Post training 4-bit quantization of convolution networks for rapid-deployment

Ron Banner [1] **Yury Nahshan** [1] **Elad Hoffer** [2] **Daniel Soudry** [2]

Abstract

Neural network quantization has significant benefits for deployment on dedicated accelerators. We introduce the first practical 4-bit post training quantization approach: it does not involve training the quantized model ("fine-tuning"), nor it requires the availability of the full dataset. Yet, it maintains accuracy that is just a few percents less the state-of-the-art baseline across a wide range of convolutional models. This is unlike traditional approaches that fail entirely in these settings. To achieve this, we convert a full precision pre-trained network to a limited precision network by minimizing the quantization error at the tensor level. We analyze the trade-off between quantization noise and clipping distortion in low precision networks. This enables us to derive approximate analytical expressions for the mean-square-error degradation due to clipping. By optimizing these expressions, we show marked improvements over standard quantization schemes that normally avoid clipping.

1. Introduction

A significant drawback of deep learning models is their computational costs. Low precision is one of the key techniques being actively studied recently to conquer the problem. With hardware support, low precision training and inference can compute more operations per second, reduce memory bandwidth and power consumption, and allow larger networks to fit into a device.

For faster inference, it is often desirable to reduce the model size by quantizing weights and activations post-training, without the need to re-train/fine-tune the model. These methods commonly referred to as *post-training quantization*, are simple to use and allow for quantization with limited data. At 8-bit precision, they provide close to floating point accuracy in several popular models, e.g., ResNet, VGG and AlexNet.

Unfortunately, post-training quantization below 8 bits usually incurs significant accuracy degradation (Krishnamoorthi, 2018; Jacob et al., 2018). Several techniques to recover accuracy have been suggested by modeling the effect of quantization during training. These techniques, known as *quantization-aware training*, show significant improvement over post-training quantization (Krishnamoorthi, 2018; Choi et al., 2018; Jung et al., 2018). Yet, they involve some sort of training which greatly hamper their applicability in practice due to the following reasons.

First, these approaches are data-dependent as they need the availability of the full-size training and validation datasets, which are often unavailable from reasons such as privacy, proprietary or massiveness. They are also hardware and platform dependent since quantization related artifacts need to faithfully be modeled at training time (e.g., precise transformation to fixed-point values and fusion operations at training and inference need to agree)(Krishnamoorthi, 2018). Finally, they are rather time-consuming, requiring very long periods of optimization on deep neural network accelerators for re-training.

In this paper, we suggest a post-training quantization pipeline for converting pre-trained convolution models in full precision directly to 4-bit representation. In the absence of a training set, our pipeline only aims at minimizing the local error introduced during the quantization process (e.g., round-off errors). A key principle we adopt is knowledge about the statistical characterization of neural network distributions. These tend to have a bell-curved distribution around the mean, which enables to design efficient quantization schemes that minimize the mean value of weighted quantization errors. In this work, we apply this knowledge to minimize the mean-squared error (mse) distortion measure.

Considering activations first, we suggest to limit (henceforth, clip) the range of activation values within the tensor. With integer quantization, this clipped range is then divided into L quantization levels (as determined by the bit-width). While this introduces distortion to the original tensor, it

[1]Intel - Artificial Intelligence Products Group (AIPG) [2]Technion - Israel Institute of Technology, Haifa, Israel. Correspondence to: Cieua Vvvvv <c.vvvvv@googol.com>, Eee Pppp <ep@eden.co.uk>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

reduces the rounding error in the part of the distribution containing most of the information. Our method approximates the optimal clipping value analytically from the distribution of the tensor, by minimizing the mse measure. This analytical threshold is simple to use during run-time and can easily be integrated with other techniques for quantization. Finally, by combining this method with other precision-preserving quantization techniques (e.g., per-channel quantization and fused ReLU), we report in Table 2 small accuracy degradations on a variety of convolution models, which significantly improves prior attempts to quantize activations to 4 bits without re-training (the degradation in validation error is at most 1.6% with respect to baseline).

We next turn to consider the quantization of weights. While these have been recognized to be harder for quantization in the setting of post-training (Krishnamoorthi, 2018; Choi et al., 2018; Jung et al., 2018), they can be calculated offline, i.e., before the model is being deployed. To that end, we employ K-means clustering to group the tensor values into K centeroids corresponding to K quantum representations. By doing so we move each value from its original position to its associated quantization level in a way that minimizes the mean-square-error. To make this quantization process even more accurate, we introduce a bias correction term to ensure the mean of the quantized values is equal to the original mean.

Table 3 reports our results related to 4-bit weight quantization. While weight quantization incurs somewhat larger degradations compared to activations, we observe approximately 2-3% accuracy loss in all models excluding Inception v3. Note that a naive conversion of weights to 4-bit representation usually results with validation accuracy that approaches zero, as can be seen by the left column of Table 3. Our final results with 4-bit weights and activations (4W4A) are summarized in Table 1 as follows.

Model	Naive (4W4A)	Our pipeline (4W4A)	Reference (float32)
VGG16	23.7%	68.90%	71.59%
VGG16-BN	0.5%	69.00%	73.36%
ResNet-18	0.6%	65.28%	69.75%
ResNet-50	0.4%	72.60%	76.10%
ResNet-101	0.2%	73.10%	77.30%
Inception v3	0.0%	59.20% [1]	77.20%
AlexNet	1.8%	53.00%	56.52%

Table 1. Comparison of top-1 accuracy between our post-training 4-bit quantization pipeline and full precision baseline on ImageNet classification. Full precision models are converted directly to 4-bit weights and activations without re-training or the availability of the full datasets.

.

2. Overview of our quantization pipeline

It has been shown by (Yang et al., 2017) that almost 70% of the power consumption is done by data movement to and from memory. Therefore. Our quantization pipeline focuses mainly on limiting the memory bandwidth requirements. As such, inner calculations are still done at higher precision, enabling the use of the following methods to reduce quantization effect at the tensor level: (i) placing an upper bound on the output to control dynamic range (i.e., clipping function); (ii) fusing the ReLU into the convolution layer; (iii) using a different scale and offset for each convolution kernel, also known as per-channel quantization; (iv) non-uniformally positioning the quantized weights to match the fact that some weights tend to occur more frequently compared to others.

2.1. Optimized clipping for activations

This analytical clipping method is the main novel part of our work. We study the effect of clipping with the aim of improving overall quantization noise. To this end, we first study the distribution of values within activation tensors. By running a few statistical tests (see Appendix), we were able to see on a variety of convolution models that activation tensors follow either a Gaussian or Laplacian distributions. This modeling of activation tensors enables a clear formulation of the quantization process and constitutes the first step for its optimization. In subsection 4.1, we provide a rigorous formulation to optimize the quantization effect of activations by analyzing both the Gaussian and the Laplace distributions.

2.2. Per-Channel-Quantization

It is often the case where activation distributions vary significantly between different channels. In these cases, calculating a scale-factor per channel can provide good accuracy for post-training quantization (Krishnamoorthi, 2018). The per-channel scale has shown to be important both for inference (Rastegari et al., 2016) and for training (Wu et al., 2018).

2.3. Fused ReLU

In convolution neural networks, most convolutions are followed by a rectified linear unit (ReLU), zeroing the negative values. There are many scenarios where these two operations can be fused to avoid accumulation of quantization noise. In these settings, we can ignore the negative values and find an optimal clipping value α for the positive half space $[0, \alpha]$. Fused ReLU provides a smaller dynamic range, which leads to a smaller spacing between the different quantization levels and therefore smaller roundoff error

[1] results turn to be 73.03% when we allow 19% of the weight parameters to be at 8-bit precision.

upon quantization. In subsection 4.2 we provide a detailed analysis for optimal value of α.

2.4. Non-uniform quantization of weights

Neural network distributions are not uniform but rather have bell-shaped distributions. For these cases, non-uniform quantization enables to assign more quantization levels to regions with high concentration of values while under-utilized regions with fewer values will have fewer levels. These methods are more complicated and cannot be used for activations during inference. Yet, unlike activations, weights can be quantized offline once training is complete. Therefore, assuming the availability of a look-up table, weights can be quantized offline in a non-uniform manner optimizing the quantization of each particular tensor.

3. Previous work

In many cases, taking a model trained for full precision and directly quantizing it to 8-bit precision, without any re-training, can result in a relatively low loss of accuracy (Jacob et al., 2018; Gong et al., 2018; Krishnamoorthi, 2018). Yet, naively converting a full precision model below 8-bit representation usually incurs significant accuracy degradation (Krishnamoorthi, 2018; Jacob et al., 2018). Many attempts have been made to diminish this effect, but they usually employ training of the model with quantization constraints or modifying network structure (Lin et al., 2017; McKinstry et al., 2018; Rastegari et al., 2016; Zhou et al., 2016; Choi et al., 2018). To the best of our knowledge, there have been only a few attempts to clip activations before. Choi et al. (2018); Jung et al. (2018) have proposed an activation clipping parameter that is optimized during training. These previous works introduce an activation function with a parameterized clipping level that is dynamically adjusted via gradient descent-based training.

Perhaps the most relevant previous work that relates to our clipping study is due to (Migacz, 2017) who also proposes to clip activations post-training. Migacz (2017) suggests an iterative time-consuming method to search for a good clipping threshold based on the Kullback-Leibler Divergence (KLD) measure. Our analytical clipping approach outperforms KLD in almost all models, as well as being orders of magnitude faster. For example, when using KLD for ResNet50 we need to iterate for the best clipping value 4,000 iterations per activation tensor, resulting with 48 hours on Xeon HW. Moreover, since KLD is an exhaustive search procedure, it cannot be integrated efficiently with per-channel quantization to minimize quantization noise. For example, there are approximately thousand of times more channel activations than layer activations in ResNet50, rendering unfeasible the combination of quantization per channel and a search for optimized clipping threshold using the KLD measure. This stands in contrast with our simple approach that can easily be integrated with other techniques for quantization.

Considering now the quntization of weights, the concept of non-uniform quantization has been investigated by (Park et al., 2018; Baskin et al., 2018; Han et al., 2015). These works either train the network to perform well with quantized values or use an iterative quantization method where quantization is conducted repeatedly after re-training the model. Among these works, the work that is most relevant to us has been suggested by (Han et al., 2015), where the authors replace the weight values with indexes pointing to a finite codebook of shared values. Like our approach, they also use K-means clustering to identify the optimal quantum values. Yet, they reduce the quantization effect by re-train the code book, a process which is not possible in our settings. On the other hand, we improve the quantization process without the use of the training and validation data-sets by correcting the mean per channel. This bias correction method ensures that the mean of the quantized weights in each channel would equal to the original mean before quantization was made.

4. Low-Bit Width Activations

In this section, we suggest a three stage process for quantization: analytic clipping, fuse the ReLU into the convolution layer and use a different scale and offset for each convolution kernel (i.e., per-channel quantization). We first provide a detailed analysis of the clipping method, which is novel to this work. We then turn to provide a short explanation about the per-channel quantization and Fused ReLU methods and describe how these are adjusted to work in synergy with the clipping method. To the best of our knowledge, we are the first to quantize activations to 4-bit precision without re-training while maintaining accuracy close to floating-point accross a wide range of networks.

4.1. Analytical study for optimal quantization

In the following we derive a generic expression for the expected mean-square-error as a function of clipping value for either Gaussian or Laplace distributions. Let X be a high precision random variable with a probability density function $f(x)$. Without loss of generality, we assume a prepossessing step has been made so that the average value in the tensor zero i.e., $\overline{X} = \mu = 0$ (we do not lose generality since we can always subtract and add this mean). Assuming bit-width M, we would like to quantize the values in the tensors uniformally to 2^M discrete values.

Commonly (e.g., in GEMMLOWP (Jacob et al., 2017)), integer tensors are uniformly quantized in the range $[-\alpha, \alpha]$, where α is determined by the tensor maximal absolute value. In the following we show that the this choice of α is sub-

optimal, and suggest a model where the tensor values are clipped to reduce quantization noise. For any $x \in \mathbb{R}$, we define the clipping function $\text{clip}(x, \alpha)$ as follows

$$\text{clip}(x, \alpha) = \begin{cases} x & \text{if } |x| \leq \alpha \\ \text{sign}(x) \cdot \alpha & \text{if } |x| > \alpha \end{cases} \tag{1}$$

Denoting by α the clipping value, the range $[\alpha, -\alpha]$ is partitioned to 2^M equal quantization regions. Hence, the quantization step Δ between two adjacent quantized values is established as follows:

$$\Delta = \frac{2\alpha}{2^M} \tag{2}$$

Our model assumes values are rounded to the midpoint of the region (bin) i.e., for every index $i \in [0, 2^M - 1]$ all values that fall in $[-\alpha + i \cdot \Delta, -\alpha + (i+1) \cdot \Delta]$ are rounded to the midpoint $q_i = -\alpha + (2i+1)\frac{\Delta}{2}$, as illustrated in Figure 1. Then, the expected mean-square-error between X and its quantized version $Q(X)$ can be written as follows:

$$\begin{aligned} &E[(X - Q(X))^2] = \\ &= \int_{-\infty}^{-\alpha} f(x) \cdot (x + \alpha)^2 dx + \\ &+ \sum_{i=0}^{2^M - 1} \int_{-\alpha + i \cdot \Delta}^{-\alpha + (i+1) \cdot \Delta} f(x) \cdot (x - q_i)^2 dx + \\ &+ \int_{\alpha}^{\infty} f(x) \cdot (x - \alpha)^2 dx \end{aligned} \tag{3}$$

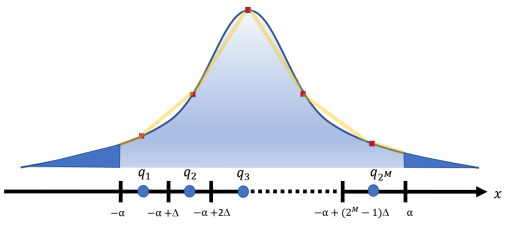

Figure 1. uniform quantization

Equation 3 is composed of three parts. The first and last terms quantify the contribution of $\text{clip}(x, \alpha)$ to the expected mean-square-error. Note that for symmetrical distributions around zero (e.g., Gaussian $N(0, \sigma^2)$ or $\text{Laplace}(0, b)$) these two terms are equal and their sum can therefore be evaluated by multiplying any of the terms by 2. The second term corresponds to the expected mean-square-error when the range $[-\alpha, \alpha]$ is quantized uniformly to 2^M discrete levels. This terms corresponds to the quantization noise introduced when high precision values in the range $[-\alpha, \alpha]$ are rounded to the nearest discrete value.

4.1.1. Quantization noise

We approximate the density function f by a construction of a piece-wise linear function whose segment breakpoints are points in f, as illustrated in Figure 1. Since we consider only smooth probability density functions (e.g., Gaussian or Laplace), the resulting approximation error is small for sufficient resolution i.e., small quantization step size Δ. In the appendix we show that given a density function f, the quantization noise can be approximated as follows:

$$\begin{aligned} &\sum_{i=0}^{2^M - 1} \int_{-\alpha + i \cdot \Delta}^{-\alpha + (i+1) \cdot \Delta} f(x) \cdot (x - q_i)^2 dx \approx \\ &\approx \frac{2 \cdot \alpha^3}{3 \cdot 2^{3M}} \cdot \sum_{i=0}^{2^M - 1} f(q_i) \end{aligned} \tag{4}$$

Equation 4 represents the rounding error (as opposed to clipping error) due to the rounding of all values in the bin i to its center q_i. For sufficient resolution and a smooth density function, the density function f can be approximated by a uniform distribution in the range $[-\alpha, \alpha]$ (Marco & Neuhoff, 2005), which enables much simpler analysis with little effect on the accuracy. In Figure 2, we show that with this assumption the analytic results are in a good agreement with the simulation results. By substituting the uniform density function $f(x) = \frac{1}{2\alpha}$ into Equation 4, the following simpler rounding error can be computed:

$$\begin{aligned} &\sum_{i=0}^{2^M - 1} \int_{-\alpha + i \cdot \Delta}^{-\alpha + (i+1) \cdot \Delta} f(x) \cdot (x - q_i)^2 dx \approx \\ &\approx \frac{2 \cdot \alpha^3}{3 \cdot 2^{3M}} \cdot \sum_{i=0}^{2^M - 1} \frac{1}{2\alpha} = \frac{\alpha^2}{3 \cdot 2^{2M}} \end{aligned} \tag{5}$$

By substituting Equation 5 into Equation 3, and using the symmetrical argument mentioned above, Equation 3 can be simplified for symmetrical distributions as follows:

$$\begin{aligned} &E[(X - Q(X))^2] = \\ &= \frac{\alpha^2}{3 \cdot 2^{2M}} + 2 \cdot \int_{\alpha}^{\infty} f(x) \cdot (x - \alpha)^2 dx \end{aligned} \tag{6}$$

In the following we provide a closed form solution for the case where the density probability distribution function $f(x)$ is either Gaussian $N(0, \sigma^2)$ or $\text{Laplace}(0, b)$.

4.1.2. Clipping noise

In the following we develop an expression based on Equation 6 for the Laplace case. In the appendix we provide a similar analysis for the case where the probability density function is Gaussian $N(0, \sigma^2)$

Assuming $\mu = 0$, we have the following Laplace density function $f(x) = \frac{1}{2b}\mathrm{e}^{-\frac{|x|}{b}}$. In order to derive a closed form solution for Equation 6, we need to evaluate

$$2 \cdot \int_{\alpha}^{\infty} f(x) \cdot (x-\alpha)^2 dx. \tag{7}$$

Let $\Psi(x)$ represent the expression below:

$$\Psi(x) = \frac{\mathrm{e}^{-\frac{x}{b}}}{2}\left[2\alpha b - 2b^2 - \alpha^2 - x^2 - 2\left(b-\alpha\right)x\right] \tag{8}$$

By taking the derivative of $\Psi(x)$ with respect to x, it is easy to see that $\Psi(x)$ is the correct antiderivative of the integrand in equation 7. Hence,

$$\int_{\alpha}^{\infty} f(x) \cdot (x-\alpha)^2 dx = \Psi(\mathrm{inf}) - \Psi(\alpha) = b^2 \cdot e^{-\frac{\alpha}{b}}$$

We can finally state Equation 6 for the laplace case as follows.

$$\begin{aligned} &E[(X - Q(X))^2] \approx \\ &\approx 2 \cdot b^2 \cdot e^{\frac{-\alpha}{b}} + \frac{2 \cdot \alpha^3}{3} \cdot \sum_{i=0}^{2^M-1} f(q_i) = \\ &= 2 \cdot b^2 \cdot e^{\frac{-\alpha}{b}} + \frac{\alpha^2}{3 \cdot 2^{2M}} \end{aligned} \tag{9}$$

In figure 2 we introduce the mean-square-error as a function of clipping value for various bit widths.

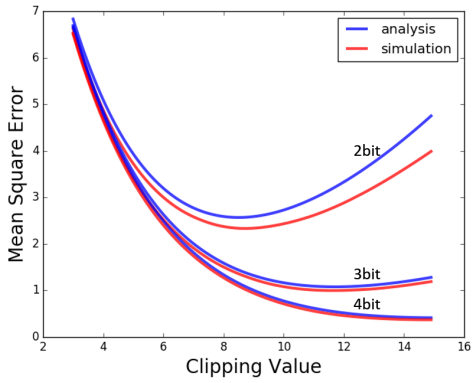

Figure 2. Expected mean-square-error as a function of clipping value for different quantization levels (Laplace ($\mu = 0$ and $b = 1$)). Analytical results, stated by Equation 9, are in a good agreement with simulations, which where obtained by clipping and quantizing 10,000 values, generated from a Laplace distribution. As expected, the difference occurs only for very low-bit width and large clipping values where the uniform assumption tends to break.

Finally, in order to find the α that give the minimum MSE, the corresponding derivative with respect to α is set equal to zero as follows:

$$\frac{\partial E[(X - Q(X))^2]}{\partial \alpha} = \frac{2\alpha}{3 \cdot 2^{2M}} - 2b\mathrm{e}^{-\frac{\alpha}{b}} = 0 \tag{10}$$

4.2. Fused ReLU

We turn to adjust Equations 9 for the case where convolutions and rectified linear units (ReLU) are fused to avoid accumulation of noise. The corresponding analysis for the Gaussian is available in the Apendix.

Given a high precision random variable X with a probability density function $f(x)$ and a ReLU activation $g(x) = \max(0, x)$, we would like to minimize the following expected mean square-error

$$E\Big[\Big(g(X) - Q(g(X))\Big)^2\Big] \tag{11}$$

Assuming the probability density function $f(x)$ has a symmetrical distribution around zero, there are two adjustments that need to be made in the analysis of Section 4:

- The quantization step Δ is now set according to the range $[0, \alpha]$. Hence, Equation 2 should be modified as follows:
$$\Delta = \frac{\alpha}{2^M}$$
- Since we consider only the positive values, Equation 3 should ignore the negative contribution i.e.,
$$\begin{aligned} &E[(X - Q(X))^2] = \\ &= \sum_{i=0}^{2^M-1} \int_{i\cdot\Delta}^{(i+1)\cdot\Delta} f(x) \cdot (x - q_i)^2 dx + \\ &+ \int_{\alpha}^{\infty} f(x) \cdot (x-\alpha)^2 dx \end{aligned}$$

This translates to the following adjustments in Equation 9 for the Laplace case:

$$\begin{aligned} &E\Big[\Big(g(X) - Q(g(X))\Big)^2\Big] \approx \\ &\approx b^2 \cdot e^{\frac{-\alpha}{b}} + \frac{\alpha^2}{24 \cdot 2^{2M}} \end{aligned} \tag{12}$$

4.3. Per-channel quantization

Our analytical clipping approach needs to estimate the location and scale parameters of either a Gaussian or a Laplace distribution. Yet, estimation at the channel level often turns to be too noisy. For example, some of the channels in ResNet18 are with dimensions of just 7x7, making high sampling error for these population parameters. Therefore, to estimate channel-level distribution more accurately, we use a profiling data-set of 256 images sampled from the training set. By running a network in forward pass and collecting statistics, we can estimate correctly the population parameters at the channel level. These values are then used for our analytical derivations to estimate the corresponding clipping values.

5. Low-Bit Width Weights

Unlike activations, the quantization of weights can be done offline. This allows to model this process as an optimization problem which seeks to minimize the mean-square-error. Given a tensor of weights W, we would like to partition the weights into n discrete quantization levels $Q = \{q_1, q_2, ..., q_n\}$ so as to minimize the overall sum of square distance i.e.,

$$\arg\min_{Q} \sum_{i=1}^{n} \sum_{w \in q_i} ||w - \mu_i||^2 \tag{13}$$

where μ_i is the mean of the points in q_i.

K-means clustring can be used for solving this l_2 optimization problem (Kanungo et al., 2002). We initialize the K-means clustering with centroids that are spread linearly between the maximum and minimum tensor values. It is important to note that standard random assignment of centroids does not cover the outliers and might result with an underestimate of the dynamic range.

Finally, due to the non-uniform nature of the K-means quantization there exists a drift from the mean. Denoting by $W_c \subseteq W$ the weight filter of channel c and its quantized version by W_c^q, the mean of the quantized weights is no longer equal to the mean of the high precision weights i.e., in $\overline{W_c^q} \neq \overline{W_c}$. The difference between the two is a bias that needs to be corrected as follows:

$$Bias_c = \overline{W_c} - \overline{W_c^q} \tag{14}$$

Then, for each channel c we perform bias-correction for all weights in W_c^q as follows:

$$w \longleftarrow w + Bias_c, \quad \forall w \in W_c^q \tag{15}$$

Implementation wise, an additional offset for bias correction is needed for each channel. In figure 3 we visually demonstrate the advantage of K-means clustering over standard GEMMLOWP (Jacob et al., 2017) for the quantization of weights.

6. Experiments

To evaluate how good we can get without re-training, we performed an extensive performance anaslysis of our schemes and compared against the current state-of-the-art baselines. We quantize seven convolution models originally pre-trained on the ImageNet dataset and, consider three setups for our experiments. The first setup keeps all weights at 8-bit precision and quantize only the activations to 4 bits (8W4A). In the second setup we quantize the weights to 4 bits and keep the activations at 8 bit precision (4W8A). Finally, in the last setup we quantize both weights and activations to 4-bit precision (4W4A). In all setups we use the common practice to quantize the first and the last layer as well as average/max pooling layers to 8-bit precision. All results are obtained using simulated quantization of weights and activations. The code to replicate all our experiments is available online [2].

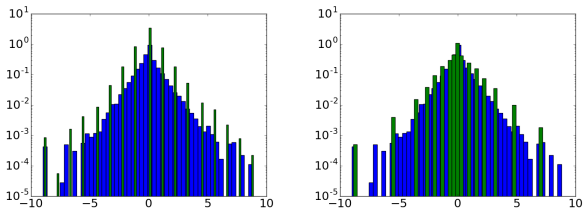

Figure 3. Histograms of uniform and non-uniform weight quantization (semi-log plot). The blue histogram corresponds to the high precision values and the green histogram corresponds to the quantized values. On the left we see standard 4-bit GEMMLOWP (uniform) quantizer and on the right we see non-uniform quantization calculated using K-means clustering forming 16 quantization levels. With K-means clustering, quantization levels are spaced more closely for values that are more frequent, resulting with lower average rounding error (i.e., quantization noise).

Table 2 summaries the classification test accuracies of different popular pre-trained convolution networks after activations are quantized to 4-bit precision in a post-training manner (8W4A). In Table 3 we summarize the results related to the setting of 4-bit weight and 8-bit activations (4W8A). We compare between three approaches: per-channel quantization, K-means, and K-means followed by a bias-correction to equalize the mean of the quantized weights with the mean of the original weights. Note that from a hardware perspective, the K-means quantization based approach mandates the availability of a lookup table to transform each quantization level (cluster) to its quantum value (centroid). Therefore, it cannot be combined efficiently with per-channel quantization in the same tensor. Specifically, for 4-bit weights we need 16 quantization levels per channel, and if a layer has, say, 1024 channels, we may run out of internal memory. Finally, in Table 1, we provide the final results of 4-bit weights and activations (4W4A).

[2] https://github.com/submission2019/cnn-quantization

Model	Naive (8W4A)	KLD (8W4A)	Analytical Clipping (8W4A)	Full Pipeline (8W4A)	Reference (float32)
VGG16	53.90%	67.04%	67.40%	70.49%	71.59%
VGG16-BN	29.50%	65.85%	67.60%	72.44%	73.36%
ResNet-18	53.20%	65.06%	65.80%	68.89%	69.75%
ResNet-50	52.70%	70.80%	71.45%	74.84%	76.10%
ResNet-101	50.80%	71.70%	69.53%	76.02%	77.30%
Inception v3	41.40%	59.25%	60.80%	75.60%	77.20%
AlexNet	41.60%	49.55%	52.20%	55.22%	56.52%

Table 2. Validation accuracy of various architectures quantized post-training to 8-bit weights and 4-bit activations (8W4A): *Reference (8W4A)* refers to the conventional quantization method based on the maximum and minimum representable value which shows severe accuracy loss. *KLD* refers to the iterative method suggested by NVIDIA to search for a good clipping threshold based on the Kullback-Leibler Divergence measure Migacz (2017). *Analytical clipping* refers to our analytic clipping approach described in Section 4.1; unlike KLD, which is a brute force technique, our approach is order of times faster, and, excluding ResNet 101, maintains higher validation accuracy. *Full pipeline* refers to the combination of our anlytical approach with the precision-preserving quantization techniques discussed in Sections 4.2 and 4.3. Due to scaling issues of KLD, we could not test its performance in conjecture with the other search-based quantizaton schemes (e.g., KLD with per-channel quantization). *Reference (float32)* uses full precision models with 32 bit weights and activations. For comparison, Krishnamoorthi (2018) reports for this post-training quantization setting a validation accuracy of 36% for ResNet50 and 59% for Inception v3.

Model	Naive (4W8A)	Channel wise (4W8A)	K-Means (4W8A)	K-means+Bias Corr (4W8A)	Reference (float32)
VGG16	29%	70.2%	69.2%	70%	71.59%
VGG16-BN	0.5%	68.4%	68.8%	70.1%	73.36%
ResNet-18	0.8%	59.3%	65%	67%	69.75%
ResNet-50	0.4%	72.4%	72.8%	74.2%	76.13%
ResNet-101	0.2%	74.6%	74%	75%	77.3%
Inception v3	0.1%	37.6%	53.4%	62.4% [3]	77.2%
AlexNet	1.8%	52.9%	54.2%	54.7%	56.52%

Table 3. Validation accuracy for 4-bit weights and 8-bit activations (4W8A): *Reference (4W8A)* refers to the conventional quantization method, where weights are quantized uniformly between the maximum and minimum representable values.*Channel-wise* refers to the use of an exclusive scale factor per-channel, known as per-channel quantization. *K-means* clustering is used to quantize the weights so that all values that fall into the same cluster are quantized to the same weight. *K-means + bias correction* is used to correct the K-means quantization by an additive correction so that mean of quantized values would coincide with the original values. *Reference (float32)* uses full precision models with 32 bit weights and activations.

7. Discussion

Post-training quantization techniques are simpler to use and allow for quantization with limited data-sets. They have been widely adopted by many practitioners for 8-bit quantization due to their easy deployment that does not incur significant loss in accuracy. Yet, prior attempts to get below 8-bit precision usually incur severe accuracy degradation and require re-training to obtain reasonable accuracy (Krishnamoorthi, 2018). Our findings suggest that by reducing quantization errors locally at the tensor level, post-training quantization is possible also at 4-bit precision.

To make that happen, we employ a variety of techniques to limit the effect of quantization. We make a fundamental use of the statistical dispersion of weights and activations. These tend to have a bell curved distribution around the mean i.e., most values lie within a small range and very few tend to be much larger/ smaller than the mean (Baskin et al., 2018; Han et al., 2015). Assuming activation values follow a Laplace/Gaussian distribution, we develop an optimized framework to clip these statistical outliers dynamically during run-time (i.e., extreme values will incur a larger quantization error, but in total this will reduce the distortion due to quantization). On the other hand, since weights can be quantized "offline", we adopt the K-means quantizer for weights, which is the optimal choice in the ℓ_2 sense.

Our simulations highlight the major advantage of quantizers

[3] results turn to be 74.90% when we allow 19% of the weights to be 8-bit precision

adapted to neural network distributions over conventional methods. Moreover, by applying other techniques such per-channel quantization, fused-ReLU and bias-correction further improvements in a variety of models are obtained, offering for the first time a degradation of a few percent, while previous approaches such as GEMMLOWP (Jacob et al., 2017)) and KLD (Migacz, 2017) completely fail.

References

Baskin, Chaim, Schwartz, Eli, Zheltonozhskii, Evgenii, Liss, Natan, Giryes, Raja, Bronstein, Alex M, and Mendelson, Avi. Uniq: Uniform noise injection for the quantization of neural networks. *arXiv preprint arXiv:1804.10969*, 2018.

Choi, Jungwook, Wang, Zhuo, Venkataramani, Swagath, Chuang, Pierce I-Jen, Srinivasan, Vijayalakshmi, and Gopalakrishnan, Kailash. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

Gong, Jiong, Shen, Haihao, Zhang, Guoming, Liu, Xiaoli, Li, Shane, Jin, Ge, Maheshwari, Niharika, Fomenko, Evarist, and Segal, Eden. Highly efficient 8-bit low precision inference of convolutional neural networks with intelcaffe. *arXiv preprint arXiv:1805.08691*, 2018.

Han, Song, Mao, Huizi, and Dally, William J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Jacob, Benoit, Kligys, Skirmantas, Chen, Bo, Zhu, Menglong, Tang, Matthew, Howard, Andrew, Adam, Hartwig, and Kalenichenko, Dmitry. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2704–2713, 2018.

Jacob, Benoit et al. gemmlowp: a small self-contained low-precision gemm library.(2017), 2017.

Jung, Sangil, Son, Changyong, Lee, Seohyung, Son, Jinwoo, Kwak, Youngjun, Han, Jae-Joon, and Choi, Changkyu. Joint training of low-precision neural network with quantization interval parameters. *arXiv preprint arXiv:1808.05779*, 2018.

Kanungo, Tapas, Mount, David M, Netanyahu, Nathan S, Piatko, Christine D, Silverman, Ruth, and Wu, Angela Y. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7):881–892, 2002.

Krishnamoorthi, Raghuraman. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

Lin, Xiaofan, Zhao, Cong, and Pan, Wei. Towards accurate binary convolutional neural network. In *Advances in Neural Information Processing Systems*, pp. 345–353, 2017.

Lopes, Raul HC. Kolmogorov-smirnov test. In *International encyclopedia of statistical science*, pp. 718–720. Springer, 2011.

Marco, Daniel and Neuhoff, David L. The validity of the additive noise model for uniform scalar quantizers. *IEEE Transactions on Information Theory*, 51(5):1739–1755, 2005.

McKinstry, Jeffrey L, Esser, Steven K, Appuswamy, Rathinakumar, Bablani, Deepika, Arthur, John V, Yildiz, Izzet B, and Modha, Dharmendra S. Discovering low-precision networks close to full-precision networks for efficient embedded inference. *arXiv preprint arXiv:1809.04191*, 2018.

Migacz, S. 8-bit inference with tensorrt. In *GPU Technology Conference*, 2017.

Park, Eunhyeok, Yoo, Sungjoo, and Vajda, Peter. Value-aware quantization for training and inference of neural networks. *arXiv preprint arXiv:1804.07802*, 2018.

Rastegari, Mohammad, Ordonez, Vicente, Redmon, Joseph, and Farhadi, Ali. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pp. 525–542. Springer, 2016.

Soudry, Daniel, Hubara, Itay, and Meir, Ron. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. pp. 963–971, 2014.

Wu, Shuang, Li, Guoqi, Chen, Feng, and Shi, Luping. Training and inference with integers in deep neural networks. *arXiv preprint arXiv:1802.04680*, 2018.

Yang, Tien-Ju, Chen, Yu-Hsin, Emer, Joel, and Sze, Vivienne. A method to estimate the energy consumption of deep neural networks. *Energy*, 1(L2):L3, 2017.

Zhou, Shuchang, Wu, Yuxin, Ni, Zekun, Zhou, Xinyu, Wen, He, and Zou, Yuheng. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

A. Probability distribution fitting for activations

It has already been noted by prior arts that neural network distributions are near Gaussian in practice, sometimes further controlled by procedures such as batch normalization Soudry et al. (2014). In this section, we construct an estimate of the underlying probability density function of the activation tensors. Traditionally, no clipping was made in standard GEMMLWOP. Hence, quantization levels are uniformity spaced between the largest and the smallest values in the tensor. Yet, this approach is non-optimal, due to the fact that the activation tensors have bell-shaped distributions. Therefore, to reduce quantization noise (or increase resolution), it might be desirable to clip the values above a certain threshold. Specifically, we need to find the best clipping value that, on the one hand, maintains a low clipping rate, but, on the other hand, improves resolution. To do so we must first understand the underlying data distribution.

To that end, we collect the data of various tensors at different layers. We observe that the data is in general symmetrically distributed around a mean. We next estimate the goodness of fit to several bell-shaped distributions. This is done by measuring the static distance (largest vertical line) between the cumulative distribution function (CDF) of the empirically observed distribution and the CDF of the reference distribution (also know by Kolmogorov-Smirnov test Lopes (2011). By considering the activations of all layers of ResNet50 on ImageNet, we obtain the average static distance to each of the following distributions, with p-value< 0.01:

Distribution	Static Distance
Laplace	0.070
Normal	0.053
Logistic	0.150
Cauchy	0.142
Uniform	0.490
Loglaplace	0.505
Lognorm	0.540

As one can see, the best fit is established by the Laplace and Normal distributions. In figures 4 and 5, we plot the normalized distributions of the activation tensors at different layers for Resnet50, and compare them against both priors. The statistical fit to both models is acceptable and similar.

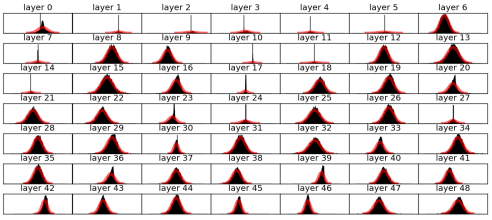

Figure 4. Fitting activation tensors to Gaussian distribution at different ResNet50 layers.

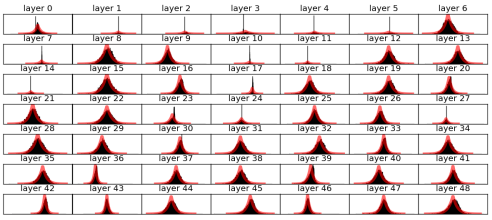

Figure 5. Fitting activation tensors to Laplace distribution at different ResNet50 layers.

B. Piece-wise linear approximation

Here we provide a more accurate analysis related to the qunatization noise (i.e., the second term in Equation 3), measured as the expected mean-square-error when the range $[-\alpha, \alpha]$ is quantized uniformly to 2^M discrete levels. To that end, we approximate the density function f by a construction of a piece-wise linear function g such that $f(q_i) = g(q_i)$ for each $i \in [0, 2^M - 1]$. Since we consider only smooth probability density functions (e.g., Gaussian or Laplace), the resulting approximation error is small for sufficient resolution i.e., small quantization step size Δ. In figure 1 we provide an illustration for this construction.

We turn to calculate the linear equation for each line segment of the piece-wise linear function g, falling in the range $[-\alpha + i \cdot \Delta, -\alpha + (i+1) \cdot \Delta]$. To that end, we consider the slope (derivative) and the value of the density function at the midpoint q_i. With these two values we can define for each segment $i \in [0, 2^M - 1]$ the corresponding form of linear approximation:

$$g(x) = f(q_i) + \frac{df}{dx}(q_i) \cdot (x - q_i), \quad \text{where } x \in [-\alpha + i \cdot \Delta, -\alpha + (i+1) \cdot \Delta] \tag{16}$$

We now turn to calculate the second term in Equation 3. By equation 16, and since q_i is defined to be the midpoint

between the integration limits, the following holds true

$$\begin{aligned}
&\sum_{i=0}^{2^M-1}\int_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} f(x)\cdot(x-q_i)^2dx \approx \\
&\approx \sum_{i=0}^{2^M-1}\int_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} g(x)\cdot(x-q_i)^2dx = \\
&= \sum_{i=0}^{2^M-1}\int_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} f(q_i)\cdot(x-q_i)^2 + \\
&+ \sum_{i=0}^{2^M-1}\int_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} \frac{df}{dx}(q_i)\cdot(x-q_i)^3dx = \\
&= \frac{1}{3}\sum_{i=0}^{2^M-1} f(q_i)\cdot(x-q_i)^3\Big|_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} + \\
&+ \frac{1}{4}\sum_{i=0}^{2^M-1}\frac{df}{dx}(q_i)\cdot(x-q_i)^4\Big|_{-\alpha+i\cdot\Delta}^{-\alpha+(i+1)\cdot\Delta} = \\
&= \frac{\Delta^3}{12}\sum_{i=0}^{2^M-1} f(q_i) = \frac{2\cdot\alpha^3}{3\cdot 2^{3M}}\cdot\sum_{i=0}^{2^M-1} f(q_i)
\end{aligned}$$

C. Clipping noise (Gaussian case)

We now turn to evaluate Equation 6 for the Gaussian case. Given a Gaussian random variable $X \sim N(0,\sigma^2)$, we define $\Psi(x)$ to represent the expression below:

$$\Psi(x) = \frac{(\alpha^2+\sigma^2)\,\mathrm{erf}\left(\frac{x}{\sqrt{2}\sigma}\right)}{2} + \\ - \frac{(x\sigma - 2\alpha\sigma)\,\mathrm{e}^{-\frac{x^2}{2\sigma^2}}}{\sqrt{2\pi}} \tag{17}$$

As in subsection 4.1.2, one can observe that by taking the derivative of $\Psi(x)$ with respect to x, it is easy to show that $\Psi(x)$ is the correct antiderivative of Equation 7 for the case where f represents the Gaussian density function i.e., $f(x) = \frac{1}{\sqrt{2\pi}\sigma}\mathrm{e}^{-\frac{x^2}{2\sigma^2}}$. Next, we use $\Psi(x)$ on the range $[\alpha,\infty]$ to evaluate Equation 7 for the Gaussian case as follows:

$$\begin{aligned}
&\int_\alpha^\infty f(x)\cdot(x-\alpha)^2dx = \Psi(\infty) - \Psi(\alpha) \\
&= \frac{\alpha^2+\sigma^2}{2}\cdot\left[1-\mathrm{erf}\left(\frac{\alpha}{\sqrt{2}\sigma}\right)\right] - \frac{\alpha\cdot\sigma\cdot\mathrm{e}^{-\frac{\alpha^2}{2\cdot\sigma^2}}}{\sqrt{2\pi}}
\end{aligned}$$

Equation 6 can thus be written for the case of Gaussian distribution as follows:

$$\begin{aligned}
&E[(X-Q(X))^2] \approx \\
&\approx (\alpha^2+\sigma^2)\cdot\left[1-\mathrm{erf}\left(\frac{\alpha}{\sqrt{2}\sigma}\right)\right] + \\
&+ \frac{\alpha^2}{3\cdot 2^{2M}} - \frac{\sqrt{2}\alpha\cdot\sigma\cdot\mathrm{e}^{-\frac{\alpha^2}{2\cdot\sigma^2}}}{\sqrt{\pi}}
\end{aligned} \tag{18}$$

In figure 6 we introduce the mean-square-error as a function of clipping value for various bit widths.

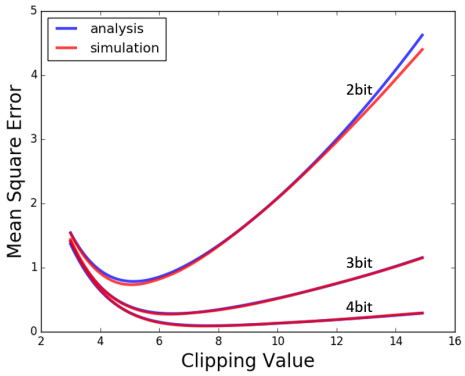

Figure 6. Expected mean-square-error as a function of clipping value for different quantization levels (Gaussian ($\mu = 0$ and $\sigma = 1$)). Analytical results , stated by Equation 24, are in a good agreement with simulations, which where obtained by clipping and quantizing 10,000 values, generated from a Laplace distribution. As expected, the difference occurs only for very low-bit width and large clipping values where the uniform assumption tends to break.

In order to find the optimal clipping values for which mean-square-error is minimized, we need to differentiate $E[(X-Q(X))^2]$ with respect to α and set the derivative equal to zero as follows.

$$\begin{aligned}
&\frac{\partial E[(X-Q(X))^2]}{\partial\alpha} = \\
&= \alpha\left[1-\mathrm{erf}\left(\frac{\alpha}{\sqrt{2}\sigma}\right)\right] - \frac{\sigma^2\mathrm{e}^{-\frac{\alpha^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} - \frac{\sigma\mathrm{e}^{-\frac{\alpha^2}{2\sigma^2}}}{\sqrt{2\pi}} + \\
&+ \frac{2\alpha}{3\cdot 2^{2M}} = 0
\end{aligned} \tag{19}$$

D. Optimal Quantizer for fused ReLU Activations

In this section we adjust Equations 9 and 18 for the case where convolutions and rectified linear units (ReLU) are fused to avoid accumulation of noise.

The ReLU is defined by zeroing the negative half space i.e., $g(x) = \max(0,x)$. Given a high precision random variable

X with a probability density function $f(x)$ we would like to minimize the following expected mean square-error

$$E\Big[\Big(g(X) - Q(g(X))\Big)^2\Big] \tag{20}$$

Assuming the probability density function $f(x)$ has a symmetrical distribution around zero, there are two adjustments that need to be made in the analysis of Section 4:

(1) The quantization step Δ is now set according to the range $[0, \alpha]$. Hence, Equation 2 should be modified as follows:

$$\Delta = \frac{\alpha}{2^M} \tag{21}$$

(2) Since we consider only the positive values, Equation 3 should ignore the negative contribution i.e.,

$$\begin{aligned} &E[(X - Q(X))^2] = \\ &= \sum_{i=0}^{2^M-1} \int_{i\cdot\Delta}^{(i+1)\cdot\Delta} f(x)\cdot(x - q_i)^2 dx + \\ &+ \int_{\alpha}^{\infty} f(x)\cdot(x-\alpha)^2 dx \end{aligned} \tag{22}$$

This translates to the following adjustments in Equation 9 for the Laplace case:

$$\begin{aligned} &E\Big[\Big(g(X) - Q(g(X))\Big)^2\Big] \approx \\ &\approx b^2 \cdot e^{\frac{-\alpha}{b}} + \frac{\alpha^2}{24\cdot 2^{2M}} \end{aligned} \tag{23}$$

Similarly, for the Gaussian case Equation 18 is modified as follows:

$$\begin{aligned} &E\Big[\Big(g(X) - Q(g(X))\Big)^2\Big] \approx \\ &\approx \frac{\alpha^2+\sigma^2}{2}\cdot\left[1 - \operatorname{erf}\left(\frac{\alpha}{\sqrt{2}\sigma}\right)\right] + \\ &+ \frac{\alpha^2}{24\cdot 2^{2M}} - \frac{\alpha\cdot\sigma\cdot \mathrm{e}^{-\frac{\alpha^2}{2\cdot\sigma^2}}}{\sqrt{2\pi}} \end{aligned} \tag{24}$$