and can be ignored during optimization. Therefore, we can derive (13) as

$$\mathcal{F}(\omega, \pi) = \mathbb{E}_{z\sim q_\pi(z)}\left[\log p_\omega(\mathcal{D}|z)\right]$$
$$+ \sum_{k=1}^{K}\sum_{b\in\mathcal{B}} \pi_{k,b} \log p_{k,b} + H(\pi) \quad \text{(S6)}$$
$$= \mathbb{E}_{z\sim q_\pi(z)}\left[\log p_\omega(\mathcal{D}|z)\right]$$
$$- \eta \sum_{k=1}^{K} \mathrm{MACs}(k) \sum_{b\in\mathcal{B}} \pi_{k,b} b_k^\omega b_k^x + H(\pi) \quad \text{(S7)}$$
$$= \mathbb{E}_{z\sim q_\pi(z)}\left[\log p_\omega(\mathcal{D}|z)\right]$$
$$- \eta \sum_{k=1}^{K} \tilde{b}_k^\omega \tilde{b}_k^x \,\mathrm{MACs}(k) + H(\pi). \quad \text{(S8)}$$

Note that a similar – but additive w.r.t. $\tilde{b}^\omega$ and $\tilde{b}^x$ – hardware term can be derived for random-access memory (RAM) constraints defined in (S14).

Finally, in Figure S1, we compare the performance of the EdMIPS loss [3] with original hardware constraints formulation and those defined by the Boltzmann distribution (3). As we can see, all studied losses perform similarly. This supports our theoretical approximation of the Boltzmann distribution provided here.

S2. One-Shot MPS algorithm

The proposed One-Shot MPS method is summarized in Algorithm S1. Optional actions can be used for selecting Pareto-optimal architectures w.r.t. the searching dataset. We used optional actions for removing models which rest inside Pareto fronts in Figure 5.

S3. The additional details of the experiments

S3.1. Quantizers

S3.1.1 General notes

We use a uniform quantization grid. For training the quantized networks, we follow the common approach of simulating the quantizing operations. Please, refer to the LSQ [6] article for a practical description of the simulation approach.

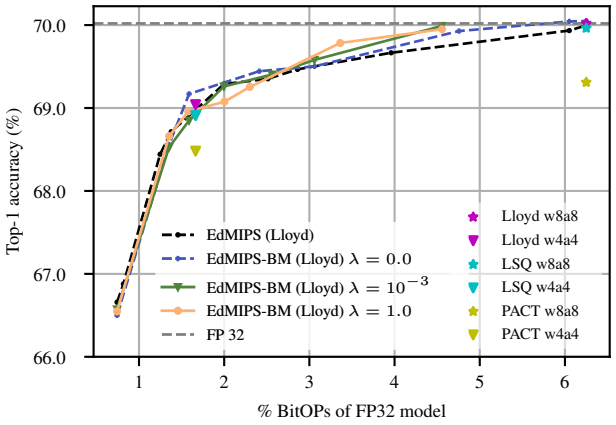


Figure S1. The ResNet-18 Pareto front obtained for EdMIPS with the original hardware constraints formulation [3] ("EdMIPS") and those defined by the Boltzmann distribution (3) ("EdMIPS-BM").

Algorithm S1 One-Shot MPS

Inputs: Pre-trained FP32 model M_ω, range of hardware penalties $[\eta_0, \eta_1]$, searching dataset $\mathcal{D} = \{\mathcal{D}_{train}, \mathcal{D}_{val}\}$
Outputs: Found bit width options B
Transform M_ω to a one-shot supernet $S_{\omega,\theta}$ as shown in Figure 3
while one-shot model is not converged **do**
    Sample $x \sim \mathcal{D}_{train}$ and $\eta \sim \text{LogUniform}(\eta_0, \eta_1)$
    Update $S_{\omega,\theta}$ using a grad step on $\{x, \eta\}$ by minimizing loss (15)
end while
for $i = 0, I$ **do** ▷ Linear (or possibly nonlinear) sweep
    $\eta \leftarrow \eta_0 + i(\eta_1 - \eta_0)/I$
    Select bit width $b_\eta \leftarrow \arg\max f_\theta(\eta)$
    Store $B \leftarrow B \cup b_\eta$
    (optional) Evaluate child model performance $S_{\omega,\theta}(b_\eta)$ on $\mathcal{D}_{val}$
    (optional) Compute hardware metrics for b_η
end for
(optional) Select Pareto front architectures using data from steps 12-13
(optional) Fine-tune the selected architectures

S3.1.2 LSQ and PACT

We use the original LSQ [6] and PACT [4] quantizers.

S3.1.3 Lloyd and Half-Wave Gaussian Quantizer

The Lloyd quantizer [9, 10] uses fixed steps obtained by the Lloyd's algorithm assuming normally distributed tensors. Such an assumption is reasonable for weights [3]. It is also reasonable for activations that follow Batch Normalization (BN) layers. In the case of positive activations due to ReLU non-linearity, we employ the Half-Wave Gaussian Quantizer (HWGQ) [2] with steps also computed using the Lloyd's algorithm. For simplicity, we mention only the Lloyd quantizer in the experiments, meaning that HWGQ is used for positive activations. The Lloyd quantizer has no trainable variables, which makes it easy to set up and use. For this reason, we use it to compare DNAS, EdMIPS, and One-Shot searchers.

The Lloyd quantizer equation for weight tensor $\hat{\omega} \in \mathbb{R}$ is

$$\hat{\omega} = \left\lfloor \text{clip}\left(\frac{\omega}{\text{std}(\omega)s_\omega}, -Q_N, Q_P\right)\right\rceil \text{std}(\omega)s_\omega, \quad \text{(S9)}$$

where integer positive and negative clip values are $Q_N = 2^{b-1}$ and $Q_P = 2^{b-1} - 1$, respectively. The step size s_ω is precomputed as a mean distance of the quantized bin centers. The bin centers are obtained by minimizing the expected distance between the FP32 weights $\omega \in \mathbb{R}$ and their quantized counterpart $\hat{\omega}$ assuming that $\omega \sim \mathcal{N}(0, 1)$. We obtain the following steps for each bit width

$$s_2 = 1.0069, \quad s_4 = 0.3644, \quad s_8 = 0.0365. \quad \text{(S10)}$$

Channel-wise standardization for weight quantizers is applied for SRResNet and MobileNet-v2 because it has been shown that the channels of each kernel have very different ranges of values [8, 16]. Thus, for each kernel in SRResNet and MobileNet-v2, we calculate $\text{std}(\omega)$ in (S9) along its channel dimension.

The quantized input tensor $\hat{x} \in \mathbb{R}$ is calculated as

$$\hat{x} = \left\lfloor \text{clip}\left(\frac{x}{s_x}, -Q_N, Q_P\right)\right\rceil s_x, \quad \text{(S11)}$$

where $x \in \mathbb{R}$, Q_N, and Q_P are defined as in (S9).

We apply HWGQ when the input tensor $x \in \mathbb{R}^+$ is positive, *e.g.* it arrives after the ReLU activation function. In such a case, we use $Q_N = 0$, $Q_P = 2^b - 1$, and steps

$$s_2 = 0.6356, \quad s_4 = 0.2131, \quad s_8 = 0.0203. \quad \text{(S12)}$$

S3.1.4 Trainable Lloyd

As for the trainable step case, we use a Trainable Lloyd quantizer which is a regular Lloyd quantizer with a trainable step. We use the values defined in (S10) and (S12) for initializing the steps. The steps are fine-tuned using the optimizers described in Table S1. In our experience, a Trainable Lloyd quantizer for mixed-precision networks is more stable than LSQ due to its initialization, which does not depend on a chosen batch of data or outliers.

For SRResNet and MobileNet-v2, we also use channel-wise standardization for weight quantizers, *i.e.* $\hat{\omega} = \omega/\text{std}(\omega)$ [8, 16]. Note that the standardization is a hardware-friendly operation that can be fused with quantization steps at inference.

S3.2. Hardware resource equations

The BOPs number for a model is given by

$$BOPs = \sum_k \text{MACs}(k) \cdot b_k^\omega \cdot b_k^x, \tag{S13}$$

where $\text{MACs}(k)$ gives the number of multiply-accumulate operations in layer k, $b^\omega \in \mathcal{B}^\omega$ and $b^x \in \mathcal{B}^x$ are the bit widths of weights and inputs, respectively.

The total RAM used at inference is defined as

$$RAM = I + \sum_k K_k \cdot b_k^\omega / 8 + \sum_k F_k \cdot b_k^x / 8, \tag{S14}$$

where I is an input image size, K_k is the number of trainable variables per layer, and F_k is a feature size excluding a batch dimension. We sum up RAM over all network layers, assuming that each non-quantized feature is 32 bit. Note that both BOPs and RAM are differentiable w.r.t. bit width.

S3.3. Baseline algorithms

All implementations are done in Tensorflow framework. We compare the One-Shot methods with our implementation of EdMIPS and DNAS. The EdMIPS implementation strictly follows the original paper. The DNAS implementation is made comparable to EdMIPS by (I) using the same additive loss and Lloyd quantizer, and (II) selecting and fine-tuning a single architecture after searching. We also compare One-Shot results with the Bayesian Bits searcher applied to ResNet-18 and the fixed-precision FQSR [12] method applied to SRResNet. The results of these methods are taken from the corresponding articles. To the best of our knowledge, FQSR and One-Shot MPS are the only works that quantize all inputs and weights in super-resolution (SR) networks. Finally, we compare our method to HAQ [13] and GMPQ [14]. For a fair comparison, we reproduced both methods using Tensorflow framework. We carefully compared our reproduced results with those obtained using the original code available online, and did our best in hyperparameter tuning.

For reference, we plot the performance of fixed bit width architectures quantized by Lloyd, LSQ, and PACT. These algorithms always quantize the input of the first layer in 8 bits.

S3.4. Training hyperparameters

The hyperparameters used for the bit width searching and child model fine-tuning are summarized in Table S1.

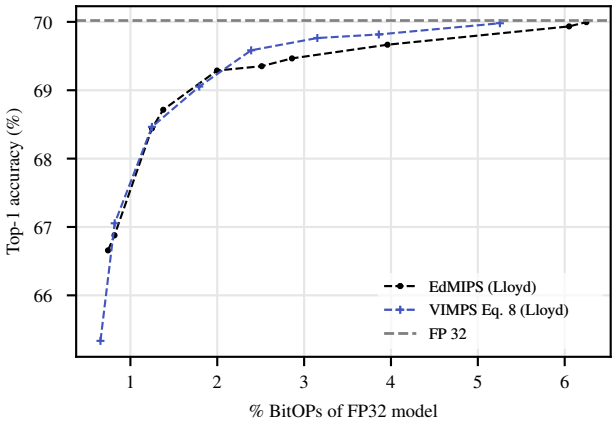


Figure S2. The ResNet-18 Pareto front obtained for EdMIPS and VIMPS (8) loss functions.

S4. Extra results

S4.1. Variational Inference Mixed-Precision Search (VIMPS)

Following the comments during the reviewing process, we decided to include the comparison plot of EdMIPS and VIMPS (8) losses. The difference between the two losses is summarized as follows:

1. EdMIPS uses approximation (11), while VIMPS uses approximation (9),
2. EdMIPS uses additive hardware constraints, while VIMPS uses Boltzmann form (3),
3. EdMIPS does not use entropy, while VIMPS uses entropy $H(\pi)$.

The comparison is done for the ResNet-18 model only. Results are presented in Figure S2 and Table S2. Given that the accuracy on ResNet-18 has a confidence interval of about $0.1\,\%$, VIMPS looks slightly better. However, more in-depth study is required.

S4.2. Tabular results

For easy comparison in future works, we summarize our results from Figure 5 in Tables S3, S4, S5, S6, S7, S8, and S9.

S4.3. Extra correlation plots

Figure S3 shows correlation plots for ESPCN, SRResNet, and MobileNet-v2. From the plots, we see that child models found by One-Shot MPS have target metrics closest to those of standalone models, except for MobileNet-v2. This is explained by the proxy dataset used during the searching stage. The dataset is simpler than the full ImageNet dataset due to which child models attain higher accuracies compared to standalone models.

Table S1. Training hyperparameters, and training time measured on a single (ESPCN and SRResNet) and a pair (ResNet-18 and MobileNet-v2) of Nvidia Tesla V100 GPU. SGD optimizer with 10^{-4} ($4 \cdot 10^{-5}$) weight decay and momentum 0.9 is used in runs marked with *(**). Adam is used in all other runs. Step optimizer is not applied for Lloyd quantizer at fine-tuning. The regularization η is applied to RAM measured in MB (for ESPCN and SRResNet) and bit operations measured in TeraBOPs (for ResNet-18 and MobileNet-v2).

Model	Algorithm	Regularizer η	Initial learning rates for:			Epochs	Single model training time
			weights ω	steps	architecture params θ		
ESPCN	One-Shot MPS	$[0.001, 50]$	0.001		0.005	50	30 min
	EdMIPS DNAS	$[0, 5]$	0.0001		0.001	25	10 min
	Fine-tune		0.0001	$5 \cdot 10^{-6}$		100	30 min
SRResNet	One-Shot MPS	$[10^{-6}, 1]$	0.0001		0.001	50	5 h
	EdMIPS	$[0, 0.05]$	0.001		0.001	25	1 h
	DNAS	$[0, 0.05]$	0.0001		0.01	25	45 min
	Fine-tune		0.001	0.0001		100	2 h
ResNet-18	One-Shot MPS	$[0.01, 20]$	0.1*		0.0005	20	3.4 h
	EdMIPS	$[0, 20]$	0.01*		0.001	25	1.7 h
	DNAS	$[0, 20]$	0.01*		0.001	25	3.4 h
	Fine-tune		0.01*	0.0001*		30	11.0 h
MobileNet-v2	One-Shot MPS	$[0.0003, 300]$	0.01**		0.001	20	11.0 h
	EdMIPS	$[0, 30]$	0.01**		0.001	25	3.8 h
	DNAS	$[0, 30]$	0.01**		0.001	25	6.7 h
	Fine-tune		0.01**	0.0001*		30	29.0 h

S4.4. The ESPCN and SRResNet quantization results on Set5

In Figure S4, the One-Shot MPS Pareto front on Set5 shows a similar behavior as for Set14 in Figure 5. One-Shot MPS with a Lloyd quantizer outperforms EdMIPS and DNAS searchers. The Trainable Lloyd quantizer improves the results even further. Note that the architectures found by the proposed method outperform 4-bit FQSR by a large margin.

S4.5. The SRResNet qualitative results

The existing SR metrics do not always reflect human perception of visual quality [5]. For this reason, we compare the visual quality of images produced by FP32 SRResNet and the mixed-precision models at roughly 35 % of the total RAM reduction. All searching algorithms (DNAS, EdMIPS, One-Shot MPS) find a mixed-precision model that achieves a similar visual quality compared to the FP32 SRResNet model. All SR Set5 and Set14 images obtained by the original FP32 SRResNet as well as by the searchers can be found in the folder named "SRResNet_Set5_set14_resulting_images". We hand-picked two images on which we could observe the visual quality difference. Figure S5 shows the cutout regions. The proposed One-Shot MPS produces more detailed textures at a higher compression rate compared to DNAS and EdMIPS. Also, note that all mixed-precision methods do not change the color temperature or hue.

S4.6. Bit width model probabilities

S4.6.1 ESPCN bit width model probabilities

Figures S6 and S7 show the input and weight probabilities depending on a logarithmic hardware penalty, respectively. As expected, the 8-bit precision (*green line*) is selected for all quantizers at the lowest penalty. For larger penalties, we can see that precision gradually changes from 8 bit, to 4, and then to 2 bits. One exception is the "conv2d" weight quantizer where 4-bit was never selected. We attribute this to the choice of the bit width model, which is a 1-layer model for ESPCN. We hypothesize that using a more complex bit width model (2-layer model as in other experiments) may fix this issue, *cf*. Figure 7 (*left*).

S4.6.2 SRResNet bit width model probabilities

In Figures S8 and S9, we show input and weight probabilities learned by the bit width model in each matrix multiplication layer of the SRResNet network. As with ESPCN in the previous section, we see that 8-bit precision is selected for the lowest values of the regularization parameter η, and then it changes to 4 and 2 bits for higher η. The red vertical line slices the plots at

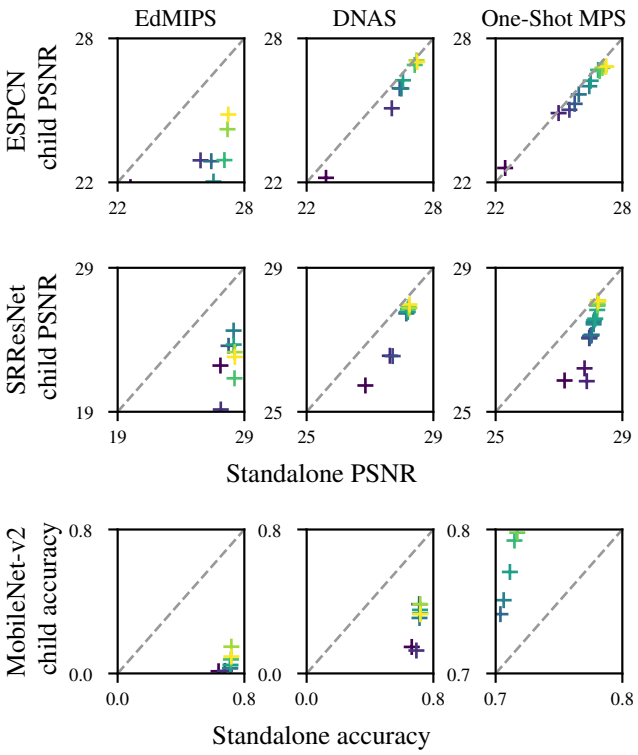


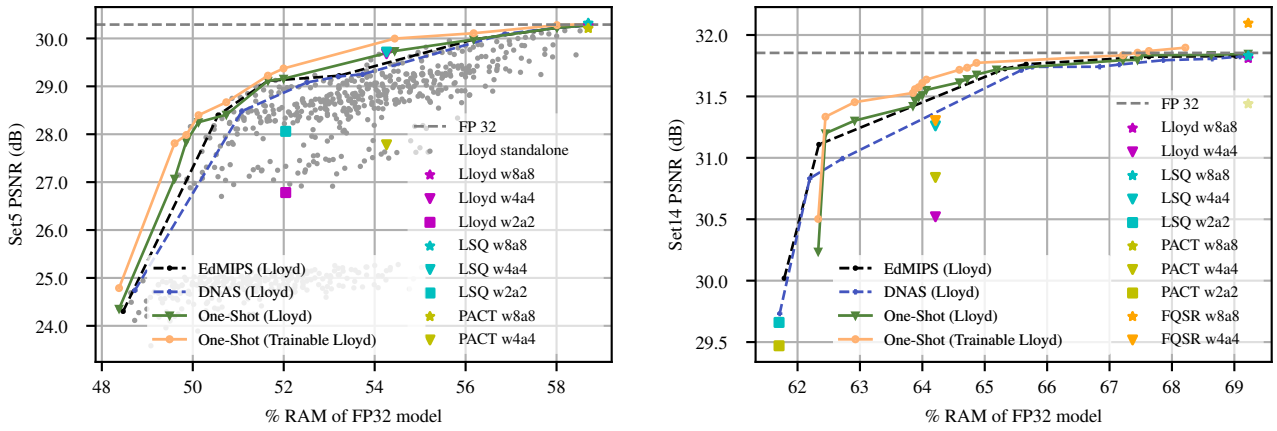
Figure S3. The plots visualize the correlation between the child and standalone models in terms of PSNR (ESPCN and SRResNet) and top-1 accuracy (MobileNet-v2). Brighter colors depict models of higher BOPs or RAM. For MobileNet-v2, the standalone model is trained and evaluated on a full ImageNet, while the child model is taken from a One-Shot supernet that is trained and evaluated on a proxy ImageNet. Note the difference in axis ranges.

$\eta = 0.0173$ which corresponds to the One-Shot MPS result in Figure S5. The input and weight bit widths at this intersection are $[8, 8, 4, 2, 4, 2, 4, 2, 4, 2, 4, 4, 8, 4, 8]$ and $[8, 4, 4, 2, 4, 4, 4, 4, 2, 4, 2, 4, 8, 4, 8]$, respectively.

We can also hypothesize why it is challenging to quantize SRResNet layers. The structure of SRResNet, presented in [7] in Figure 4, contains a long-distant skip connection between the low-level feature extractor and the up-sampler parts. The residual blocks, shortcut by the skip connection, have names from "conv2d_1" to "conv2d_11" in Figures S8 and S9. The shortcut is responsible for propagating the low-resolution feature which is essential for a good-quality reconstruction by the up-sampler. Therefore, it is preferable to keep the higher precision for the feature extractor (layer "conv2d") and the up-sampler (layers from "conv2d_12" to "conv2d_14"). One-Shot MPS finds 8 or 4 bit widths for these layers. On the other hand, the layers from "conv2d_1" to "conv2d_11" learn the high-resolution details. As we can see, they can be quantized to 2 bits without much degradation of PSNR, *cf*. Figure 5 and Figure S4. We think that such a precision contrast poses a challenge when the network is quantized in the same number of bits except 8.

S4.6.3 ResNet-18 bit width model probabilities

In Figures S10 and S11, we show input and weight probabilities learned by the bit width model for the ResNet-18 network. A similar picture to ESPCN and SRResNet is observed.



(a) ESPCN on Set5 (b) SRResNet on Set5

Figure S4. The additional verification of the quality of architectures found by One-Shot MPS on Set5. The notation "wXaY" indicates a fixed bit width architecture with 8-bit model input, X-bit weights, and Y-bit activations. All results in this plot are obtained by the authors.

Full-precision [7]	EdMIPS [3]	DNAS [15]	One-Shot MPS
RAM: 100 %	RAM: 65.3 %	RAM: 65.6 %	RAM: 64.9 %
Set5 PSNR: 31.9 dB	Set5 PSNR: 31.7 dB	Set5 PSNR: 31.7 dB	Set5 PSNR: 31.8 dB
Set14 PSNR: 28.3 dB	Set14 PSNR: 28.2 dB	Set14 PSNR: 28.2 dB	Set14 PSNR: 28.2 dB

Figure S5. The SRResNet qualitative results. The cutouts of a baboon from Set14 [17] (*first two rows*) and a baby from Set5 [1] (*last row*). Each column shows images produced by the same model specified below the images.

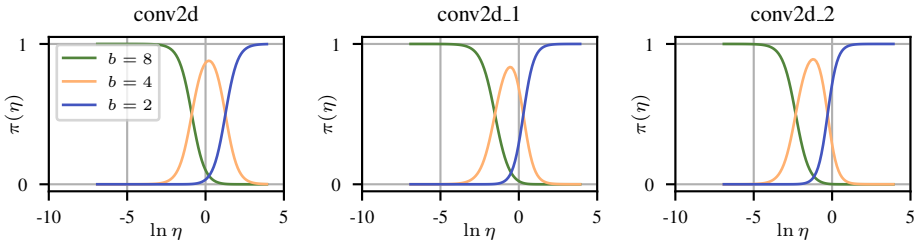


Figure S6. ESPCN bit width model input probabilities per layer.

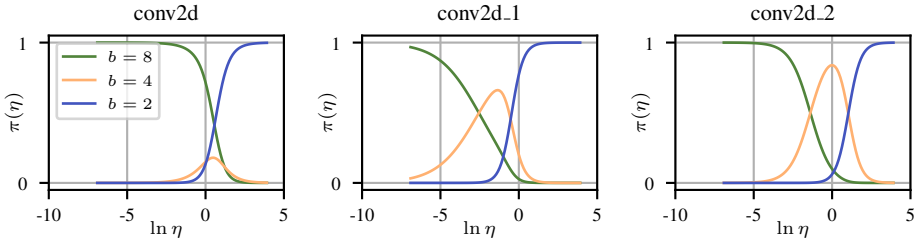


Figure S7. ESPCN bit width model weight probabilities per layer.

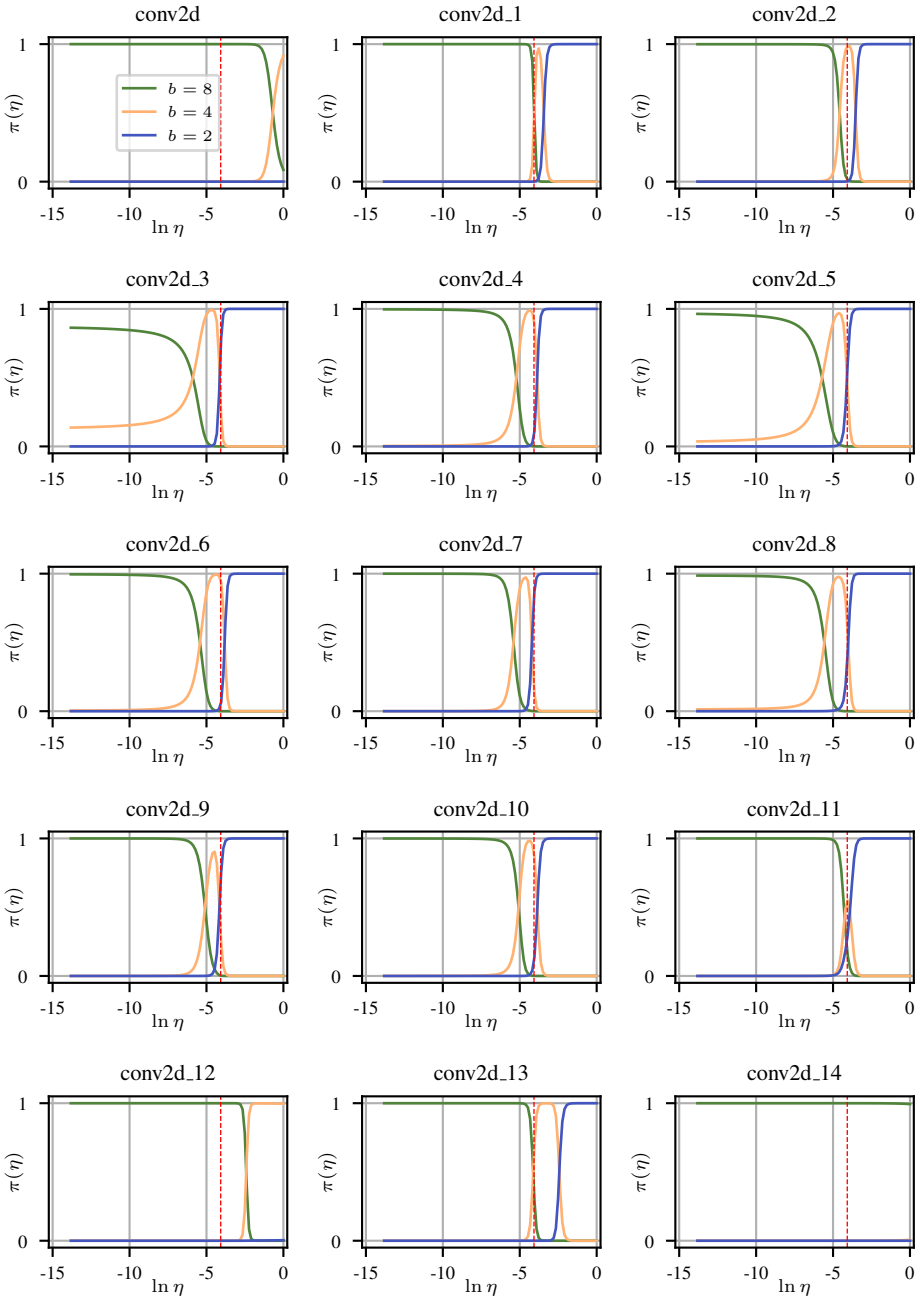


Figure S8. SRResNet bit width model input probabilities per layer.

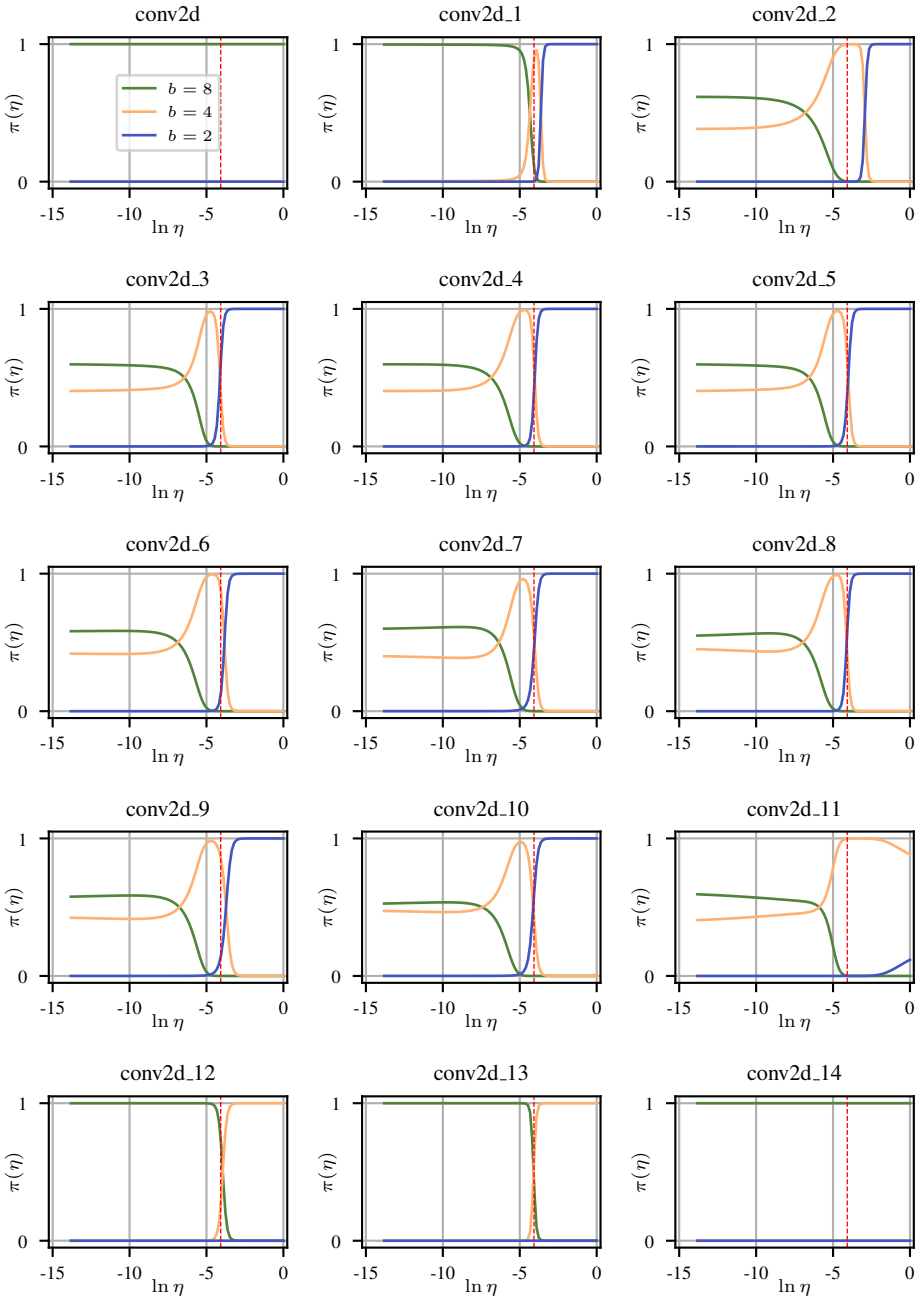


Figure S9. SRResNet bit width model weight probabilities per layer.

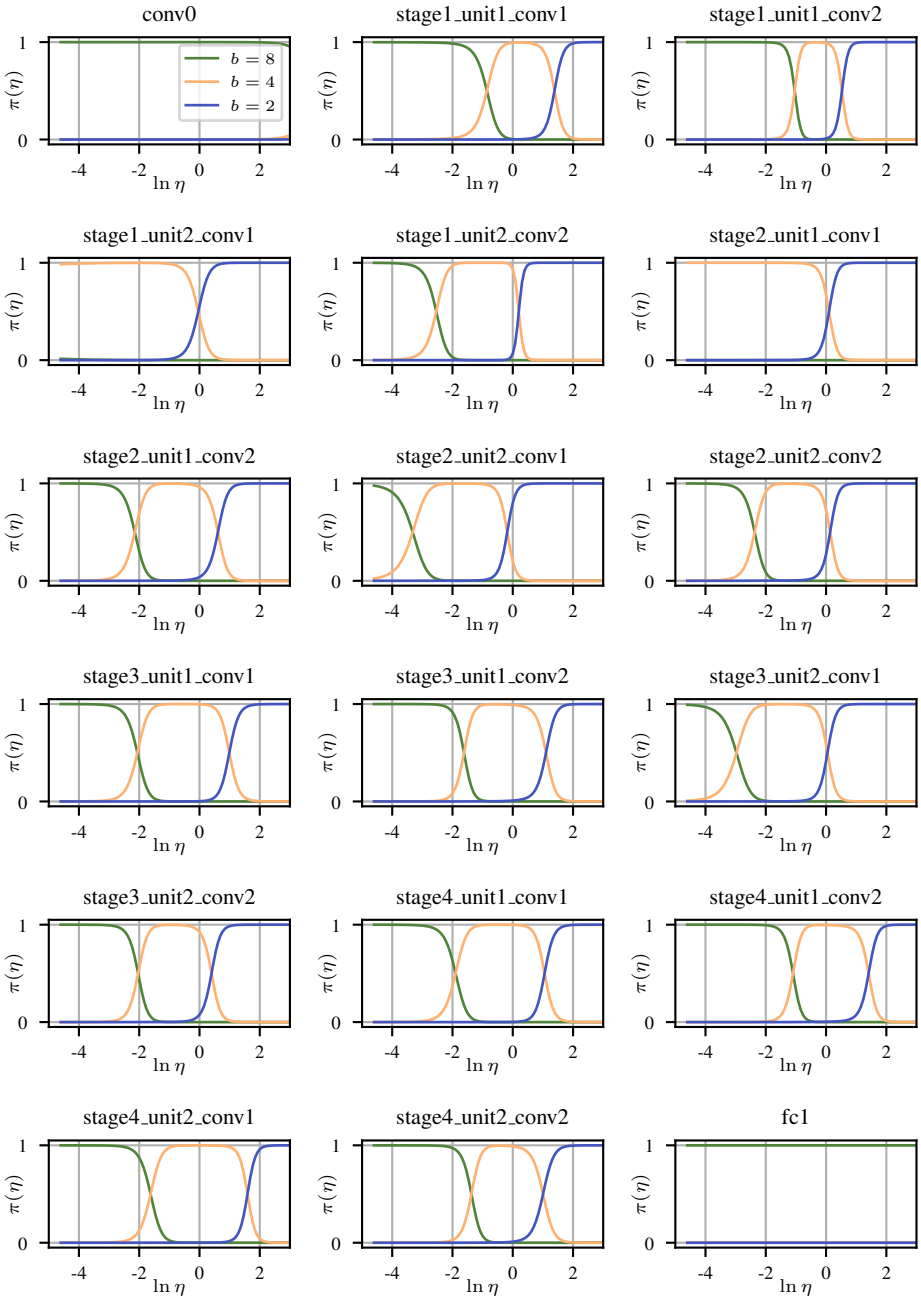


Figure S10. ResNet-18 bit width model input probabilities per layer.

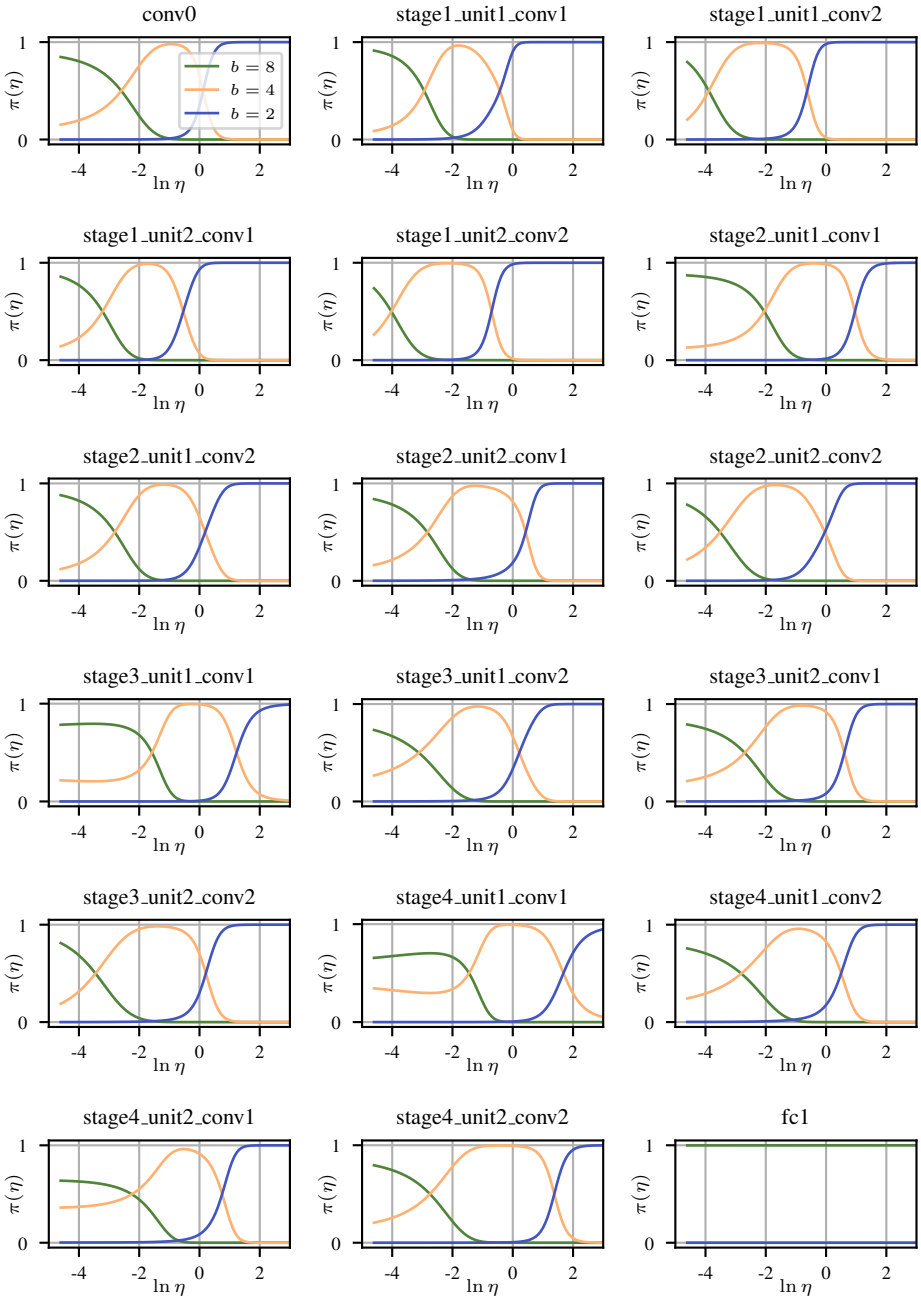


Figure S11. ResNet-18 bit width model weight probabilities per layer.

Table S2. Tabular results used for plotting Figure S2.

Method	BOPs $\times 10^9$	Top-1 accuracy, %
FP	1857.6	70.0
EdMIPS (Lloyd)	13.8	66.7
EdMIPS (Lloyd)	15.2	66.9
EdMIPS (Lloyd)	23.2	68.4
EdMIPS (Lloyd)	25.6	68.7
EdMIPS (Lloyd)	37.1	69.3
EdMIPS (Lloyd)	46.7	69.4
EdMIPS (Lloyd)	53.1	69.5
EdMIPS (Lloyd)	73.6	69.7
EdMIPS (Lloyd)	112.4	69.9
EdMIPS (Lloyd)	116.1	70.0
VIMPS Eq. 8 (Lloyd)	12.2	65.3
VIMPS Eq. 8 (Lloyd)	15.2	67.1
VIMPS Eq. 8 (Lloyd)	23.2	68.5
VIMPS Eq. 8 (Lloyd)	33.3	69.1
VIMPS Eq. 8 (Lloyd)	44.4	69.6
VIMPS Eq. 8 (Lloyd)	58.6	69.8
VIMPS Eq. 8 (Lloyd)	71.7	69.8
VIMPS Eq. 8 (Lloyd)	97.6	70.0

Table S3. Results obtained on ESPCN used for creating Figure 5a.

Method	RAM, MB	Set14 PSNR, dB
FP	1.28	27.26
Lloyd w8a8	0.75	27.24
Lloyd w4a4	0.69	26.86
Lloyd w2a2	0.67	24.94
LSQ w8a8	0.75	27.26
LSQ w4a4	0.69	26.84
LSQ w2a2	0.67	25.71
PACT w8a8	0.75	27.19
PACT w4a4	0.69	25.99
EdMIPS (Lloyd)	0.62	22.61
EdMIPS (Lloyd)	0.65	25.93
EdMIPS (Lloyd)	0.66	26.43
EdMIPS (Lloyd)	0.68	26.54
EdMIPS (Lloyd)	0.72	27.06
EdMIPS (Lloyd)	0.74	27.20
EdMIPS (Lloyd)	0.75	27.24
DNAS (Lloyd)	0.62	22.92
DNAS (Lloyd)	0.65	26.03
DNAS (Lloyd)	0.67	26.41
DNAS (Lloyd)	0.68	26.49
DNAS (Lloyd)	0.69	26.56
DNAS (Lloyd)	0.73	27.12
DNAS (Lloyd)	0.74	27.20
DNAS (Lloyd)	0.75	27.24
One-Shot (Lloyd)	0.62	22.45
One-Shot (Lloyd)	0.63	24.98
One-Shot (Lloyd)	0.64	25.49
One-Shot (Lloyd)	0.64	25.74
One-Shot (Lloyd)	0.65	25.94
One-Shot (Lloyd)	0.66	26.43
One-Shot (Lloyd)	0.67	26.50
One-Shot (Lloyd)	0.70	26.93
One-Shot (Lloyd)	0.72	27.06
One-Shot (Lloyd)	0.74	27.20
One-Shot (Lloyd)	0.75	27.24
One-Shot (Trainable Lloyd)	0.62	22.82
One-Shot (Trainable Lloyd)	0.63	25.48
One-Shot (Trainable Lloyd)	0.64	25.60
One-Shot (Trainable Lloyd)	0.64	25.83
One-Shot (Trainable Lloyd)	0.65	26.05
One-Shot (Trainable Lloyd)	0.66	26.49
One-Shot (Trainable Lloyd)	0.67	26.58
One-Shot (Trainable Lloyd)	0.70	27.03
One-Shot (Trainable Lloyd)	0.72	27.13
One-Shot (Trainable Lloyd)	0.74	27.24
One-Shot (Trainable Lloyd)	0.75	27.25

Table S4. Conventional method's results obtained on SRResNet used for creating Figure 5b.

Method	RAM, MB	Set14 PSNR, dB
FP	26.75	28.26
Lloyd w8a8	18.52	28.23
Lloyd w4a4	17.18	27.42
LSQ w8a8	18.52	28.25
LSQ w4a4	17.18	27.88
LSQ w2a2	16.51	26.80
PACT w8a8	18.52	27.98
PACT w4a4	17.18	27.64
PACT w2a2	16.51	26.69
FQSR w8a8	18.52	28.56
FQSR w4a4	17.18	28.05
EdMIPS (Lloyd)	16.53	27.11
EdMIPS (Lloyd)	16.59	27.14
EdMIPS (Lloyd)	16.68	27.75
EdMIPS (Lloyd)	17.48	28.15
EdMIPS (Lloyd)	17.57	28.16
EdMIPS (Lloyd)	18.09	28.22
EdMIPS (Lloyd)	18.47	28.24
EdMIPS (Lloyd)	18.52	28.25
DNAS (Lloyd)	16.51	26.86
DNAS (Lloyd)	16.64	27.63
DNAS (Lloyd)	16.78	27.72
DNAS (Lloyd)	17.55	28.15
DNAS (Lloyd)	17.59	28.16
DNAS (Lloyd)	17.88	28.20
DNAS (Lloyd)	18.15	28.22
DNAS (Lloyd)	18.36	28.24
DNAS (Lloyd)	18.48	28.25

Table S5. One-Shot MPS results obtained on SRResNet used for creating Figure 5b.

Method	RAM, MB	Set14 PSNR, dB
One-Shot (Lloyd)	16.67	27.18
One-Shot (Lloyd)	16.71	27.81
One-Shot (Lloyd)	16.83	27.88
One-Shot (Lloyd)	17.08	27.95
One-Shot (Lloyd)	17.09	27.96
One-Shot (Lloyd)	17.11	27.98
One-Shot (Lloyd)	17.12	28.02
One-Shot (Lloyd)	17.14	28.03
One-Shot (Lloyd)	17.28	28.09
One-Shot (Lloyd)	17.35	28.11
One-Shot (Lloyd)	17.44	28.14
One-Shot (Lloyd)	17.98	28.21
One-Shot (Lloyd)	18.04	28.21
One-Shot (Lloyd)	18.06	28.21
One-Shot (Lloyd)	18.25	28.22
One-Shot (Lloyd)	18.36	28.24
One-Shot (Trainable Lloyd)	16.67	27.31
One-Shot (Trainable Lloyd)	16.71	27.90
One-Shot (Trainable Lloyd)	16.83	27.97
One-Shot (Trainable Lloyd)	17.08	28.05
One-Shot (Trainable Lloyd)	17.11	28.06
One-Shot (Trainable Lloyd)	17.14	28.10
One-Shot (Trainable Lloyd)	17.28	28.15
One-Shot (Trainable Lloyd)	17.31	28.16
One-Shot (Trainable Lloyd)	17.35	28.18
One-Shot (Trainable Lloyd)	17.98	28.23
One-Shot (Trainable Lloyd)	18.04	28.23
One-Shot (Trainable Lloyd)	18.25	28.25
One-Shot (Trainable Lloyd)	18.36	28.27
One-Shot (Trainable Lloyd)	18.52	28.27

Table S6. Conventional method's results obtained on ResNet-18 used for creating Figure 5c.

Method	BOPs $\times 10^9$	Top-1 accuracy, %
FP	1857.6	70.0
Lloyd w8a8	116.1	70.0
Lloyd w4a4	30.9	69.0
LSQ w8a8	116.1	70.0
LSQ w4a4	30.9	68.9
PACT w8a8	116.1	69.3
PACT w4a4	30.9	68.5
EdMIPS (Lloyd)	13.8	66.7
EdMIPS (Lloyd)	15.2	66.9
EdMIPS (Lloyd)	23.2	68.4
EdMIPS (Lloyd)	25.6	68.7
EdMIPS (Lloyd)	37.1	69.3
EdMIPS (Lloyd)	46.7	69.4
EdMIPS (Lloyd)	53.1	69.5
EdMIPS (Lloyd)	73.6	69.7
EdMIPS (Lloyd)	112.4	69.9
EdMIPS (Lloyd)	116.1	70.0
DNAS (Lloyd)	19.4	67.8
DNAS (Lloyd)	24.9	68.6
DNAS (Lloyd)	27.2	68.9
DNAS (Lloyd)	30.4	69.1
DNAS (Lloyd)	36.5	69.2
DNAS (Lloyd)	60.5	69.7
DNAS (Lloyd)	112.4	70.1
Bayesian Bits (PACT)	18.9	67.1
Bayesian Bits (PACT)	26.7	68.2
Bayesian Bits (PACT)	33.4	69.2
Bayesian Bits (PACT)	39.2	69.6
Bayesian Bits (PACT)	46.6	69.9
HAQ (Lloyd)	22.2	67.6
HAQ (Lloyd)	36.6	69.1
HAQ (Lloyd)	74.9	69.6
HAQ (Lloyd)	81.9	69.8
GMPQ (Lloyd)	21.9	66.5
GMPQ (Lloyd)	33.2	68.1
GMPQ (Lloyd)	42.8	68.4
GMPQ (Lloyd)	101.1	69.6
GMPQ (Lloyd)	116.1	70

Table S7. One-Shot MPS results obtained on ResNet-18 used for creating Figure 5c.

Method	BOPs $\times 10^9$	Top-1 accuracy, %
One-Shot (Lloyd)	15.2	66.1
One-Shot (Lloyd)	15.7	66.7
One-Shot (Lloyd)	17.1	66.8
One-Shot (Lloyd)	17.6	67.0
One-Shot (Lloyd)	19.5	67.4
One-Shot (Lloyd)	22.7	68.2
One-Shot (Lloyd)	30.2	69.2
One-Shot (Lloyd)	34.8	69.3
One-Shot (Lloyd)	53.0	69.6
One-Shot (Lloyd)	62.5	69.7
One-Shot (Lloyd)	73.6	69.8
One-Shot (Trainable Lloyd)	15.2	66.7
One-Shot (Trainable Lloyd)	15.7	67.1
One-Shot (Trainable Lloyd)	17.1	67.4
One-Shot (Trainable Lloyd)	17.6	67.5
One-Shot (Trainable Lloyd)	19.5	67.7
One-Shot (Trainable Lloyd)	22.7	68.4
One-Shot (Trainable Lloyd)	30.2	69.2
One-Shot (Trainable Lloyd)	34.8	69.3
One-Shot (Trainable Lloyd)	53.0	69.8
One-Shot (Trainable Lloyd)	56.9	69.9
One-Shot (Trainable Lloyd)	62.5	69.9

Table S8. Conventional method's results obtained on MobileNet-v2 used for creating Figure 5d. Note that Bayesian Bits numbers were taken from Figure 2b of the arXiv version of [11].

Method	BOPs $\times 10^9$	Top-1 accuracy, %
FP	308.0	72.2
Lloyd w8a8	19.2	71.7
Lloyd w4a4	5.3	68.2
LSQ w8a8	19.2	71.7
LSQ w4a4	5.3	68.3
PACT w8a8	19.2	70.2
PACT w4a4	5.3	67.7
EdMIPS (Lloyd)	2.3	63.7
EdMIPS (Lloyd)	3.7	67.7
EdMIPS (Lloyd)	7.5	71.0
EdMIPS (Lloyd)	8.4	71.1
EdMIPS (Lloyd)	9.4	71.3
EdMIPS (Lloyd)	11.1	71.5
EdMIPS (Lloyd)	16.5	71.8
DNAS (Lloyd)	2.8	66.4
DNAS (Lloyd)	4.6	69.3
DNAS (Lloyd)	7.9	71.0
DNAS (Lloyd)	9.4	71.3
DNAS (Lloyd)	10.7	71.6
DNAS (Lloyd)	13.0	71.7
DNAS (Lloyd)	17.6	71.8
DNAS (Lloyd)	19.2	71.8
Bayesian Bits (PACT)	3.4	63.2
Bayesian Bits (PACT)	4.0	66.3
Bayesian Bits (PACT)	4.8	67.7
Bayesian Bits (PACT)	5.9	69.0
Bayesian Bits (PACT)	7.1	69.5
Bayesian Bits (PACT)	10.5	70.8
Bayesian Bits (PACT)	14.6	71.8
Bayesian Bits (PACT)	19.2	72.0

Table S9. One-Shot MPS results obtained on MobileNet-v2 used for creating Figure 5d.

Method	BOPs $\times 10^9$	Top-1 accuracy, %
One-Shot (Lloyd)	2.8	63.7
One-Shot (Lloyd)	4.5	67.7
One-Shot (Lloyd)	6.0	69.9
One-Shot (Lloyd)	7.5	70.4
One-Shot (Lloyd)	8.3	70.6
One-Shot (Lloyd)	10.2	71.1
One-Shot (Lloyd)	11.7	71.5
One-Shot (Lloyd)	13.6	71.7
One-Shot (Lloyd)	16.7	71.8
One-Shot (Trainable Lloyd)	2.8	66.9
One-Shot (Trainable Lloyd)	4.5	69.1
One-Shot (Trainable Lloyd)	6.0	70.8
One-Shot (Trainable Lloyd)	7.5	71.1
One-Shot (Trainable Lloyd)	8.3	71.3
One-Shot (Trainable Lloyd)	10.2	71.5
One-Shot (Trainable Lloyd)	11.7	71.7
One-Shot (Trainable Lloyd)	13.6	71.8
One-Shot (Trainable Lloyd)	16.7	72.0

References

[1] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-line Alberi Morel. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In *Proceedings of the British Machine Vision Conference*, 2012.

[2] Z. Cai, X. He, J. Sun, and N. Vasconcelos. Deep Learning with Low Precision by Half-Wave Gaussian Quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[3] Zhaowei Cai and Nuno Vasconcelos. Rethinking Differentiable Search for Mixed-Precision Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2349–2358, 2020.

[4] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I.-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: Parameterized Clipping Activation for Quantized Neural Networks. *arXiv:1805.06085*, 2018.

[5] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P. Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[6] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned Step Size Quantization. In *Proceedings of the International Conference on Learning Representations*, 2020.

[7] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[8] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, Baochang Zhang, Fan Yang, and Rongrong Ji. PAMS: Quantized Super-Resolution via Parameterized Max Scale. In *Proceedings of the European Conference on Computer Vision*, pages 564–580, 2020.

[9] S. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[10] William A. Pearlman and Amir Said. *Digital Signal Compression: Principles and Practice*. Cambridge University Press, 2011.

[11] Mart Van Baalen, Christos Louizos, Markus Nagel, Rana Ali Amjad, Ying Wang, Tijmen Blankevoort, and Max Welling. Bayesian Bits: Unifying Quantization and Pruning. *Advances in Neural Information Processing Systems*, 33:5741–5752, 2020.

[12] Hu Wang, Peng Chen, Bohan Zhuang, and Chunhua Shen. Fully Quantized Image Super-Resolution Networks. *arXiv:2011.14265*, 2021.

[13] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-Aware Automated Quantization with Mixed Precision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8612–8620, 2019.

[14] Ziwei Wang, Han Xiao, Jiwen Lu, and Jie Zhou. Generalizable mixed-precision quantization via attribution rank preservation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5291–5300, 2021.

[15] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed Precision Quantization of Convnets via Differentiable Neural Architecture Search. *arXiv:1812.00090*, 2018.

[16] Stone Yun and Alexander Wong. Do All MobileNets Quantize Poorly? Gaining Insights into the Effect of Quantization on Depthwise Separable Convolutional Networks Through the Eyes of Multi-scale Distributional Dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2447–2456, 2021.

[17] Roman Zeyde, Michael Elad, and Matan Protter. On Single Image Scale-Up Using Sparse-Representations. In *Proceedings of the Curves and Surfaces*, pages 711–730, 2012.