Machine Learning (2021) 110:3245–3262
https://doi.org/10.1007/s10994-021-06053-z

Loss aware post-training quantization

Yury Nahshan[1] · Brian Chmiel[1,2] · Chaim Baskin[3] · Evgenii Zheltonozhskii[3] · Ron Banner[1] · Alex M. Bronstein[3] · Avi Mendelson[3]

Received: 29 November 2020 / Revised: 10 June 2021 / Accepted: 23 August 2021 /
Published online: 1 October 2021

Abstract
Neural network quantization enables the deployment of large models on resource-constrained devices. Current post-training quantization methods fall short in terms of accuracy for INT4 (or lower) but provide reasonable accuracy for INT8 (or above). In this work, we study the effect of quantization on the structure of the loss landscape. We show that the structure is flat and separable for mild quantization, enabling straightforward post-training quantization methods to achieve good results. We show that with more aggressive quantization, the loss landscape becomes highly non-separable with steep curvature, making the selection of quantization parameters more challenging. Armed with this understanding, we design a method that quantizes the layer parameters jointly, enabling significant accuracy improvement over current post-training quantization methods. Reference implementation is available at https://github.com/ynahshan/nn-quantization-pytorch/tree/master/lapq.

Keywords Convolutional neural networks · Post-training quantization

1 Introduction

Deep neural networks (DNNs) are a powerful tool that have shown unmatched performance in various tasks in computer vision, natural language processing and optimal control, to mention only a few. The high computational resource requirements, however, constitute one of the main drawbacks of DNNs, hindering their massive adoption on edge devices. With the growing number of tasks performed on edge devices, e.g.,

Editor: Hendrik Blockeel.

Yury Nahshan, Brian Chmiel, Chaim Baskin and Evgenii Zheltonozhskii have contributed equally.

✉ Evgenii Zheltonozhskii
evgeniizh@campus.technion.ac.il

1 Habana Labs-An Intel company, Caesarea, Israel

2 Department of Electrical Engineering-Technion, Haifa, Israel

3 Department of Computer Science-Technion, Haifa, Israel

smartphones or embedded systems, and the availability of dedicated custom hardware for DNN inference, the subject of DNN compression has gained popularity.

One way to improve DNN computational efficiency is to use lower-precision representation of the network, also known as quantization. Most of the literature on neural network quantization involves training either from scratch (Bethge et al. 2019; Jin et al. 2019) or performing fine-tuning on a pre-trained full-precision model (Yang et al. 2019; Hubara et al. 2017). While training is a powerful method to compensate for accuracy loss due to quantization, it is both resource consuming and requires access to the data the model was trained on, which is not always available. Thus, it is often desirable to be able to quantize the model without training. These methods are commonly referred to as *post-training quantization* and usually require only a small calibration dataset. Unfortunately, current post-training methods are not very efficient, and most existing works only manage to quantize parameters to the 8-bit integer representation (INT8).

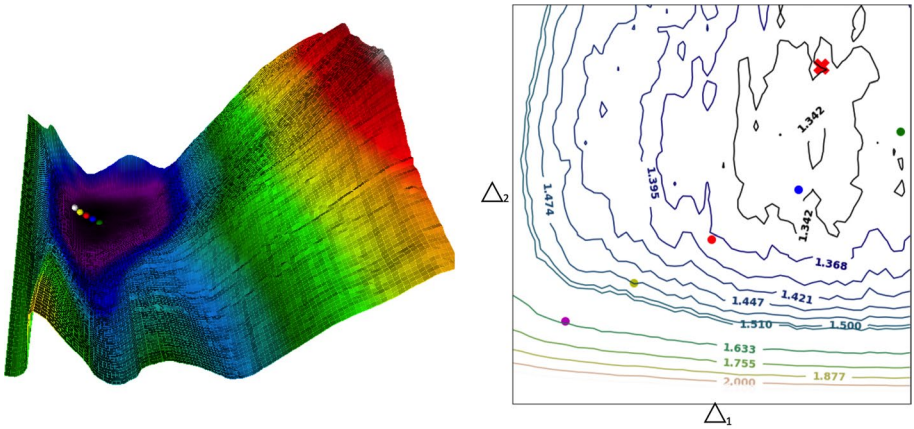
In the absence of a training set, these methods typically aim at minimizing the local error introduced during the quantization process (e.g., round-off errors). A popular approach to minimizing this error has been to clip the tensor outliers. This means that peak values will incur a larger error, but, in total, this will reduce the distortion introduced by the limited resolution (Lee et al. 2018; Banner et al. 2018; Zhao et al. 2019). However, these schemes suffer from two fundamental drawbacks.

Firstly, it is hard or even impossible to choose an optimal metric for the network performance based on the tensor level quantization error. In particular, even for the same task, similar architectures may favor different objectives. Secondly, the noise in earlier layers might be amplified by successive layers, creating a dependency between quantizaton errors of different layers. This cross-layer dependency makes it necessary to jointly optimize the quantization parameters across all network layers; however, current methods optimize them separately for each layer. Figure 1 plots the challenging loss surface of this optimization process.

Below, we outline the main contributions of the present work along with the organization of the remaining sections.

- First we consider current layer-by-layer quantization methods where the quantization step size within each layer is optimized to accommodate the dynamic range of the tensor while keeping it small enough to minimize quantization noise. Although these methods optimize the quantization step size of each layer independently of the other layers, we observe strong interactions between the layers, explaining their suboptimal performance at the network level.
- Accordingly, we consider network quantization as a multivariate optimization problem where the layer quantization step sizes are jointly optimized to minimize the cross-entropy loss. We observe that layer-by-layer quantization identifies solutions in a small region around the optimum, where degradation is quadratic in the distance from it. We provide analytical justification as well as empirical evidence showing this effect.
- Finally, we propose to combine layer-by-layer quantization with multivariate quadratic optimization. Our method is shown to significantly outperform state-of-the-art methods on two different challenging tasks and six DNN architectures.

The rest of the paper is organized as follows: Sect. 2 reviews the related work, Sect. 3 studies the properties of the loss function of the quantized network, Sect. 4 describes a proposed method, Sect. 5 provides the experimental results, and Sect. 6 concludes the paper.



(a) (b)

Fig. 1 A visualization of the loss surface for the pair of layers using one batch of 512 images (ResNet18 on ImageNet). X and Y-axis are quantization ranges of those layers where we estimate the cross-entropy loss (z-axis). The colored dots mark the quantization range found by optimizing different p-norm metrics. Each layer quantized in a way that minimizes the p-norm distance between the quantized tensor to its original full precision counterpart (e.g., MSE distortion). **a** One can see high coupling between two quantization ranges, making layer-wise optimization sub-optimal. **b** A zoom-in of sub-figure (a) to the point where the optimal cross-entropy loss (red cross) clearly distinguished from the optimized p-norm solutions.

2 Related work

Quantization is a long-studied topic in computer science. In classical digital signal processing, the goal of quantization is usually to minimize some kind of distortion of the signal (Max 1960; Berger 1972; Lloyd 1982). In neural networks, however, the goal is to maximize some performance metric of the network, rather then distortion of the weights and activations. For example, Hou et al. (2016); Hou and Kwok (2018) have shown that we can utilize the Hessian of the loss to improve quantization of the CNNs.

The approaches to neural network quantization can be divided into two major categories (Krishnamoorthi 2018): quantization-aware training, which introduces the quantization at some point during the training, and post-training quantization, where the network weights are not optimized during the quantization. The recent progress in quantization-aware training has allowed the realization of results comparable to a baseline for as low as 2–4 bits per parameter (Baskin et al. 2018; Zhang et al. 2018; Gong et al. 2019; Yang et al. 2019; Jin et al. 2019) and show decent performance even for single bit (binary) parameters (Liu et al. 2019; Peng and Chen 2019; Kim et al. 2020). The major drawback of quantization-aware methods is the necessity for a vast amount of labeled data and high computational power. Consequently, post-training quantization is widely used in existing embedded hardware solutions. Most work that proposes hardware-friendly post-training schemes has only managed to get to 8-bit quantization without significant degradation in performance or requiring substantial modifications in existing hardware.

Lin et al. (2015) proposed to use SQNR-optimal quantization, using Gaussian assumption to implement efficient post-training quantization. Moreover, using the approximation of SQNR of the whole network, they were able to find optimal allocation of different bitwidth to CNN layers. Zhao et al. (2017) enhanced the approach by utilizing robustness estimation (Fawzi et al. 2016).

Gong et al. (2018) have shown that min-max (ℓ_∞ norm) as a threshold can be used for 8-bit quantization. It produces a small performance degradation and can be deployed efficiently on the hardware. Afterward, better schemes involving the choosing of the clipping value, such as minimizing the Kullback-Leibler divergence (Migacz 2017), assuming the known distribution of weights (Banner et al. 2018), or minimizing quantization MSE iteratively (Kravchik et al. 2019), were proposed.

Another efficient approach quantization is to change the distribution of the tensors, making it more suitable for quantization, such as equalizing the weight ranges (Nagel et al. 2019) or splitting outlier channels (Zhao et al. 2019).

Recent works noted that the quantization process introduces a bias into distributions of the parameters and attempt to correct this bias. Finkelstein et al. (2019) addressed a problem of MobileNet quantization. They claimed that the source of degradation was shifting in the mean activation value caused by inherent bias in the quantization process and proposed a scheme for fixing this bias. Multiple alternative schemes for the correction of quantization bias were proposed, and those techniques are widely applied in state-of-the-art quantization approaches (Banner et al. 2018; Finkelstein et al. 2019; Nagel et al. 2019; Fang et al. 2020). In our work, we utilize the bias correction method developed by Banner et al. (2018).

While the simplest form of quantization applies a single quantizer to the whole tensor (weights or activations), finer quantization allows reduction of performance degradation. Even though this approach usually boosts performance significantly (Mellempudi et al. 2017; Banner et al. 2018; Fraser et al. 2018; Kravchik et al. 2019; Lee et al. 2018), it requires more parameters and special hardware support, which makes it unfavorable for real-life deployment.

One can achieve more powerful quantization by using more sophisticated ways to map values to a particular bin as clustering of the tensor entry values (Nayak et al. 2019), aka non-uniform quantization (Fang et al. 2020). By allowing a more general quantizer, these approaches provide better performance than uniform quantization, as a drawback requires complex hardware support. Similarly to fine-grained quantization, they are less suitable for deployment on consumer-grade hardware. In current work, we focus on the case of the symmetric tensor-wise quantizer, which is easy to implement in the hardware.

To the best of our knowledge, our work is first to consider dependencies between layers in the post-training regime.

3 Loss landscape of quantized DNNs

In this section, we introduce the notion of separability of the loss function. We study the separability and the curvature of the loss function and show how quantization of DNNs affect these properties. Finally, we show that during aggressive quantization, the loss function becomes highly non-separable with steep curvature, which is unfavorable for existing post-training quantization methods. Our method addresses these properties and makes post-training quantization possible at low bit quantization.

We focus on uniform quantization with a fixed number of bits M for all layers and quantization step size Δ that maps a value $x \in \mathbb{R}$ into a discrete representation,

$$Q_\Delta(x) = \begin{cases} x_{\min} & x \leqslant x_{\min} \\ x_{\min} + \left\lfloor \dfrac{x - x_{\min}}{\Delta} \right\rceil \Delta & x_{\min} < x < x_{\max} \\ x_{\max} & x \geqslant x_{\max}. \end{cases} \tag{1}$$

By constraining the range of x to $[-c, c]$ (i.e., $-x_{\min} = x_{\max} = c$), the connection between c and Δ is given by:

$$\Delta = \frac{c}{2^{M-1} - 1}, \tag{2}$$

where we use odd number of bins so that one of the bins is mapped to zero. For example, if $c = 1$ and $M = 4$, the step size is $\Delta = \frac{1}{7}$, and there are total of 15 possible quatization values: $Q_{\Delta,M}(x) \in \{-1, -6/7, \ldots, 0, 1/7, \ldots, 1\}$.

In the case of activations, we limit ourselves to the ReLU function, which allows us to choose a quantization range of $[0, c]$. In such cases, the quantization step Δ is given by:

$$\Delta = \frac{c}{2^M - 1}. \tag{3}$$

3.1 Separable optimization

Suppose the loss function of the network $\mathcal{L}$ depends on a certain set of variables (weights, activations, etc.), which we denote by a vector $\mathbf{v}$. We would like to measure the effect of adding quantization noise to this set of vectors. In the following we show that for sufficiently small quantization noise, we can treat it as an additive noise vector $\boldsymbol{\varepsilon}$, allowing coordinate-wise optimization. However, when quantization noise is increased, the degradation in one layer is associated with other layers, calling for more laborious non-separable optimization techniques. Since the quantization is emulated with an additive noise, the loss is smooth and thus can be expanded to Taylor series:

$$\Delta\mathcal{L} = \mathcal{L}(\mathbf{v} + \boldsymbol{\varepsilon}) - \mathcal{L}(\mathbf{v}) \tag{4}$$

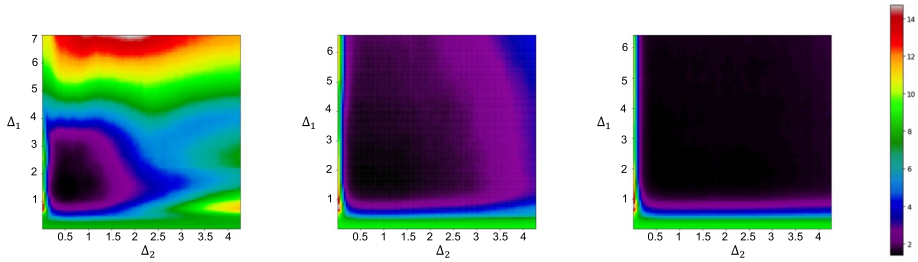
$$= \frac{\partial \mathcal{L}}{\partial \mathbf{v}}^\top \boldsymbol{\varepsilon} + \boldsymbol{\varepsilon}^\top \frac{\partial^2 \mathcal{L}}{\partial \mathbf{v}^2} \boldsymbol{\varepsilon} + \mathcal{O}(||\boldsymbol{\varepsilon}||^3). \tag{5}$$

When the quantization error $\boldsymbol{\varepsilon}$ is sufficiently small, higher-order terms can be neglected so that degradation $\Delta\mathcal{L}$ can be approximated as a sum of the quadratic functions,

$$\Delta\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \mathbf{v}}^\top \boldsymbol{\varepsilon} + \boldsymbol{\varepsilon}^\top \frac{\partial^2 \mathcal{L}}{\partial \mathbf{v}^2} \boldsymbol{\varepsilon} \tag{6}$$

$$= \sum_i^n \frac{\partial \mathcal{L}}{\partial v_i} \cdot \varepsilon_i + \sum_i^n \sum_j^n \frac{\partial \mathcal{L}}{\partial v_i v_j} \varepsilon_i \cdot \varepsilon_j. \tag{7}$$

One can see from Eq. 7 that when quantization error $||\boldsymbol{\varepsilon}||^2$ is sufficiently small, the overall degradation $\Delta\mathcal{L}$ can be approximated as a sum of N independent separable degradation processes as follows:



(a) 2 bit **(b)** 3 bit **(c)** 4 bit

Fig. 2 Visualization of the loss surface as a function of quantization ranges of two subsequent layers of ResNet18. At higher bit-width, with more fine-grained quantization 2c, the interactions between layers are relatively small, making layer-wise optimization possible. On the other hand, at lower bit-width 2a, with an increase of the quantization noise, the interactions between two layers become tangible. Due to the high coupling, the per layer (per axis) optimization results will entirely depend on the initial point and potentially sub-optimal.

$$\Delta\mathcal{L} \approx \sum_{i}^{n} \frac{\partial \mathcal{L}}{\partial v_i} \cdot \varepsilon_i \tag{8}$$

On the other hand, when $||\boldsymbol{\varepsilon}||^2$ is larger, one needs to take into account the interactions between different layers, corresponding to the second term in Eq. 7 as follows:

$$\text{QIT} = \sum_{i}^{n} \sum_{j}^{n} \frac{\partial \mathcal{L}}{\partial v_i \partial v_j} \varepsilon_i \cdot \varepsilon_j, \tag{9}$$

where QIT refers to the quantization interaction term.

One can see that at fine-grained quantization when $||\boldsymbol{\varepsilon}||^2$ is small the overall degradation, $\Delta\mathcal{L}$, is additively separable and can be layer-wise optimized. However, for more aggressive quantization, the layer-wise optimization without taking into account interactions between layers (QIT) will lead to suboptimal results.

In Fig. 2 we provide a visualization of these interactions at 2, 3 and 4 bitwidth representations.

3.2 Curvature

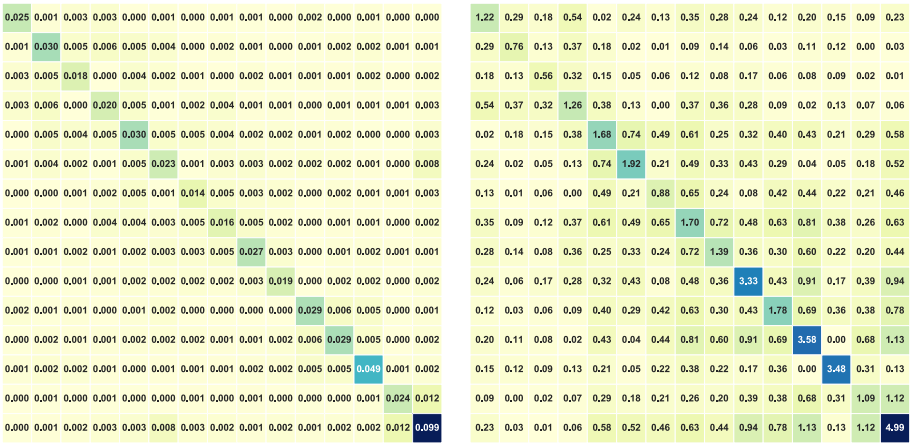
We now analyze how the steepness of the curvature of the loss function with respect to the quantization step changes as the quantization error increases. We will show that at aggressive quantization, the curvature of the loss becomes steep, which is unfavorable for the methods that aim to minimize quantization error on the tensor level.

We start by defining a measure to quantify curvature of the loss function with respect to quantization step size Δ. Given a quantized neural network, we denote $\mathcal{L}(\Delta_1, \Delta_2, \ldots, \Delta_n)$ the loss with respect to quantization step size Δ_i of each individual layer. Since $\mathcal{L}$ is twice differentiable with respect to Δ_i we can calculate the Hessian matrix:

$$H[\mathcal{L}]_{ij} = \frac{\partial \mathcal{L}}{\partial \Delta_i \partial \Delta_j}. \tag{10}$$

Table 1 Gaussian curvature, $K[\mathcal{L}_{\text{b bit}}](\Delta)$, for various models and bit count.

Bit count	ResNet-18	ResNet-50	Inception-V3
2	0.58	2.15	1.96
3	$3.2 \cdot 10^{-5}$	$1.3 \cdot 10^{-4}$	$9.7 \cdot 10^{-5}$
4	$6.7 \cdot 10^{-25}$	$5.4 \cdot 10^{-22}$	$1.1 \cdot 10^{-23}$



(a) Quantization of 4 bit **(b)** Quantization of 2 bit

Fig. 3 Absolute value of the Hessian matrix of the loss function with respect to quantization steps calculated over 15 layers of ResNet-18. Higher values at a diagonal of the Hessian at 2-bit quantization suggest that the minimum is sharper than at 4 bits. Non-diagonal elements provide an indication of the coupling between parameters of different layers: closer layers generally exhibit stronger interactions.

To quantify the curvature, we use Gaussian curvature (Goldman 2005), which is given by:

$$K[\mathcal{L}](\Delta) = \frac{\det(H[\mathcal{L}](\Delta))}{(||\nabla \mathcal{L}(\Delta)||_2^2 + 1)^2}, \tag{11}$$

where $\Delta = (\Delta_1, \Delta_2, \ldots, \Delta_n)$.

We calculated the Gaussian curvature at the point that minimizes the L_2 norm of the quantization error and present them in Table 1. This means that the flat surface for 4 bits, shown in Fig. 2, is a generic property of the fine-grained quantization loss and not of the specific layer choice. Similarly, we conclude that coarser quantization generally has steeper curvature than more fine-grained quantization.

Moreover, the Hessian matrix provides additional information regarding the coupling between different layers. As could be expected, adjacent off-diagonal terms have higher values than distant elements, corresponding to higher dependencies between clipping parameters of adjacent layers (the Hessian matrix is presented in Fig. 3).

When the curvature is steep, even small changes in quantization step size may change the results drastically. To justify this hypothesis experimentally, in Fig. 4, we evaluate the accuracy of ResNet-50 for five different quantization steps. We choose quantization steps which minimize the L_p norm of the quantization error for different values of p:

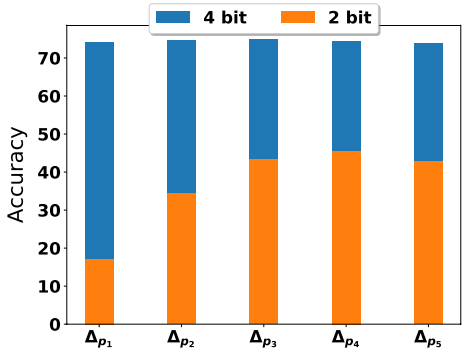


Fig. 4 Accuracy of ResNet-50 quantized to 2 and 4 bits, respectively. Quantization steps are chosen such that they minimize the L_p norm of the quantization error for different values of p.

$$e_p(\Delta) = (||Q_\Delta(X) - X||^p)^{1/p}. \tag{12}$$

While at 4-bit quantization, the accuracy is almost not affected by small changes in the quantization step size, at 2-bit quantization, the same changes shift the accuracy by more than 20%. Moreover, the best accuracy is obtained with a quantization step that minimizes $L_{3.5}$ and not the MSE, which corresponds to L_2 norm minimization.

3.3 Hessian of the loss function

To estimate dependencies between clipping parameters of different layers, we analyze the structure of the Hessian matrix of the loss function. The Hessian matrix contains the second-order partial derivatives of the loss $\mathcal{L}(\Delta)$, where $\mathbf{\Delta}$ is a vector of quantization steps:

$$H[\mathcal{L}]_{ij} = \frac{\partial \mathcal{L}}{\partial \Delta_i \partial \Delta_j}. \tag{13}$$

In the case of separable functions, the Hessian is a diagonal matrix. This means that the magnitude of the off-diagonal elements can be used as a measure of separability. In Fig. 3 we show the Hessian matrix of the loss function under quantization of 4 and 2 bits. As expected, higher dependencies between quantization steps emerge under more aggressive quantization.

4 Loss aware post-training quantization

In the previous section, we showed that the loss function $\mathcal{L}(\Delta)$, with respect to the quantization step size Δ, has a complex, non-separable landscape that is hard to optimize. Now we suggest a method to overcome this difficulty. Our optimization process involves three consecutive steps.

In the first phase, we find the quantization step Δ_p that minimizes the L_p norm of the quantization error of the individual layers for several different values of p. Then, we perform quadratic interpolation to approximate an optimum of the loss with respect to p. Finally, we jointly optimize the parameters of all layers acquired on the previous step by applying a gradient-free optimization method (Powell 1964). The pseudo-code of the whole algorithm is presented in Algorithm 1. Figure 8 provides the algorithm visualization.

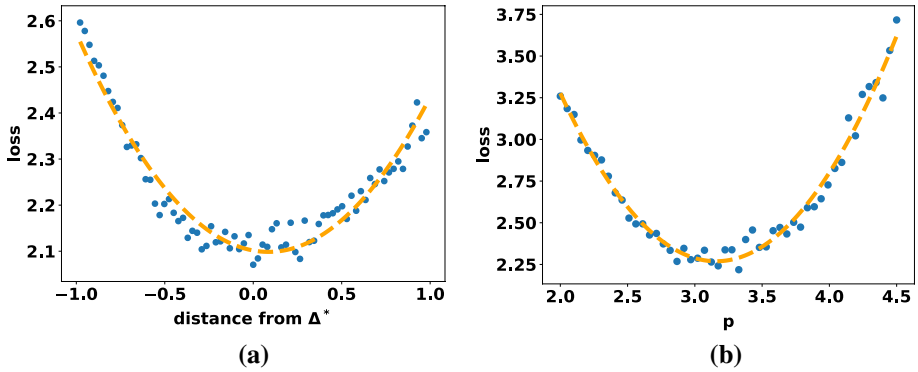


Fig. 5 Loss of ResNet-18 quantized with different quantization steps. The orange line shows quadratic interpolation: **a** with respect to the distance from optimal quantization step Δ^* and **b** on the trajectory defined by L_p norm minimization.

4.1 Layer-wise optimization

Our method starts by minimizing the L_p norm of the quantization error of weights and activations in each layer with respect to clipping values, defined in Eq. 12.

Given a real number $p > 0$, the set of optimal quantization steps $\Delta_p = \{\Delta_{1_p}, \Delta_{2_p}, \ldots, \Delta_{n_p}\}$, according to Eq. 12, minimizes the quantization error within each layer. In addition, optimizing the quantization error allows us to get Δ_p in the vicinity of the optimum Δ^* . Different values of p result in different quantization step sizes Δ_p, which are optimal under the L_p metric due to the trade-off between clipping and quantization error (Fig. 7).

4.2 Quadratic approximation

Assuming a quantization step size Δ in the vicinity of the optimal quantization step Δ^*, the loss function can be approximated with a Taylor series as follows:

$$\mathcal{L}(\Delta) - \mathcal{L}(\Delta^*) = (\Delta^* - \Delta)^\top \nabla \mathcal{L}(\Delta^*) + \tag{14}$$

$$+ \frac{1}{2}(\Delta^* - \Delta)^\top \mathbf{H}(\Delta^*)(\Delta^* - \Delta) + \tag{15}$$

$$+ \mathcal{O}(||\Delta^* - \Delta||^3), \tag{16}$$

where $\mathbf{H}(\Delta^*)$ is the Hessian matrix with respect to Δ. Since Δ^* is a minimum, the first derivative vanishes and we acquire a quadratic approximation of $\mathcal{L}$,

$$\Delta\mathcal{L} = \mathcal{L}(\Delta) - \mathcal{L}(\Delta^*) \approx \frac{1}{2}(\Delta^* - \Delta)^\top \mathbf{H}(\Delta^*)(\Delta^* - \Delta) \tag{17}$$

Our method exploits this quadratic property for optimization. Figures 5,6a demonstrates empirical evidence of such a quadratic relationship for ResNet-18 and ResNet-50 around the

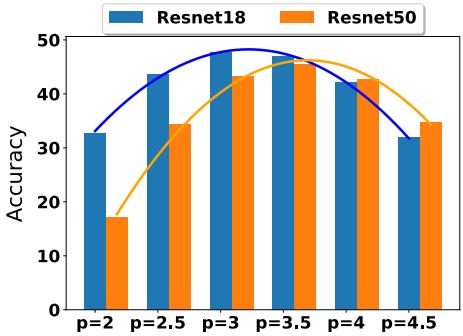


Fig. 6 Accuracy of ResNet-18 and ResNet-50 quantized to 2 bits with respect to layer wise optimization of different L_p norms. Accuracy with respect to p has approximately quadratic form.

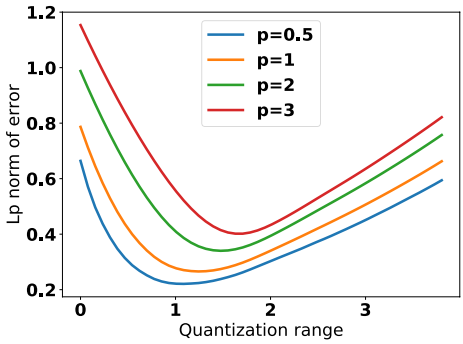


Fig. 7 The L_p norm of the quantization error generated by optimizing for different values of p according to Eq. 12. For different L_p metrics, the optimal quantization step is different.

Algorithm 1 LAPQ

Layer-wise optimization:
for $p = \{p_1, p_2, \ldots p_k\}$ **do**
  $\mathbf{C}_p \leftarrow \arg\min_{C} \left(\left\|Q_{\Delta,c}(x) - x\right\|^p\right)^{1/p}$ for every layer
end for
Quadratic approximation:
$\Delta_{p^*} \leftarrow \arg\min_{\Delta_p} \mathcal{L}(\Delta_p)$ using quadratic interpolation of Δ_p
Joint optimization (Powell's):
Starting point $t_0 \leftarrow \Delta_{p^*}$.
Initial direction vector $D = \{d_1, d_2, \ldots, d_N\}$.
while not converged **do**
  for $k = 1 \ldots N$ **do**
    $\lambda_k \leftarrow \arg\min_{\lambda} \mathcal{L}(t_{k-1} + \lambda_k d_k)$
    $t_k \leftarrow t_{k-1} + \lambda_k d_k$
  end for
  for $j = 1 \ldots N - 1$ **do**
    $d_j \leftarrow d_{j+1}$
  end for
  $d_N \leftarrow t_N - t_0$
  $\lambda_N \leftarrow \arg\min_{\lambda} \mathcal{L}(t_N + \lambda_N d_N)$
  $t_0 \leftarrow t_0 + \lambda_N d_N$
end while
$\Delta^* \leftarrow t_0$
return Δ^*

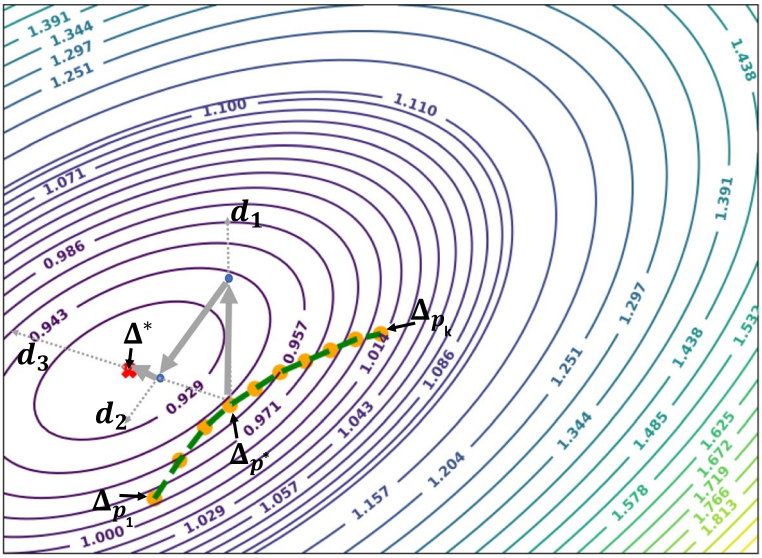


Fig. 8 Intuition about the LAPQ algorithm in two dimensions. The visualization shows the loss as a function of quantization ranges of two layers on synthetic data. Yellow dots correspond to the quantization step size $\{\Delta_p\}$, which minimizes the L_p norm of the quantization error. $\{\Delta_p\}$ build a trajectory in the vicinity of the minimum Δ^*. The optimal quantization step size on that trajectory Δ_{p^*} is used as a starting point for a joint optimization algorithm (Powell's). Vectors $\{d_1, d_2, d_3\}$ demonstrate the first iteration of Powell's method that approaches the global minimum Δ^* (Color figure online).

optimal quantization step Δ^* obtained by our method. First, we sample a few data points $\{\Delta_p\}$ to get number of samples of $\mathcal{L}(\Delta)$ (orange points in Fig. 8). Empirically we found that 10 points are enough for a good approximation as can be seen in Fig. 5. Then, we use the prior quadratic assumption to approximate the minimum of the $\mathcal{L}$ on that trajectory by fitting a quadratic function $f(p)$ to the sampled Δ_p. Finally, we minimize $f(p)$ and use the optimal quantization step size Δ_{p^*} as a starting point for a gradient-free joint optimization algorithm, such as Powell's method (Powell 1964), to minimize the loss and find Δ^*.

4.3 Joint optimization

By minimizing both the quantization error and the loss using quadratic interpolation, we get a better approximation of the global minimum Δ^*. Due to steep curvature of the minimum, however, for a low bitwidth quantization, even a small error in the value of Δ leads to performance degradation. Thus, we use a gradient-free joint optimization, specifically a Powell's method (Powell 1964), to further optimize Δ_{p^*}.

At every iteration, we optimize the set of parameters, initialized by Δ_{p^*}. Given a set of linear search directions $D = \{d_1, d_2, \ldots, d_N\}$, the new position Δ_{t+1} is expressed by the linear combination of the search directions as following $\Delta_t + \sum_i \lambda_i d_i$. The new displacement vector $\sum_i \lambda_i d_i$ becomes part of the search directions set, and the search vector, which contributed most to the new direction, is deleted from the search directions set. Powell's

Table 2 Comparison with other methods on ResNet-18, ResNet-50, and MobileNet V2. MMSE refers to minimization of MSE. W refers to the weights bitwidth and A to the activations bitwidth.

Model	W/A	Method	Accuracy(%)
ResNet-18	32 / 32	FP32	69.7
		LAPQ (Ours)	69.4
	8 / 8	DFQ Nagel et al. (2019)	**69.7**
		MMSE	69.0
		LAPQ (Ours)	**68.8**
		DUAL Kravchik et al. (2019)	68.38
		ACIQ Banner et al. (2018)	65.528
	8 / 4	MMSE	68.0
		LAPQ (Ours)	**66.3**
		ACIQ Banner et al. (2018)	52.476
	8 / 3	MMSE	63.3
		LAPQ (Ours)	**60.3**
		ACIQ Banner et al. (2018)	4.1
		KLD Migacz (2017)	31.937
	4 / 4	MMSE	43.6
		LAPQ (Ours)	**68.1**
	4 / 32	MMSE	68
		LAPQ (Ours)	**42.2**
	3 / 3	MMSE	1.1
ResNet-50	32 / 32	FP32	76.1
		LAPQ (Ours)	**74.8**
		DUAL Kravchik et al. (2019)	73.25
		ACIQ Banner et al. (2018)	68.92
	8 / 4	MMSE	74.0
		LAPQ (Ours)	**70.8**
		ACIQ Banner et al. (2018)	51.858
	8 / 3	MMSE	66.3
		LAPQ (Ours)	**70.0**
		ACIQ Banner et al. (2018)	3.3
		KLD Migacz (2017)	46.19
	4 / 4	MMSE	36.4
		LAPQ (Ours)	**45.5**
	3 / 3	MMSE	0.2
MobileNet v2	32 / 32	FP32	71.8
		LAPQ (Ours)	**71.6**
	8 / 8	DFQ Nagel et al. (2019)	71.2
		MMSE	71.6
		LAPQ (Ours)	**59.4**
	4 / 32	MMSE	47.6
		LAPQ (Ours)	**52.2**
	4 / 4	MMSE	26.0

Bold values indicate best result in each category

Table 3 Comparison with other methods on ResNet-101 and Inception-V3. MMSE refers to minimization of MSE. W refers to the weights bitwidth and A to the activations bitwidth.

Model	W/A	Method	Accuracy(%)
ResNet-101	32 / 32	FP32	77.3
		LAPQ (Ours)	73.6
		DUAL Kravchik et al. (2019)	**74.26**
		ACIQ Banner et al. (2018)	66.966
	8 / 4	MMSE	72.0
		LAPQ (Ours)	**65.7**
		ACIQ Banner et al. (2018)	41.46
	8 / 3	MMSE	56.7
		LAPQ (Ours)	**59.2**
		ACIQ Banner et al. (2018)	1.5
		KLD Migacz (2017)	49.948
	4 / 4	MMSE	9.8
		LAPQ (Ours)	**7.6**
	3 / 3	MMSE	0.1
Inception-V3	32 / 32	FP32	77.2
		LAPQ (Ours)	**75.2**
		DUAL Kravchik et al. (2019)	73.06
		ACIQ Banner et al. (2018)	66.42
	8 / 4	MMSE	74.3
		LAPQ (Ours)	**64.4**
		ACIQ Banner et al. (2018)	31.01
	8 / 3	MMSE	54.1
		LAPQ (Ours)	**38.6**
		ACIQ Banner et al. (2018)	0.5
		KLD Migacz (2017)	1.84
	4 / 4	MMSE	2.2
		LAPQ (Ours)	**4.5**
	8 / 2	MMSE	3

Bold values indicate best result in each category

Table 4 Hit rate of NCF-1B applying our method, LAPQ, and MMSE. W refers to the weights bitwidth and A to the activations bitwidth.

Model	W/A	Method	Hit rate(%)
NCF 1B	32/32	FP32	51.5
	32/8	LAPQ (Ours)	**51.2**
		MMSE	51.1
	8/32	LAPQ (Ours)	**51.4**
		MMSE	33.4
	8/8	LAPQ (Ours)	**51.0**
		MMSE	33.5

Bold values indicate best result in each category

Table 5 Accuracy of ResNet-18 with different initializations for the joint optimization. LW refers to only applying layer-wise optimization for $p = 2$. LW + QA refers to the proposed initialization that includes layer-wise optimization and quadratic approximation. W refers to the weights bitwidth and A to the activations bitwidth.

W / A	Method	Accuracy (%)	
		Initial	Joint
4 / 4	Random	0.1	7.8
	LW	43.6	57.8
	LW + QA	54.1	60.3
32 / 2	Random	0.1	0.1
	LW	33	50.3
	LW + QA	48.1	50.7

Table 6 Effect of applying bias correction on top of LAPQ for ResNet-18, ResNet-50 and MobileNet-V2. LAPQ significantly outperforms naive minimization of the mean square error (MSE). Notice the importance of bias correction on MobileNet-V2. W refers to the weights bitwidth and A to the activations bitwidth.

W	A	ResNet-18	ResNet-50	MobileNet-V2
LAPQ				
32	4	68.8%	74.8%	65.1%
32	2	51.6%	54.2%	1.5%
4	32	62.6%	69.9%	29.4%
4	4	58.5%	66.6%	21.3%
LAPQ + bias correction				
4	32	63.3%	71.8%	59.4%
4	4	60.3%	70.0%	52.2%
FP32		69.7%	76.1%	71.8%

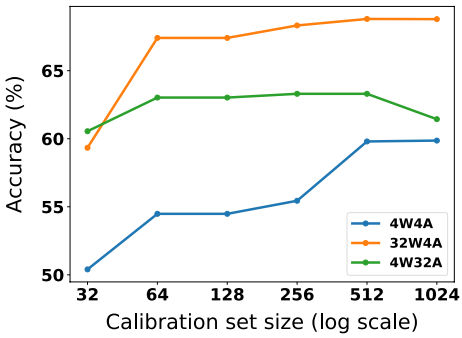


Fig. 9 Accuracy of ResNet-18 for different sized calibration sets at various quantization levels.

method (Powell 1964) is quadratically convergent (Brent 2013). In practice, in our experiments it converges in a few hundreds of iterations. For further details, see Algorithm 1.

Table 7 Comparison with other methods.

Model	W/A	Method	Accuracy(%)
ResNet-18	32 / 32	FP32	69.7
	8 / 2	LAPQ (Ours)	**51.6**
		ACIQ Banner et al. (2018)	7.07
ResNet-50	32 / 32	FP32	76.1
		LAPQ (Ours)	**54.2**
	8 / 2	ACIQBanner et al. (2018)	2.92
		LAPQ (Ours)	**71.8**
	4 / 32	OCS Zhao et al. (2019)	69.3
ResNet-101	32 / 32	FP32	77.3
		LAPQ (Ours)	**29.8**
	8 / 2	ACIQ Banner et al. (2018)	3.826
		LAPQ (Ours)	**66.5**
	4 / 32	MMSE	18

Bold values indicate best result in each category

5 Experimental results

In this section we conduct extensive evaluations and offer a comparison to prior art of the proposed method on two challenging benchmarks, image classification on ImageNet and a recommendation system on NCF-1B. In addition, we examine the impact of each part of the proposed method on the final accuracy. In all experiments we first calibrate the optimal clipping values on a small held-back calibration set using our method and then evaluate the validation set.

5.1 ImageNet

We evaluate our method on several CNN architectures on ImageNet. We select a calibration set of 512 random images for the optimization step. The size of the calibration set defines the trade-off between generalization and runtime (the analysis is given in 5.3). Following the convention (Baskin et al. 2018; Yang et al. 2019), we do not quantize the first and last layers.

Many successful methods of post-training quantization perform finer parameter assignment, such as group-wise (Mellempudi et al. 2017), channel-wise (Banner et al. 2018), pixel-wise (Fraser et al. 2018), or filter-wise (Kravchik et al. 2019) quantization, requiring special hardware support and additional computational resources. Finer parameter assignment appears to provide definite improvement, independently of the underlying methods used, but requires more complex and resource-demanding hardware. In contrast with those approaches, our method performs layer-wise quantization, which is simple to implement on any hardware that supports low precision integer operations. Thus, we do not include the methods with finer assignment in our comparison study. We apply bias correction, as proposed by Banner et al. (2018), on top of the proposed method. In Tables 2, 3 and 7 we compare our method with several other layer-wise quantization methods, as well as the minimal MSE baseline. In most cases, our method significantly outperforms all the competing methods, showing acceptable performance even for 4-bit quantization.

5.2 NCF-1B

In addition to the vision models, we evaluated our method on a recommendation system task, specifically on a Neural Collaborative Filtering (NCF) (He et al. 2017) model. We use mlperf[1]implementation to train the model on the MovieLens-1B dataset. Similarly to the ImageNet, the calibration set of 50k random user/item pairs is significantly smaller than both the training and validation sets.

In Table 4 we present results for the NCF-1B model compared to the MMSE method. Even at 8-bit quantization, NCF-1B suffers from significant degradation when using the naive MMSE method. In contrast, LAPQ achieves near baseline accuracy with 0.5% degradation from FP32 results.

5.3 Ablation study

Initialization The proposed method comprises of two steps: layer-wise optimization and quadratic approximation as the initialization for the joint optimization. In Table 5, we show the results for ResNet-18 under different initializations and their results after adding the joint optimization. LAPQ suggest a better initialization for the joint optimization.

Bias correction

Prior research (Banner et al. 2018; Finkelstein et al. 2019) has shown that CNNs are sensitive to quantization bias. To address this issue, we perform bias correction of the weights as proposed by Banner et al. (2018), which can easily be combined with LAPQ, in all our CNN experiments. In Table 6 we show the effect of adding bias correction to the proposed method and compare it with MMSE. We see that bias correction is especially important in compact models, such as MobileNet.

Calibration set size Calibration set size reflects the balance between the running time and the generalization. To determine the required size, we ran the proposed method on ResNet-18 for various calibration set sizes and different bitwidths. As shown in Fig. 9, a calibration set size of 512 is a good choice to balance this trade-off.

6 Conclusion

We have analyzed the loss function of quantized neural networks. At low precision, the function is non-separable, with steep curvature, which is unfavorable for existing post-training quantization methods. Accordingly, we have introduced Loss Aware Post-training Quantization (LAPQ), which jointly optimizes all quantization parameters by minimizing the loss function directly. We have shown that our method outperforms current post-training quantization methods. Also, our method does not require special hardware support such as channel-wise or filter-wise quantization. In some models, LAPQ is the first to show compatible performance in 4-bit post-training quantization regime with layer-wise quantization, almost achieving a the full-precision baseline accuracy.

Acknowledgements The research was funded by National Cyber Security Authority and the Hiroshi Fujiwara Technion Cyber Security Research Center.

[1] https://github.com/mlperf/training/tree/master/recommendation/pytorch.

References

Banner, R., Nahshan, Y., Hoffer, E., Soudry, D. (2018) Post-training 4-bit quantization of convolution networks for rapid-deployment. arXiv preprint arXiv:181005723http://arxiv.org/abs/1810.05723

Baskin, C., Liss, N., Chai, Y., Zheltonozhskii, E., Schwartz, E., Girayes, R., Mendelson, A., Bronstein, A. M. (2018) Nice: Noise injection and clamping estimation for neural network quantization. arXiv preprint arXiv:181000162http://arxiv.org/abs/1810.00162

Berger, T. (1972). Optimum quantizers and permutation codes. *IEEE Transactions on Information Theory, 18*(6), 759–765.

Bethge, J., Yang, H., Bornstein, M., Meinel, C., (2019) Back to simplicity: How to train accurate bnns from scratch? arXiv preprint arXiv:190608637http://arxiv.org/abs/1906.08637

Brent, R. P. (2013) Algorithms for minimization without derivatives. Courier Corporation.

Fang, J., Shafiee, A., Abdel-Aziz, H., Thorsley, D., Georgiadis, G., Hassoun, J. (2020) Near-lossless post-training quantization of deep neural networks via a piecewise linear approximation. arXiv preprint arXiv:200200104https://arxiv.org/abs/2002.00104

Faraone, J., Fraser, N., Blott, M., Leong, P. H. (2018) Syq: Learning symmetric quantization for efficient deep neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), http://openaccess.thecvf.com/content_cvpr_2018/html/Faraone_SYQ_Learning_Symmetric_CVPR_2018_paper.html

Fawzi, A., Moosavi-Dezfooli, S. M., Frossard, P., (2016) Robustness of classifiers: from adversarial to random noise. arXiv preprint arXiv:160808967https://arxiv.org/abs/1608.08967

Finkelstein, A., Almog, U., Grobman, M. (2019) Fighting quantization bias with bias. arXiv preprint arXiv:190603193http://arxiv.org/abs/1906.03193

Goldman, R. (2005). Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design, 22*(7), 632–658. https://doi.org/10.1016/j.cagd.2005.06.005.

Gong, J., Shen, H., Zhang, G., Liu, X., Li, S., Jin, G., Maheshwari, N., Fomenko, E., Segal, E. (2018) Highly efficient 8-bit low precision inference of convolutional neural networks with intelcaffe. In: Proceedings of the 1st on Reproducible Quality-Efficient Systems Tournament on Co-designing Pareto-efficient Deep Learning, ACM, New York, NY, USA, ReQuEST '18, 10.1145/3229762.3229763, http://doi.acm.org/10.1145/3229762.3229763

Gong, R., Liu, X., Jiang, S., Li, T., Hu, P., Lin, J., Yu, F., Yan, J. (2019) Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In: The IEEE International Conference on Computer Vision (ICCV), http://openaccess.thecvf.com/content_ICCV_2019/html/Gong_Differentiable_Soft_Quantization_Bridging_Full-Precision_and_Low-Bit_Neural_Networks_ICCV_2019_paper.html

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T. S. (2017) Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, WWW '17, pp 173–182, 10.1145/3038912.3052569, https://doi.org/10.1145/3038912.3052569

Hou, L., Kwok, J. T. (2018) Loss-aware weight quantization of deep networks. https://arxiv.org/abs/1802.08635

Hou, L., Yao, Q., Kwok, J. T. (2016) Loss-aware binarization of deep networks. arXiv preprint arXiv:161101600https://arxiv.org/abs/1611.01600

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., & Bengio, Y. (2017). Quantized neural networks: Training neural networks with low precision weights and activations. *J Mach Learn Res, 18*(1), 6869–6898.

Jin, Q., Yang, L., Liao, Z. (2019) Towards efficient training for neural network quantization. arXiv preprint arXiv:191210207https://arxiv.org/abs/1912.10207

Kim, H., Kim, K., Kim, J., Kim, J. J. (2020) Binaryduo: Reducing gradient mismatch in binary activation network by coupling binary activations. arXiv preprint arXiv:200206517https://arxiv.org/abs/2002.06517

Kravchik, E., Yang, F., Kisilev, P., Choukroun, Y. (2019) Low-bit quantization of neural networks for efficient inference. In: The IEEE International Conference on Computer Vision (ICCV) Workshops, http://openaccess.thecvf.com/content_ICCVW_2019/html/CEFRL/Kravchik_Low-bit_Quantization_of_Neural_Networks_for_Efficient_Inference_ICCVW_2019_paper.html

Krishnamoorthi, R. (2018) Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:180608342http://arxiv.org/abs/1806.08342

Lee, J. H., Ha, S., Choi, S., Lee, W.J., Lee, S. (2018) Quantization for rapid deployment of deep neural networks. arXiv preprint arXiv:181005488http://arxiv.org/abs/1810.05488

Lin, D.D., Talathi, S.S., Annapureddy, V.S. (2015) Fixed point quantization of deep convolutional networks. arXiv preprint arXiv:151106393https://arxiv.org/abs/1511.06393

Liu, C., Ding, W., Xia, X., Hu, Y., Zhang, B., Liu, J., Zhuang, B., Guo, G. (2019) Rectified binary convolutional networks for enhancing the performance of 1-bit DCNNs. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization, pp 854–860, 10.24963/ijcai.2019/120, https://doi.org/10.24963/ijcai.2019/120

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory, 28*(2), 129–137.

Max, J. (1960). Quantizing for minimum distortion. *IRE Transactions on Information Theory, 6*(1), 7–12.

Mellempudi, N., Kundu, A., Mudigere, D., Das, D., Kaul, B., Dubey, P. (2017) Ternary neural networks with fine-grained quantization. arXiv preprint arXiv:170501462http://arxiv.org/abs/1705.01462

Migacz, S. (2017) 8-bit inference with tensorrt. http://on-demand.gputechconf.com/gtc/2017/presentation/s7310-8-bit-inference-with-tensorrt.pdf

Nagel, M., Baalen, M.v., Blankevoort, T., Welling, M. (2019) Data-free quantization through weight equalization and bias correction. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1325–1334, http://openaccess.thecvf.com/content_ICCV_2019/html/Nagel_Data-Free_Quantization_Through_Weight_Equalization_and_Bias_Correction_ICCV_2019_paper.html

Nayak, P., Zhang, D., Chai, S. (2019) Bit efficient quantization for deep neural networks. arXiv preprint arXiv:191004877http://arxiv.org/abs/1910.04877

Peng, H., & Chen, S. (2019). Bdnn: Binary convolution neural networks for fast object detection. *Pattern Recognition Letters*. https://doi.org/10.1016/j.patrec.2019.03.026.

Powell, M. J. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal, 7*(2), 155.

Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., Hua, X.s. (2019) Quantization networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), http://openaccess.thecvf.com/content_CVPR_2019/html/Yang_Quantization_Networks_CVPR_2019_paper.html

Zhang, D., Yang, J., Ye, D., Hua, G. (2018) Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In: The European Conference on Computer Vision (ECCV), http://openaccess.thecvf.com/content_ECCV_2018/html/Dongqing_Zhang_Optimized_Quantization_for_ECCV_2018_paper.html

Zhao, R., Hu, Y., Dotzel, J., De Sa, C., Zhang, Z. (2019) Improving neural network quantization using outlier channel splitting. arXiv preprint arXiv:190109504https://arxiv.org/abs/1901.09504

Zhou, Y., Moosavi-Dezfooli, S.M., Cheung, N.M., Frossard, P. (2017) Adaptive quantization for deep neural network. arXiv preprint arXiv:171201048https://arxiv.org/abs/1712.01048