Post-Training Non-Uniform Quantization for Convolutional Neural Networks

Anonymous submission

Abstract

Despite the success of CNN models on a variety of Image classification and segmentation tasks, their extensive computational and storage demands pose considerable challenges for real-world deployment on resource-constrained devices. Quantization is one technique that aims to alleviate these large storage requirements and speed up the inference process by reducing the precision of model parameters to lower-bit representations. In this paper, we introduce a novel post-training quantization method for model weights. Our method finds optimal clipping thresholds and scaling factors along with mathematical guarantees that our method minimizes quantization noise. Empirical results on Real World Datasets demonstrate that our quantization scheme significantly reduces model size and computational requirements while preserving model accuracy.

Introduction

Deep neural networks (DNNs) are fundamental to modern artificial intelligence, driving advancements in fields like computer vision, natural language processing, and autonomous systems. However, the deployment of DNNs in real-world applications is often limited by their significant computational and storage demands, primarily due to the vast number of model parameters. This is especially true for Convolutional Neural Networks (CNNs) used in image classification and segmentation, which have numerous layers, leading to substantial storage requirements. These demands make it challenging to deploy full models on devices with limited resources.

A practical solution to address the growing complexity of DNNs is model compression, which reduces the size and computational demands of these networks with minimal impact on performance. By compressing a model, we create a smaller, less complex version that maintains key performance metrics like accuracy and loss. The advantages of model compression include reduced model size, lower computational complexity, decreased memory usage, improved energy efficiency, and faster inference times. This enables efficient deployment of DNNs on resource-constrained devices and can, in some cases, improve generalization by mitigating overfitting.

Model compression can be achieved through techniques such as pruning (removing redundant parameters), knowledge distillation (transferring knowledge from a larger model to a smaller one), and matrix factorization (decomposing weight matrices into low-rank approximations). In our paper, we focus on quantization, a method that reduces the precision of numerical values in model parameters. Quantization can be applied to both the weights and activations of a DNN. It involves converting high-precision weights and biases (e.g., 32-bit floating point) to lower precision (e.g., 8-bit integers), and quantizing intermediate activations during inference. This enables low-precision computation, leveraging fixed-bit arithmetic for efficient hardware implementation, making it feasible to deploy models on edge devices. Recent research has explored enhancing quantization by combining it with other compression techniques, further improving its effectiveness (Ko et al. 2021; Wang, Lu, and Blankevoort 2020).

Quantization Aware Training (QAT) incorporates quantization directly into the training process, allowing the model to "learn" the effects of quantization. However, this is time-consuming and difficult to implement (Gholami et al. 2022). Post-Training Quantization (PTQ) adjusts the weights of the quantized model without the need for further training (Banner, Nahshan, and Soudry 2019). PTQ can be applied to both weights and activations.

Uniform quantization divides the parameter value range into equal-sized intervals, offering a simple and efficient method. However, it may be suboptimal for parameters with non-uniform distributions. Non-uniform quantization, in contrast, adjusts interval sizes based on the parameter distribution, potentially offering a better fit for data with long tails, which is common for weights and activations. This approach can improve accuracy by assigning more precision to frequently occurring values. However, it is more complex, may lack compatibility with certain hardware, and could increase computational costs, limiting its portability and efficiency.

In this paper, we propose a novel post-training quantization approach that uniformly quantizes model weights and biases to any user-defined number of bits. Since model parameters generally have bell-shaped distribution, most commonly Laplace or Gaussian distribution (Baskin et al. 2021; Li, Dong, and Wang 2019; Banner, Nahshan, and Soudry 2019), we derive an analytical solution for uniform quantization. We determine the optimal clipping range, quanti-

zation intervals, and quantization levels, ensuring minimal quantization error.

The key contributions of this work are as follows:

- Demonstrating that weights and activations predominantly follow either a Gaussian or Laplacian distribution.
- Introducing a numerical quantization solution for non-uniform distributions with analytical proof.
- Showing that our quantization scheme outperforms existing methods in terms of mean square error (MSE) performance.
- Providing evidence that well-known models quantized using our scheme exhibit minimal loss of accuracy.

Related Work

This section outlines some related work on model compression with a focus on quantization. The challenge of reducing model size, improving inference latency, and minimizing power consumption in deep neural networks is addressed (Gholami et al. 2022) through knowledge distillation methods (Gou et al. 2021) and matrix factorization (Ko et al. 2021; Gao et al. 2021, 2022), and pruning (Chen et al. 2024).

(Cho, Adya, and Naik 2024) introduces an efficient train-time pruning scheme that uses a dynamic function of weights to generate soft pruning masks without additional parameters. In (El Halabi, Srinivas, and Lacoste-Julien 2022), authors introduce a data-efficient structured pruning method based on submodular optimization. This ensures minimal input changes in the next layer and provides an exponentially decreasing error with pruning size. Hardware-aware latency Pruning (Shen et al. 2022) formulates structural pruning as a global resource allocation optimization problem, aiming to maximize accuracy while keeping latency within a predefined budget on the target device.

Network quantization compresses and accelerates networks by mapping floating point weights to lower-bit integers. Activations are only known when the network is run on training data, activation quantization can only be done using calibration data. Selection of calibration training data is critical. Williams et al. (Williams and Aletras 2023) demonstrate how random sampling to get calibration data is not optimal. Chen et al. (Chen, Yang, and Chen 2022) shows the need to prevent class-imbalanced calibration data. Zhang et al. (Zhang et al. 2023) propose a method to get a truly representative sample to be used to get correct statistics.

Quantization Aware Training (QAT) incorporates quantization effects during training, allowing models to adapt to lower precision (Ahmadian et al. 2024). Post Training Quantization (PTQ), on the other hand, adjusts quantized model weights post-training, eliminating the need for additional training cycles and access to training data.

(Sun et al. 2022) proposed entropy-driven mixed-precision quantization, which assigns optimal bit-widths to each layer based on weight entropy, facilitating deployment on IoT devices. In (Tukan, Mualem, and Maalouf 2022), the authors propose a robust, data-independent framework for computing coresets with mild model assumptions, using Löwner ellipsoid and Caratheodory theorem to assess neuron importance. The method works across various networks and datasets, achieving a 62% compression rate on ResNet50 with only a 1.09% accuracy drop on ImageNet. In (Ma et al. 2023), mixed precision quantization is proposed, which utilizes hardware's multiple bit-width arithmetic operations. Kuzmin et al. (Kuzmin et al. 2022) showed how having 8 bits does not necessitate using integers and that FP8 can perform better than INT8.

Uniform quantization, a straightforward approach, involves partitioning the range of parameter values into equal-sized intervals (Banner, Nahshan, and Soudry 2019). Neural Network distributions tend to have a bell-curved distribution around the mean (Baskin et al. 2021). Several papers take advantage of this and allocate bins accordingly. Banner et al. (Banner, Nahshan, and Soudry 2019) introduce Analytical Clipping for Integer Quantization (ACIQ), which limits activation value ranges to reduce rounding error by minimizing mean-square-error. They also propose a bit allocation policy to optimize bit-width for each channel, formulated as an optimization problem to minimize overall mean-squared error under quantization constraints. Their approach achieves accuracy that is just a few percent less than the state-of-the-art baseline across a wide range of convolutional models. Zhao et al. (Zhao et al. 2019) expand on (Banner, Nahshan, and Soudry 2019) by artificially shifting outliers closer to the center of the distribution.

Nonuniform quantization, which adapts interval sizes to the parameter distribution, generally outperforms uniform quantization in compressing neural networks due to its better representational capacity (Liu et al. 2022).

(Li, Dong, and Wang 2019) proposed a method to discretize parameters in a non-uniform manner to additive powers of two, aligning more closely with their distribution and improving efficiency.

Our proposed method contributes to this work by providing an analytical framework for non-uniform quantization that does not depend on training data. This independence is particularly advantageous for scenarios where data availability is limited or privacy concerns restrict the use of actual data for model compression.

Proposed Method

Our method first uses the Kolmogorov-Smirnov test to evaluate whether the given model has weights that more closely resemble a Normal Distribution or a Gaussian Distribution. After finding the best fitting distribution we use it to calculate the optimal quantization levels and intervals for each layer of the network. Finding these intervals is computationally intensive as it involves solving a system of linear equations, so we use an iterative approach instead.

Preliminaries

Let X be a high-precision tensor-valued random variable with a probability density function $f(x)$. Assuming target bit-width M, we clip the range of values in the tensor and quantize the values in the clipped range to 2^M discrete values. We give an analytical expression for expected clipping and quantization error. We find the optimal clipping range,

quantization intervals, and quantization level for each interval that minimize the sum of clipping and quantization errors for the cases when $f(X)$ is Gaussian or Laplace distribution.

We formulate the quantization problem as an optimization task with the objective of minimizing Mean Square Error (MSE).

Clipping Error For $x \in \mathbb{R}$ and constants $\alpha < \beta$, define the clip function

$$\text{clip}(x, \alpha, \beta) = \begin{cases} \alpha & \text{if } x < \alpha, \\ x & \text{if } \alpha \leq x \leq \beta, \\ \beta & \text{if } x > \beta. \end{cases}$$

α, and all weights above β would become β.

The clipping error C_E, the mean-squared error resulting from clipping, is given as:

$$C_E = C_E(\alpha, \beta) = \int_{-\infty}^{\infty} \big(x - clip(x, \alpha, \beta)\big)^2 f(x)\, dx$$

$$= \int_{-\infty}^{\alpha} (x - \alpha)^2 f(x)\, dx + \int_{\beta}^{+\infty} (x - \beta)^2 f(x)\, dx \quad (1)$$

Using the knowledge of the weight distribution, we propose a method to calculate the quantization intervals (x_i) and levels (y_i) that minimize the MSE. We introduce a clipping range with limits α and β to handle weights that fall outside the typical distribution range, reducing the clipping error.

Quantization Error Let X' be the clipped tensor, i.e., X clipped to the range $[\alpha, \beta]$. The range $[\alpha, \beta]$ is partitioned into $k = 2^M$ intervals $[x_0, x_1), [x_1, x_2), \ldots, (x_{k-1}, x_k]$, where $x_0 = \alpha$ and $x_k = \beta$. For $i = 1, \cdots, k$, all the values in X' that fall in $[x_{i-1}, x_i)$ are rounded to $y_i \in [x_{i-1}, x_i)$. The quantization error in the range $[\alpha, \beta]$ is given as

$$Q_E = \sum_{i=1}^{2^M} \int_{x_{i-1}}^{x_i} (x - y_i)^2 f(x)\, dx \quad (2)$$

Optimization Framework

Our quantization method involves determining optimal clipping thresholds and scaling factors that minimize quantization noise. Our goal is to find the optimal clipping range ($[\alpha, \beta]$), quantization intervals (given by $x_1, \ldots, x_{k-1}$), and quantization levels (given by $y_1, \ldots, y_k$). The expected mean square error between X and its clipped and quantized version $Q(X)$ is given as follows:

$$D = E[(X - Q(x))^2]$$

$$= \int_{-\infty}^{\alpha} (x - \alpha)^2 f(x)\, dx + \int_{\beta}^{+\infty} (x - \beta)^2 f(x)\, dx$$

$$+ \sum_{i=1}^{2^M} \int_{x_{i-1}}^{x_i} (x - y_i)^2 f(x)\, dx \quad (3)$$

The mathematical model for non-uniform quantization is based on the probability distribution of the data. Given a set of data points X with a probability density function $f(x)$, the goal is to find a quantization function $Q(x)$ that minimizes the expected distortion D, defined as:

$$D = E[(X - Q(X))^2] \quad (4)$$

The quantization function $Q(x)$ maps the data points to a set of quantization levels $\{y_i\}$, with the intervals $\{I_i = [x_{i-1}, x_i]\}$ determined by the distribution $f(x)$. The optimization problem can be formulated as:

$$\arg \min_{\{y_i\}, \{I_i\}} E[(X - Q(X))^2] \quad (5)$$

Optimal Quantization Levels for Known Quantization Intervals Assuming that quantization intervals $(x_i, i = 1, 2, ..., K)$ are known, we need to determine the quantization levels $(y_i, i = 1, 2, ..., K)$.

We minimize the D (3) with respect to y_i. Since D is a sum of integrals, and only one of those integrals includes y_i (the one with limits x_{i-1} and x_i), the other terms differentiate to 0.

$$\frac{\delta Q_E}{\delta y_i} = -2 \int_{x_{i-1}}^{x_i} (x - y_i) f(x)\, dx = 0 \quad (6)$$

We open up this integral and evaluate this in terms of y_i:

$$\int_{x_{i-1}}^{x_i} x f(x)\, dx - \int_{x_{i-1}}^{x_i} y_i f(x)\, dx = 0$$

$$\implies \int_{x_{i-1}}^{x_i} x f(x)\, dx = y_i \int_{x_{i-1}}^{x_i} f(x)\, dx$$

$$\implies y_i = \frac{\int_{x_{i-1}}^{x_i} x f(x)\, dx}{\int_{x_{i-1}}^{x_i} f(x)\, dx} \quad (7)$$

In other words, we get that y_i should be the conditional expected value of the weights within the interval $[x_{i-1}, x_i)$.

$$y_i = E[X | x_{i-1} < X < x_i] \quad (8)$$

We call this our Conditional Mean Criterion. This means that all values that fall in a certain interval should be quantized to their expected value in that interval. Many quantization methods like ACIQ (Banner, Nahshan, and Soudry 2019) resort to simply quantizing the values to the midpoint of the relevant interval, which is only correct should the distribution be uniform.

Optimal Quantization Intervals for Known Quantization Levels Assuming that quantization levels $(y_i, i = 1, 2, ..., K)$ are known, we need to know how to allocate the quantization intervals $(x_i, i = 1, 2, ..., K)$. We minimize the D (3) with respect to x_i.

Since D is a sum of integrals, and only two of those integrals include x_i as a limit, the rest is differentiated to 0.

$$\frac{\delta Q_E}{\delta x_i}$$

$$= \frac{\delta}{\delta x_i} \int_{x_{i-1}}^{x_i} (x - y_i)^2 f(x)\, dx + \frac{\delta}{\delta x_i} \int_{x_i}^{x_{i+1}} (x - y_{i+1})^2 f(x)\, dx$$

$$= 0 \quad (9)$$

Since we are differentiating an integral with respect to its limits, we use the Leibniz Rule 10:

$$\frac{d}{dx}\int_a^x f(t)\,dt = f(x)\cdot\frac{dx}{dx} = f(x) \tag{10}$$

Using the Leibniz Rule 10 to simplify 9:

$$\frac{\delta Q_E}{\delta x_i} = (x_i - y_i)^2 f(x_i) - (x_i - y_{i+1})^2 f(x_i) = 0 \tag{11}$$

Evaluating this in terms of x_i gives us our Midpoint Criterion. This shows us that the boundary should be the midpoint of the quantization levels, regardless of what $f(x)$ is.

$$(x_i - y_i)^2 = (x_i - y_{i+1})^2 \tag{12}$$

$$x_i = \frac{y_i + y_{i+1}}{2}, i = 1, 2, ..., K-1 \tag{13}$$

A simple calculation shows that regardless of the distribution of X, the above quantization interval's boundaries minimize the D for the given y_i. This seems to contradict the Conditional Mean Criterion, however, we can get a system of equation using these two criteria.

System of Simultaneous Equations Given the general forms of the two criterion apply for all values of $i = 1, 2, ..., K$, we get a system of $2K-1$ equations. These can be solved to get the $2K-1$ unknowns, which are the quantization levels and boundaries.

Solving this large system of equations has high computational complexity, hence we choose to compute the optimum quantizer in an iterative fashion. We start by arbitrarily setting the quantization bins in a uniform fashion. We choose to start with $x_i = 2i/M$. We then iterate between computing new quantization levels according to Equation X and new quantization bin edges according to Equation Y. After going back and forth between these two equations several times, the results converge toward a final optimum quantizer.

Iterative approach to solving the system Due to the high computational cost of solving a large system of equations directly, we employ an iterative approach to find the optimal quantizer. 1.

We begin by initializing the quantization bins uniformly across the range. A common starting point is to set each bin center, denoted by x_i, to $2i/K$, where i ranges from 1 to K (the number of bins). The optimization process then iterates between two steps:

1. Update Quantization Levels: We calculate new, improved quantization levels based on Equation 8.

2. Update Bin Edges: We compute new and refined bin boundaries using Equation 13.

By repeatedly going back and forth between these steps, the initial estimate progressively converges towards the optimal quantizer configuration.

As we iteratively update the quantization levels and boundaries, we obtain increasingly smaller intervals at the center, resulting in an increasing number of intervals in close to the center.

Algorithm 1: Iterative Quantizer

Input: Data points $\{x_n\}$, number of bins K
Initialize: Quantization levels $y_i = \frac{2i}{K}$ for $i = 1, \ldots, K$
while Stopping criterion not met **do**
  Update Quantization Levels:
  for $i = 1$ **to** K **do**
    $y_i \leftarrow$ UpdateLevel(y_i) {Using Eq. 8}
  end for
  Update Quantization Boundaries:
  for $i = 1$ **to** $K-1$ **do**
    $x_i = \frac{y_i + y_{i+1}}{2}$ {Using Eq. 13}
  end for
  $x_K = \infty$
end while
Output: Optimal quantization levels $\{y_i\}$ and boundaries $\{x_i\}$

Analytical Solutions for Bell-Shaped Distributions

We consider two common bell-shaped distributions: Gaussian and Laplace (Banner, Nahshan, and Soudry 2019). The intervals $\{I_i = [x_{i-1}, x_i)\}$ are chosen to reflect the concentration of data points, with more intervals placed where the data is denser.

Laplace Distribution The Laplace distribution is particularly well-suited for modeling data that exhibit a sharp peak at the mean, with heavy tails decaying exponentially on both sides. This characteristic makes it a compelling choice for quantizing neural network parameters, which often follow a similar distribution.

The Laplace distribution is defined by its location parameter μ, which denotes the peak of the distribution, and its diversity parameter b, which controls the decay rate of the tails. The probability density function (PDF) of the Laplace distribution is given by:

$$f(x|\mu, b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right) \tag{14}$$

The sharp peak at the mean and the heavy tails are advantageous for quantization as they allow for a more concentrated set of intervals around the mean, where the data points are most dense, and larger intervals in the tails, where data points are sparse.

To derive the analytical solution for quantization under the Laplace distribution, we seek to minimize the expected value of the quantization error, which can be expressed as:

$$E[Q_E] = \int_{-\infty}^{\infty}(x - Q(x))^2 f(x|\mu, b)dx \tag{15}$$

where $Q(x)$ is the quantization function mapping real numbers to a discrete set of quantization levels. By leveraging the symmetry and the exponential decay properties of the Laplace distribution, we can determine the optimal quantization levels and intervals that minimize this error. The resulting quantization scheme is non-uniform, with smaller intervals near the mean and progressively larger intervals as we

move away from the mean, reflecting the density of the distribution. The quantization level y_i with the standard Laplacian parameters is as follows:

$$y_i = \frac{\int_{x_{i-1}}^{x_i} xe^{-|x|}/2\,dx}{\int_{x_{i-1}}^{x_i} e^{-|x|}/2\,dx} \tag{16}$$

$$= \frac{(x_{i-1}+1)e^{-x_{i-1}} - (x_i+1)e^{-x_i}}{e^{-x_{i-1}} - e^{-x_i}}$$

We can use our iterative algorithm to calculate quantization boundaries and intervals and save them for later use. The Conditional Mean Criterion step of the algorithm will use Equation 16.

Gaussian Distribution The Gaussian distribution, often referred to as the normal distribution, is ubiquitous in statistical analysis due to its natural occurrence in many physical, biological, and social phenomena. In the context of neural networks, the Gaussian distribution is frequently observed in the distribution of weights, especially in layers that have been subject to regularization techniques.

The Gaussian distribution is characterized by its mean μ and variance σ^2, which describe the center and the spread of the data points, respectively. The probability density function (PDF) of the Gaussian distribution is given by:

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{17}$$

For quantization, these characteristics imply that most data points are concentrated around the mean, with the frequency of occurrence decreasing symmetrically as one moves away from the mean. This property is advantageous for quantization as it allows for finer quantization intervals near the mean where the data points are dense and coarser intervals in the tails where they are sparse.

We aim to minimize the expected quantization error to derive the analytical solution for quantization under the Gaussian distribution. The optimization problem can be formulated as:

$$E[Q_E] = \int_{-\infty}^{\infty} (x - Q(x))^2 f(x|\mu,\sigma^2)dx \tag{18}$$

where $Q(x)$ is the quantization function. By exploiting the properties of the Gaussian distribution, we can calculate the optimal quantization levels that minimize the expected error. The resulting quantization scheme will have non-uniformly spaced intervals, with a higher density of intervals near the mean and fewer intervals as the distance from the mean increases, thus accommodating the bell-shaped nature of the Gaussian distribution. The quantization level y_i is computed as follows:

$$y_i = \sqrt{\frac{2}{\pi}} \cdot \frac{e^{-x_{i-1}^2/2} - e^{-x_i^2/2}}{erf(x_i/\sqrt{2}) - erf(x_{i-1}/\sqrt{2})} \tag{19}$$

Where erf refers to the commonly used mathematical error function.

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{20}$$

We can use our iterative algorithm to calculate quantization boundaries and intervals and save them for later use. The Conditional Mean Criterion step of the algorithm will use Equation 19.

Dealing with Non Laplace or Gaussian Distributions

Weights in most models typically follow either a Laplacian or Normal distribution. Baskin et al. (Baskin et al. 2021) used Shapiro-Wilk statistics to demonstrate the similarity of ResNet-18 weights to a Normal distribution. We reproduce their results and additionally compare the weights to a Laplacian distribution, as shown in Figure 1.

To compare these distributions, we use the Kolmogorov-Smirnov (K-S) test, which measures the maximum difference between the cumulative distribution functions (CDFs) of the actual data and the fitted distributions. We first fit both normal and Laplacian distributions to the data using maximum likelihood estimation to obtain the distribution parameters. A lower K-S statistic indicates a better fit, allowing us to determine whether the weights and activations align more closely with a Gaussian or Laplacian distribution.

Based on the best-fit distribution, we adjust the quantization boundaries and intervals using scale and offset before applying quantization. This process is analogous to the bias correction technique used in previous works (Banner, Nahshan, and Soudry 2019).

Quantization for Activations

Quantizing activations is more technical than quantizing weights. This is because activations depend on input data. Due to computational limitations, only a subset of the data is studied. Recent work has shown the importance of choosing a representative sample (Zhang et al. 2023), (Williams and Aletras 2023), (Chen, Yang, and Chen 2022).

Knowledge about the activation function is also critical. For example, the ReLU function (common in CNNs) only outputs positive numbers. This also means the data may no longer be symmetrical. The Fused-ReLU technnique (Banner, Nahshan, and Soudry 2019) accounts for this by allocating all the bits to the positive side.

For the purpose of this paper, we have chosen to quantize the weights only.

Experiments

In this section, we provide details of the datasets and experimental setup, followed by a comprehensive evaluation of the proposed approach and its comparison with the related methods.

We present an algorithm that iteratively adjusts the quantization intervals and levels based on the conditional mean criterion and the midpoint criterion as given in equations 16 and 19. These intervals and levels can all be calculated using the standard parameters. They can later be transformed according to the data's actual mean and standard deviation.

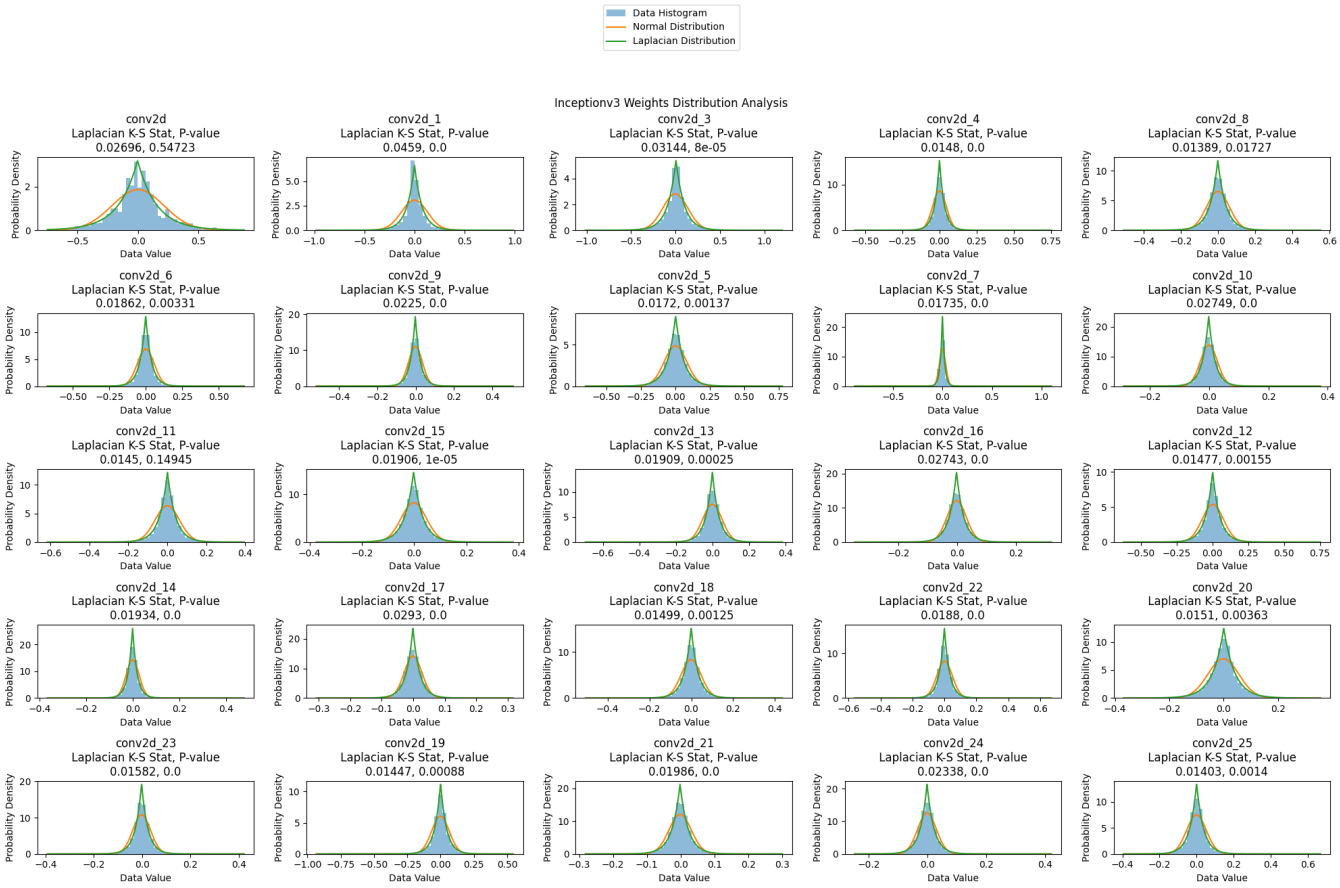


Figure 1: Weight distributions in Inceptionv3

Datasets Description

We compare each method on the popular datasets: ImageNet (ILSVRC12) (Deng et al. 2009), CIFAR10 and CIFAR100 (Krizhevsky, Hinton et al. 2009).

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. CIFAR-100 is just like the CIFAR-10, except it has 100 classes containing 600 images each.

The ImageNet dataset offers 1,000 categories for classification, with 1.2 million training images and 150,000 validation images.

For experiments, we perform quantization on ResNet-50, ResNet-18 and ResNet-20 (He et al. 2016) which are pretrained on ImageNet.

All of the pre-trained model implementations and weights were provided by Pytorchcv1.

Comparison with other quantization methods

In our research, we focus on isolating and evaluating individual weight quantization methods to analyze their direct impact on model performance and Mean Squared Error (MSE).

We use the original FP32 model as our baseline for comparison. For ResNet18, ResNet20, and ResNet50 on CIFAR-10 and CIFAR-100, the baseline models are fine-tuned for 20 epochs prior to quantization. For ResNet18, we also evaluate the effect of directly quantizing the pretrained model without additional fine-tuning.

We compare our method to several closely-related methods: Uniform Quantization like ACIQ (Banner, Nahshan, and Soudry 2019) and Non-uniform Quantization like APoT (Li, Dong, and Wang 2019).

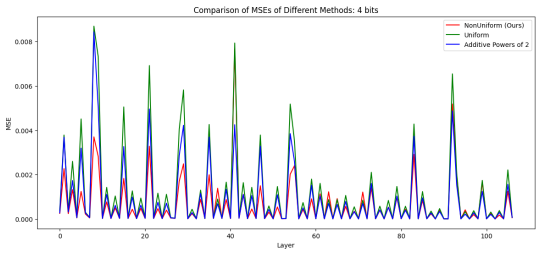
For fairness, all the models are compiled in the same way with an adam optimizer and a sparse categorical crossentropy loss. We vary the bitwidths from 4 to 8 bits.

Evaluation Metrics

Accuracy The models that are pretrained on the ImageNet dataset are evaluated on the same task. The model weights are updated and their Top-1 validation accuracy is reported.

One problem with this measure is that Model Accuracy might improve after quantization due to the introduction of regularization effects, where the reduced precision helps prevent overfitting by smoothing out minor fluctuations in the data. This can sometimes lead to better generalization on unseen data. However, accuracy alone is not the best measure of the effectiveness of quantization.

Mean-Squared Error In the following figure, we show the Mean Squared Error (MSE) of various quantization methods (Our method, ACIQ, APot) as we go deeper into ResNet50. It can be observed that at the majority of layers, our method has a lower MSE. The results were similar across models like MobileNet-V2 (Sandler et al. 2018), VGG16 (Simonyan and Zisserman 2014), Inceptionv3 (Szegedy et al. 2016).



Results on ImageNet Mini The comparison results based on ImageNet are provided in Table 1

DNN Model	FP32 Top-1 Acc	Precision	Top 1 Accuracy on ImageNet		
			Ours	Uniform	APoT
ResNet18	69.97	4 bits	46.16	44.18	47.82
		5 bits	55.88	52.03	52.13
		6 bits	59.14	52.49	51.98
		7 bits	61.41	52.64	52.66
		8 bits	62.53	52.41	52.83

Table 1: ImageNet Results

Results on CIFAR10 and CIFAR100 Our method almost always performed better than other methods for the CIFAR-100 task. For CIFAR-10, the results were the same except for the 4-bit variations.

The comparison results based on CIFAR10 and CIFAR100 are provided in Table 2 and Table 3, respectively.

DNN Model	FP32 Top-1 Acc	Precision	Top 1 Accuracy on CIFAR-10		
			Ours	Uniform	APoT
ResNet50	84.12	4 bits	64.84	66.14	68.28
		5 bits	69.97	67.04	71.06
		6 bits	66.87	66.91	69.95
		7 bits	70.51	67.51	68.48
		8 bits	72.18	67.62	69.31
ResNet18	94.54	4 bits	85.64	86.97	86.72
		5 bits	87.69	87.42	86.81
		6 bits	87.64	87.28	87.30
		7 bits	88.04	87.16	87.05
		8 bits	87.82	87.24	87.34
ResNet20	82.44	4 bits	79.42	79.87	80.20
		5 bits	81.96	80.96	81.31
		6 bits	82.43	82.35	82.16
		7 bits	82.53	82.21	82.32
		8 bits	82.46	82.38	82.48

Table 2: CIFAR-10 Results

DNN Model	FP32 Top-1 Acc	Precision	Top 1 Accuracy on CIFAR-100		
			Ours	Uniform	APoT
ResNet50	64.07	4 bits	27.45	30.51	30.77
		5 bits	43.49	28.76	29.34
		6 bits	45.02	29.74	30.01
		7 bits	46.62	29.22	29.46
		8 bits	46.40	29.33	29.72
ResNet18	65.78	4 bits	61.55	61.92	62.89
		5 bits	63.43	63.04	63.36
		6 bits	63.53	63.75	63.66
		7 bits	63.66	63.61	63.75
		8 bits	63.45	63.55	63.93
ResNet20	51.28	4 bits	47.99	43.29	41.16
		5 bits	50.19	48.67	48.46
		6 bits	50.52	50.44	49.88
		7 bits	51.33	50.94	50.82
		8 bits	51.12	51.40	50.74

Table 3: CIFAR-100 Results

Conclusion

Optimal model compression is one of the focus areas in deep learning to optimize the deployment of complex models. In this paper, we have introduced a novel post-training quantization approach that reduces the computational complexity and storage requirement of deep neural networks (DNNs) without compromising their performance. Our proposed method uses an analytical solution tailored to handle the bell-shaped distribution typically observed in model weights and activations. By determining optimal clipping ranges, quantization intervals, and levels, our approach ensures minimal quantization error, thereby preserving model accuracy.

Our empirical evaluations demonstrate that the proposed quantization scheme effectively reduces model size and computational requirements while maintaining performance, making it particularly suitable for deployment in resource-constrained environments such as mobile devices and embedded systems.

Future work could focus on optimizing the quantization process through adaptive techniques and integrating our method with other compression strategies to improve performance. We will also assess the impact of our quantization scheme on various models and tasks to validate its effectiveness. Additionally, future directions include calibrating activation quantization using representative training samples and varying bit allocation across layers based on their Mean Squared Error (MSE), assigning more bits to layers with higher MSE. These extensions remain as potential avenues for further research.

References

Ahmadian, A.; Dash, S.; Chen, H.; Venkitesh, B.; Gou, Z. S.; Blunsom, P.; Üstün, A.; and Hooker, S. 2024. Intriguing properties of quantization at scale. *Advances in Neural Information Processing Systems*, 36.

Banner, R.; Nahshan, Y.; and Soudry, D. 2019. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Advances in Neural Information Processing Systems*, 32.

Baskin, C.; Liss, N.; Schwartz, E.; Zheltonozhskii, E.; Giryes, R.; Bronstein, A. M.; and Mendelson, A. 2021. Uniq: Uniform noise injection for non-uniform quantization of neural networks. *ACM Transactions on Computer Systems (TOCS)*, 37(1-4): 1–15.

Chen, C.; Fu, Z.; Liu, K.; Chen, Z.; Tao, M.; and Ye, J. 2024. Optimal Parameter and Neuron Pruning for Out-of-Distribution Detection. *Advances in Neural Information Processing Systems*, 36.

Chen, T.-A.; Yang, D.-N.; and Chen, M.-S. 2022. Climbq: Class imbalanced quantization enabling robustness on efficient inferences. *Advances in Neural Information Processing Systems*, 35: 37134–37145.

Cho, M.; Adya, S.; and Naik, D. 2024. PDP: Parameter-free Differentiable Pruning is All You Need. *Advances in Neural Information Processing Systems*, 36.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

El Halabi, M.; Srinivas, S.; and Lacoste-Julien, S. 2022. Data-efficient structured pruning via submodular optimization. *Advances in Neural Information Processing Systems*, 35: 36613–36626.

Gao, Y.; Zhang, Z.; Hong, R.; Zhang, H.; Fan, J.; and Yan, S. 2022. Towards Feature Distribution Alignment and Diversity Enhancement for Data-Free Quantization. In *2022 IEEE International Conference on Data Mining (ICDM)*, 141–150.

Gao, Y.; Zhang, Z.; Zhang, H.; Zhao, M.; Yang, Y.; and Wang, M. 2021. Dictionary Pair-based Data-Free Fast Deep Neural Network Compression. In *2021 IEEE International Conference on Data Mining (ICDM)*, 121–130.

Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, 291–326. Chapman and Hall/CRC.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ko, Y.; Yu, J.-S.; Bae, H.-K.; Park, Y.; Lee, D.; and Kim, S.-W. 2021. MASCOT: A Quantization Framework for Efficient Matrix Factorization in Recommender Systems. In *2021 IEEE International Conference on Data Mining (ICDM)*, 290–299. IEEE.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kuzmin, A.; Van Baalen, M.; Ren, Y.; Nagel, M.; Peters, J.; and Blankevoort, T. 2022. Fp8 quantization: The power of the exponent. *Advances in Neural Information Processing Systems*, 35: 14651–14662.

Li, Y.; Dong, X.; and Wang, W. 2019. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. *arXiv preprint arXiv:1909.13144*.

Liu, Z.; Cheng, K.-T.; Huang, D.; Xing, E. P.; and Shen, Z. 2022. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4942–4952.

Ma, Y.; Jin, T.; Zheng, X.; Wang, Y.; Li, H.; Wu, Y.; Jiang, G.; Zhang, W.; and Ji, R. 2023. Ompq: Orthogonal mixed precision quantization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 9029–9037.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520.

Shen, M.; Yin, H.; Molchanov, P.; Mao, L.; Liu, J.; and Alvarez, J. M. 2022. Structural pruning via latency-saliency knapsack. *Advances in Neural Information Processing Systems*, 35: 12894–12908.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, Z.; Ge, C.; Wang, J.; Lin, M.; Chen, H.; Li, H.; and Sun, X. 2022. Entropy-driven mixed-precision quantization for deep network design. *Advances in Neural Information Processing Systems*, 35: 21508–21520.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Tukan, M.; Mualem, L.; and Maalouf, A. 2022. Pruning neural networks via coresets and convex geometry: Towards no assumptions. *Advances in Neural Information Processing Systems*, 35: 38003–38019.

Wang, Y.; Lu, Y.; and Blankevoort, T. 2020. Differentiable joint pruning and quantization for hardware efficiency. In *European Conference on Computer Vision*, 259–277. Springer.

Williams, M.; and Aletras, N. 2023. How does calibration data affect the post-training pruning and quantization of large language models? *arXiv preprint arXiv:2311.09755*.

Zhang, Z.; Gao, Y.; Fan, J.; Zhao, Z.; Yang, Y.; and Yan, S. 2023. SelectQ: Calibration Data Selection for Post-Training Quantization. *Authorea Preprints*.

Zhao, R.; Hu, Y.; Dotzel, J.; De Sa, C.; and Zhang, Z. 2019. Improving Neural Network Quantization without Retraining using Outlier Channel Splitting. In Chaudhuri, K.;

and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 7543–7552. PMLR.

Reproducibility Checklist

This paper:

1. Includes a conceptual outline and/or pseudocode description of AI methods introduced. - **Yes**
2. Clearly delineates statements that are opinions, hypotheses, and speculation from objective facts and results (yes/no). - **Yes**
3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no). - **Yes**
4. Does this paper make theoretical contributions? (yes/no). - **Yes**

If yes, please complete the list below:

1. All assumptions and restrictions are stated clearly and formally (yes/partial/no). - **Yes**
2. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no). - **Yes**
3. Proofs of all novel claims are included (yes/partial/no). - **partial**
4. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no). **Yes**
5. Appropriate citations to theoretical tools used are given (yes/partial/no). **Yes**
6. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA). **Yes**
7. All experimental code used to eliminate or disprove claims is included (yes/no/NA). **No**

Does this paper rely on one or more datasets? (yes/no) **Yes**

If yes, please complete the list below:

1. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA). **partial**
2. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA). **NA**
3. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA). **NA**
4. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA). **Yes**
5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA).**Yes**
6. All datasets that are not publicly available are described in detail, with an explanation of why publicly available alternatives are not scientifically satisfying (yes/partial/no/NA).**NA**

Does this paper include computational experiments? (yes/no) **Yes**

If yes, please complete the list below:

1. Any code required for pre-processing data is included in the appendix (yes/partial/no). **No**
2. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no). **No**
3. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no). **Yes**
4. All source code implementing new methods has comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no). **No**
5. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA). **NA**
6. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no). **Yes**
7. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no). **Yes**
8. This paper states the number of algorithm runs used to compute each reported result (yes/no). **No**
9. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no). **Yes**
10. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no). **Partial**
11. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA). **Yes**
12. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA). **Yes**