A2Q+: Improving Accumulator-Aware Weight Quantization

Ian Colbert [1] **Alessandro Pappalardo** [2] **Jakoba Petri-Koenig** [2] **Yaman Umuroglu** [2]

Abstract

Quantization techniques commonly reduce the inference costs of neural networks by restricting the precision of weights and activations. Recent studies show that also reducing the precision of the accumulator can further improve hardware efficiency at the risk of numerical overflow, which introduces arithmetic errors that can degrade model accuracy. To avoid numerical overflow while maintaining accuracy, recent work proposed accumulator-aware quantization (A2Q)—a quantization-aware training method that constrains model weights during training to safely use a target accumulator bit width during inference. Although this shows promise, we demonstrate that A2Q relies on an overly restrictive constraint and a sub-optimal weight initialization strategy that each introduce superfluous quantization error. To address these shortcomings, we introduce: (1) an improved bound that alleviates accumulator constraints without compromising overflow avoidance; and (2) a new strategy for initializing quantized weights from pre-trained floating-point checkpoints. We combine these contributions with weight normalization to introduce A2Q+. We identify and characterize the various trade-offs that arise as a consequence of accumulator constraints and support our analysis with experiments that show A2Q+ significantly improves these trade-offs when compared to prior methods.

1. Introduction

Quantizing neural network weights and activations to low-precision integers can drastically reduce the inference costs of multiplications. However, the resulting products are commonly accumulated at high-precision and thus require high-precision additions and registers. Recent studies show that reducing the standard 32-bit accumulators to 16 bits can yield a near-optimal $2\times$ increase in throughput and bandwidth efficiency on ARM processors (de Bruin et al., 2020; Xie et al., 2021) and up to a $1.6\times$ reduction in resource utilization on FPGAs (Colbert et al., 2023). However, exploiting such an optimization is highly non-trivial in practice as doing so also incurs a high risk of numerical overflow, which introduces arithmetic errors that can significantly degrade model accuracy (Ni et al., 2021).

To train quantized neural networks (QNNs) for low-precision accumulation, Colbert et al. 2023 recently proposed accumulator-aware quantization (A2Q). While prior approaches had sought to either reduce the risk of numerical overflow (Xie et al., 2021; Li et al., 2022; Azamat et al., 2022) or mitigate its impact on model accuracy (Ni et al., 2021; Blumenfeld et al., 2023), A2Q circumvents arithmetic errors caused by numerical overflow by constraining model weights to restrict the range of outputs. In doing so, A2Q provides state-of-the-art performance for low-precision accumulation with guaranteed overflow avoidance.

Our work contributes to this body of research by further improving the trade-off between accumulator bit width and model accuracy. We show that A2Q relies on: (1) an overly restrictive ℓ_1-norm bound that constrains QNNs more than necessary; and (2) a sub-optimal initialization strategy that forces QNNs to recover from superfluous quantization error. In addressing these shortcomings, we establish a new state-of-the-art for low-precision accumulation with guaranteed overflow avoidance. Our results show for the first time that ResNet50 (He et al., 2016) can maintain 95% of its baseline accuracy when trained on ImageNet (Deng et al., 2009) to accumulate at 12 bits without overflow, resulting in a $+17\%$ improvement in test top-1 accuracy over A2Q.

Our contributions are four-fold: (1) we introduce a new theoretical analysis for an improved ℓ_1-norm bound that alleviates accumulator constraints without compromising overflow avoidance; (2) we introduce a weight initialization strategy that minimizes the initial weight quantization error caused by accumulator constraints; (3) we combine (1) and (2) with weight normalization (Salimans & Kingma, 2016) to introduce A2Q+ and show significant improvements in the trade-off between accumulator bit width and model accuracy; and (4) we identify and characterize various trade-offs that arise as a consequence of accumulator constraints.

[1]AMD SW Technology Team, San Diego, California, USA [2]AMD Research and Advanced Development, Dublin, Ireland. Correspondence to: Ian Colbert <ian.colbert@amd.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

2. Background and Related Work

2.1. Low-Precision Accumulation

Neural network primitives are commonly executed as dot products consisting of numerous multiply-accumulate (MAC) operations. During inference, the inputs to these dot products (*i.e.*, the weights and activations) are increasingly being represented with lower precision integers to reduce the cost of multiplications; meanwhile, their products are still accumulated using high-precision additions.

The skew towards weight and activation quantization is in large part because the most commonly studied data formats in deep learning inference have required 8 or more bits (Wu et al., 2020; Gholami et al., 2021). Because the cost of integer MACs scales quadratically with the bit widths of the weights and activations but linearly with that of the accumulator (Horowitz, 2014; Blott et al., 2018; Hawks et al., 2021), the cost of multiplications dwarfs that of additions in such paradigms. However, with even lower precision data formats increasing in popularity (Aggarwal et al., 2023; Wu et al., 2023), ignoring the accumulator to solely focus on low-precision weight and activation quantization will have diminishing returns. For example, Ni et al. 2021 show that when constraining weights and activations to 3-bit $\times$ 1-bit multipliers, the cost of 32-bit accumulation dominates that of multiplication, consuming nearly 75% of the total power and 90% of the total area of their MAC unit. When reducing to an 8-bit accumulator, they report $4\times$ power savings and $5\times$ area reduction. In addition to power and area, recent work has also demonstrated savings in throughput and bandwidth utilization when reducing the accumulator bit width on general-purpose platforms (de Bruin et al., 2020; Xie et al., 2021). Both de Bruin et al. 2020 and Xie et al. 2021 report a near-optimal $2\times$ increase in throughput on computer vision workloads when reducing the accumulator width from 32 to 16 bits on ARM processors.

Exploiting such an optimization in a principled manner is non-trivial in practice. The risk of numerical overflow increases exponentially as the accumulator bit width is reduced (Colbert et al., 2023). The resulting arithmetic errors can lead to catastrophic degradation in model accuracy if the accumulator is not large enough (Ni et al., 2021).

2.2. Accumulator-Aware Quantization (A2Q)

Training neural networks with quantization in the loop is a useful means of recovering model accuracy lost to quantization errors (Gholami et al., 2021; Wu et al., 2020). The standard operators used to emulate quantization during training are built on top of uniform affine transformations that map high-precision values to low-precision ones. We refer to the operators that perform these transformations as quantizers. As given by Eq. 1, quantizers are commonly parameterized by zero-point z and scaling factor s. Here, z is an integer value that ensures that zero is exactly represented in the quantized domain, and s is a strictly positive real scalar that corresponds to the resolution of the mapping. Scaled values are commonly rounded to the nearest integers using half-way rounding, denoted by $\lfloor \cdot \rceil$, and elements that exceed the largest supported values in the quantized domain are clipped to n and p, which depend on the target bit width b. We assume $n = -2^{b-1}$ and $p = 2^{b-1} - 1$ when signed, and $n = 0$ and $p = 2^b - 1$ when unsigned.

$$Q(\boldsymbol{w}) := s \cdot \left(\text{clip}(\left\lfloor \frac{\boldsymbol{w}}{s} \right\rceil + z; n, p) - z \right) \tag{1}$$

One approach to training QNNs for low-precision accumulation is to mitigate the impact of numerical overflow on model accuracy during QAT. To do so, researchers have sought to either tune scale factors to control overflow rates (Xie et al., 2021; Azamat et al., 2022; Li et al., 2022) or train QNNs to be robust to wraparound arithmetic (Ni et al., 2021; Blumenfeld et al., 2023). However, empirical estimates of overflow rely on *a priori* knowledge of the input distribution, which is impractical to assume in many real-world use cases and can even introduce vulnerabilities (Baier et al., 2019). Thus, as an alternative, Colbert et al. 2023 proposed accumulator-aware quantization (A2Q) to directly train QNNs to use low-precision accumulators during inference without any risk of numerical overflow.

A2Q guarantees overflow avoidance by constraining the ℓ_1-norm of weights to restrict the range of dot product outputs. To accomplish this, Colbert et al. 2023 introduce a quantizer inspired by weight normalization (Salimans & Kingma, 2016) that re-parameterizes weights $\boldsymbol{w}$ into vector $\boldsymbol{v}$ and scalar g such that $\boldsymbol{w} = g \cdot \boldsymbol{v}/\|\boldsymbol{v}\|_1$. This allows the ℓ_1-norm of $\boldsymbol{w}$ to be learned as an independent parameter since $g = \|\boldsymbol{w}\|_1$. To avoid numerical overflow during inference, Colbert et al. 2023 constrain g according to a derived upper bound T so that $\|Q(\boldsymbol{w})\|_1 \leq T$, as further discussed in Section 3.3. The resulting quantizer is defined as:

$$Q(\boldsymbol{w}) := s \cdot \text{clip}\left(\left\lfloor \frac{\boldsymbol{w}}{s} \right\rfloor ; n, p \right) \tag{2}$$

$$\text{where } \boldsymbol{w} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_1} \cdot \min(g, T) \tag{3}$$

$$\text{and } T = s \cdot \frac{2^{P-1} - 1}{2^{N - \mathbb{1}_{\text{signed}}(\boldsymbol{x})}} \tag{4}$$

Here, P denotes the target accumulator bit width, N denotes the input activation bit width, and $\mathbb{1}_{\text{signed}}(\boldsymbol{x})$ is an indicator function that returns 1 when input activations are signed and 0 when unsigned. Unlike in Eq. 1, scaled weights are rounded towards zero (Loroch et al., 2017), denoted by $\lfloor \cdot \rfloor$, to prevent any upward rounding that may cause $\|Q(\boldsymbol{w})\|_1$ to increase past the derived upper bound T. Each output channel is assumed to have its own accumulator so g is independently defined and constrained per-channel.

3. A2Q+

Let weights $\boldsymbol{q}$ be a K-dimensional vector of M-bit integers, and let $\mathbb{Z}_N^K$ denote the set of all K-dimensional vectors of N-bit integers. When accumulating the dot product of $\boldsymbol{q}$ by any $\boldsymbol{x} \in \mathbb{Z}_N^K$ into a signed P-bit register, Colbert et al. 2023 show that one can avoid overflow if $\boldsymbol{q}$ satisfies:

$$\|\boldsymbol{q}\|_1 \leq \frac{2^{P-1}-1}{2^{N-\mathbb{1}_{\text{signed}}(\boldsymbol{x})}} \tag{5}$$

Irrespective of weight bit width M, Eq. 5 establishes an upper bound on the ℓ_1-norm of $\boldsymbol{q}$ as a function of accumulator bit width P and activation bit width N. For fixed N, reducing P exponentially tightens the constraint on $\|\boldsymbol{q}\|_1$, which restricts the range of the weights by pulling them towards zero. Colbert et al. 2023 demonstrate that learning under such a constraint introduces a trade-off in the form of a Pareto frontier, where reducing the accumulator bit width invariably limits model accuracy within a fixed quantization design space. We observe that this bound also introduces a non-trivial trade-off between activation bit width N and model accuracy. Reducing the precision of the activations alleviates pressure on $\|\boldsymbol{q}\|_1$, which becomes more significant as P is reduced. However, aggressive discretization of intermediate activations can significantly hurt model accuracy (Wu et al., 2020; Gholami et al., 2021). In Section 4.1, we show that balancing this trade-off results in the Pareto-optimal activation bit width N decreasing with P.

Rather than tackling this balancing act (which is an intriguing problem for future work), our work extends the approach of A2Q to directly improve these trade-offs. We demonstrate that A2Q relies on an overly restrictive constraint and a sub-optimal weight initialization strategy that each introduce superfluous quantization errors. In Section 4, we show that minimizing these errors ultimately leads to improved model accuracy as the accumulator bit width is reduced.

3.1. Improved ℓ_1-norm Bound via Zero-Centering

Let the closed interval $[a, b]$ denote the representation range of a signed P-bit register. To avoid overflow when accumulating $\boldsymbol{x}^T\boldsymbol{q}$ into this register, the dot product output needs to fall within $[a, b]$ for any $\boldsymbol{x} \in \mathbb{Z}_N^K$. Without loss of generality, we assume a two's complement representation in our work, where $[a, b] = [-2^{P-1}, 2^{P-1}-1]$, as is common practice (Wu et al., 2020; Gholami et al., 2021).

Colbert et al. 2023 approach this task by constraining the magnitude of $\boldsymbol{x}^T\boldsymbol{q}$ such that $|\boldsymbol{x}^T\boldsymbol{q}| \leq 2^{P-1}-1$. They use worst-case values for $\boldsymbol{x}$ to derive the upper bound on $\|\boldsymbol{q}\|_1$ given by Eq. 5. Note that this bound can similarly be constructed via Hölder's inequality (Hardy et al., 1952) as shown in Eq. 6, where $\|\boldsymbol{x}\|_\infty = 2^{N-\mathbb{1}_{\text{signed}}(\boldsymbol{x})}$.

$$|\boldsymbol{x}^T\boldsymbol{q}| \leq \|\boldsymbol{x}\|_\infty \|\boldsymbol{q}\|_1 \leq 2^{P-1}-1 \tag{6}$$

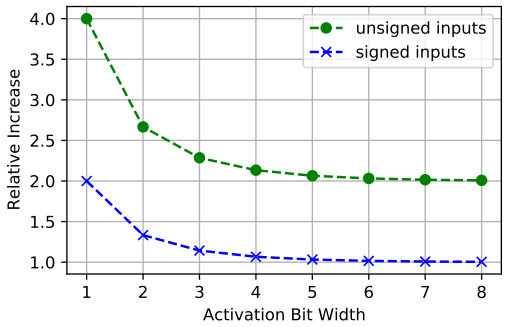


Figure 1. We visualize Eq. 8 for both signed (**blue crosses**) and unsigned (**green circles**) integers to show the relative increase in ℓ_1-norm budget that our new bound (Eq. 7) gives to $\boldsymbol{q}$ when compared to the standard A2Q bound (Eq. 5).

Note that this bound has two shortcomings: (1) it does not make full use of the representation range of the accumulator, which becomes increasingly important as its bit width P is reduced; and (2) it depends on the sign of $\boldsymbol{x}$, which tightens the constraint by $2\times$ when $\mathbb{1}_{\text{signed}}(\boldsymbol{x}) = 0$. In this work, we resolve both of these. We find that zero-centering our weight vector such that $\sum_i w_i = 0$ yields a favorable property, as formally presented in the following proposition.

Proposition 3.1. *Let $\boldsymbol{x}$ be a K-dimensional vector of N-bit integers such that the value of the i-th element x_i lies within the closed interval $[c, d]$ and $d - c = 2^N - 1$. Let $\boldsymbol{q}$ be a K-dimensional vector of signed integers centered at zero such that $\sum_i q_i = 0$. To guarantee overflow avoidance when accumulating the result of $\boldsymbol{x}^T\boldsymbol{q}$ into a signed P-bit register, it is sufficient that the ℓ_1-norm of $\boldsymbol{q}$ satisfies:*

$$\|\boldsymbol{q}\|_1 \leq \frac{2^P - 2}{2^N - 1} \tag{7}$$

The proof of this proposition is provided in Appendix A.1. It is important to note that our new bound (Eq. 7) utilizes the full representation range of the accumulator and is agnostic to the sign of the input data. Furthermore, when compared to the original bound (Eq. 5), ours is greater by a factor of:

$$\frac{2^{N+1-\mathbb{1}_{\text{signed}}(\boldsymbol{x})}}{2^N - 1} \tag{8}$$

In Fig. 1, we visualize this relationship as a function of activation bit width N. As implied by Eq. 8, the impact of our bound increases as the activation bit width is reduced, with the greatest significance in sub-4-bit quantization scenarios. In fact, our bound yields up to a $4\times$ increase in the ℓ_1-norm budget afforded to $\boldsymbol{q}$ when input activations are unsigned (*i.e.*, $\mathbb{1}_{\text{signed}}(\boldsymbol{x}) = 0$), and up to $2\times$ when they are signed (*i.e.*, $\mathbb{1}_{\text{signed}}(\boldsymbol{x}) = 1$). In Section 4, we show that this increased freedom significantly improves model accuracy.

3.2. Improved Initialization via Euclidean Projections

By re-parameterizing weight vector $\boldsymbol{w}$ as defined below in Eq. 9, A2Q introduces two new parameters to initialize: g and $\boldsymbol{v}$. However, because $\boldsymbol{w}$ is a function of these learned parameters, it can no longer be directly initialized from a pre-trained floating-point checkpoint.

$$\boldsymbol{w} = g \cdot \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_1} \tag{9}$$

One could trivially initialize $\boldsymbol{v}$ to be the pre-trained floating-point weight vector $\boldsymbol{w}_{\text{float}}$ and g to be its ℓ_1-norm, where $\boldsymbol{v} = \boldsymbol{w}_{\text{float}}$ and $g = \|\boldsymbol{w}_{\text{float}}\|_1$, making $\boldsymbol{w} = \boldsymbol{w}_{\text{float}}$. However, A2Q clips g according to T in Eq. 3. As a consequence, we observe that naïvely initializing g and $\boldsymbol{v}$ according to $\boldsymbol{w}_{\text{float}}$ introduces excessive weight quantization error when $\|\boldsymbol{w}_{\text{float}}\|_1 > T$ (see Appendix B.3). Thus, we aim to minimize weight quantization error at initialization.

We formulate our objective as a projection task described by the constrained convex optimization problem in Eq. 10. Here, the optimal initialization $\boldsymbol{v}^*$ minimizes the weight quantization error while satisfying the ℓ_1-norm accumulator constraint on the re-scaled quantized weights $Q(\boldsymbol{w})$.

$$\boldsymbol{v}^* = \min_{\boldsymbol{v}} \frac{1}{2}\|Q(\boldsymbol{w}) - \boldsymbol{w}_{\text{float}}\|_2^2 \tag{10}$$

$$\text{subject to } \|Q(\boldsymbol{w})\|_1 \leq T \tag{11}$$

$$\text{where } \boldsymbol{w} = g \cdot \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_1} \tag{12}$$

To solve this optimization problem, we exploit the round-to-zero operator, which ensures that the magnitude of any weight w_i is always greater than or equal to that of its quantized counterpart $Q(w_i)$, or more formally $|Q(w_i)| \leq |w_i|$ for all i. This allows us to solely focus on initializing $\boldsymbol{v}$ such that $\|\boldsymbol{v}\|_1 \leq T$ and then initialize g such that $g = \|\boldsymbol{v}\|_1$. Thus, we can simplify our optimization problem to:

$$\boldsymbol{v}^* = \min_{\boldsymbol{v}} \frac{1}{2}\|\boldsymbol{v} - \boldsymbol{w}_{\text{float}}\|_2^2 \tag{13}$$

$$\text{subject to } \|\boldsymbol{v}\|_1 \leq T \tag{14}$$

It is important to first note that if $\|\boldsymbol{w}_{\text{float}}\|_1 \leq T$, then the optimal solution to Eq. 13 is trivially $\boldsymbol{v}^* = \boldsymbol{w}_{\text{float}}$. In addition, when $\|\boldsymbol{w}_{\text{float}}\|_1 > T$, the optimal solution $\boldsymbol{v}^*$ lies on the boundary of the constrained set such that $\|\boldsymbol{v}^*\|_1 = T$. This allows us leverage the optimal solution derived in Duchi et al. 2008, which efficiently projects $\boldsymbol{w}_{\text{float}}$ onto an ℓ_1-ball of radius T using Eq. 15.

$$\boldsymbol{v}^* = \text{sign}(\boldsymbol{w}_{\text{float}}) \left(|\boldsymbol{w}_{\text{float}}| - \theta\right)_+ \tag{15}$$

Here, $(\cdot)_+$ denotes the rectified linear unit, which zeroes out all negative values, and θ is a Lagrangian scalar derived from the optimal solution. We direct the reader to Duchi et al. 2008 for the associated proofs and derivations.

3.3. Constructing A2Q+

Similar to A2Q, our quantizer is inspired by weight normalization (Salimans & Kingma, 2016) and leverages the re-parameterization given in Eq. 9. However, unlike A2Q, we are unable to simply constrain scalar parameter g according to Eq. 7 because Prop. 3.1 relies on the assumption that $Q(\boldsymbol{w})$ is zero-centered such that $\sum_i Q(w_i) = 0$, which is not inherently guaranteed.

Enforcing such a zero-centering constraint on a vector of integers is non-trivial in practice. Emulating quantization during QAT adds to this complexity as integer-quantized weights $Q(\boldsymbol{w})$ are a function of floating-point counterpart $\boldsymbol{w}$. However, A2Q is able to guarantee the ℓ_1-norm constraint on $Q(\boldsymbol{w})$ by constraining norm parameter g, then rounding the scaled floating-point weights $\boldsymbol{w}/s$ towards zero, which ensures that $\|Q(\boldsymbol{w})\|_1 \leq \|\boldsymbol{w}\|_1$. We similarly exploit this property to enforce our zero-centering constraint, as formally articulated in the following proposition.

Proposition 3.2. *Let $\boldsymbol{x}$ be a vector of N-bit integers such that the i-th element x_i lies within the closed interval $[c, d]$ and $d - c = 2^N - 1$. Let $\boldsymbol{w}$ be a zero-centered vector such that $\sum_i w_i = 0$. Let $Q(\boldsymbol{w})$ be a symmetric quantizer parameterized in the form of Eq. 2, where $Q(\boldsymbol{w}) = s \cdot \boldsymbol{q}$ for strictly positive scaling factor s and integer-quantized weight vector $\boldsymbol{q}$. Given that $\text{sign}(s \cdot q_i) = \text{sign}(w_i)$ and $|s \cdot q_i| \leq |w_i|$ for all i, then $\boldsymbol{x}^T\boldsymbol{q}$ can be safely accumulated into a signed P-bit register without overflow if $\boldsymbol{w}/s$ satisfies all necessary conditions for such a constraint.*

The proof of this proposition, as well as a formal definition of the necessary accumulator constraint conditions, is provided in Appendix A.2. Based on Prop. 3.2, we are able to enforce our zero-centering constraint on $\boldsymbol{w}$ without compromising overflow avoidance, assuming we maintain a symmetric quantizer that rounds towards zero. However, rather than directly zero-centering $\boldsymbol{w}$ when leveraging the re-parameterization given by Eq. 9, we enforce our constraint on $\boldsymbol{v}$ so as to control its ℓ_1-norm. Given that $\gamma = g/\|\boldsymbol{v}\|_1$ and $\gamma\boldsymbol{v} = \boldsymbol{w}$, it follows that $\sum_i w_i = 0$ when $\sum_i v_i = 0$. Thus, we construct our quantizer as follows:

$$Q(\boldsymbol{w}) := s \cdot \text{clip}\left(\left\lfloor \frac{\boldsymbol{w}}{s} \right\rfloor; n, p\right) \tag{16}$$

$$\text{where } \boldsymbol{w} = \frac{\boldsymbol{v} - \mu_{\boldsymbol{v}}}{\|\boldsymbol{v} - \mu_{\boldsymbol{v}}\|_1} \cdot \min(g, T_+) \tag{17}$$

$$\text{and } \mu_{\boldsymbol{v}} = \frac{1}{K}\sum_{i=1}^{K} v_i \tag{18}$$

$$\text{and } T_+ = s \cdot \frac{2^P - 2}{2^N - 1} \tag{19}$$

To maintain a symmetric quantizer, we eliminate the zero points in our mapping such that $z = 0$. We also use an ex-

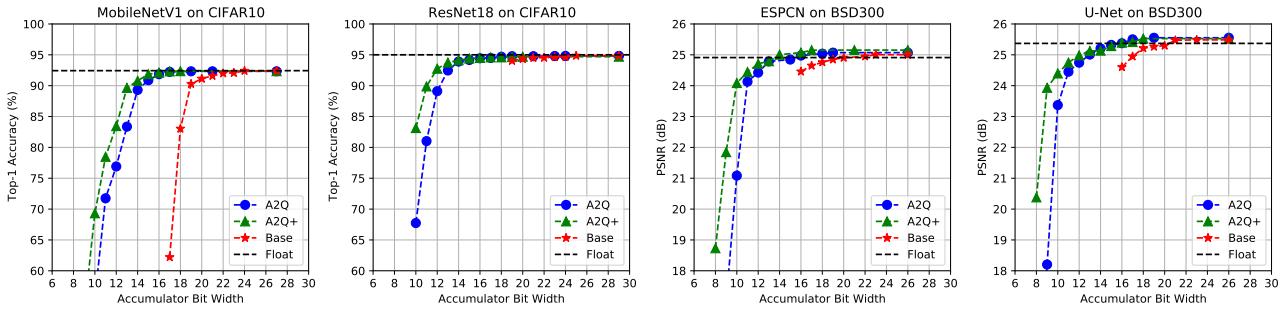


Figure 2. We visualize the trade-off between accumulator bit width and model accuracy using Pareto frontier. We observe that A2Q+ (green triangles) dominates both A2Q (blue circles) and the baseline QAT (red stars) in all benchmarks.

ponential parameterization of both the scaling factor $s = 2^d$ and norm parameter $g = 2^t$, where d and t are defined per-output channel and learned through gradient descent. Note that norm parameter g is clipped to T_+, which is our new upper bound defined in Eq. 7 scaled by s to ensure $\|\boldsymbol{w}/s\|_1 \leq T_+$. Our scaled floating-point weights are then rounded towards zero, denoted by $\lfloor \cdot \rfloor$. The rounded weights are then clipped and re-scaled. When updating learnable parameters throughout training, we use the straight-through estimator (Bengio et al., 2013) to allow gradients to permeate the rounding function, where $\nabla_x \lfloor x \rfloor = 1$ everywhere and ∇_x denotes the gradient with respect to x.

Extending our Euclidean projection-based initialization strategy to A2Q+ is non-trivial in practice. In such a scenario, our optimization problem is instead subject to the following constraint: $\|\boldsymbol{v} - \mu_{\boldsymbol{v}}\|_1 \leq T_+$. Furthermore, the optimal solution derived by Duchi et al. 2008 requires each non-zero component of the optimal solution $\boldsymbol{v}^*$ to share the same sign as its counterpart in $\boldsymbol{w}_{\text{float}}$, which is not inherently guaranteed due to our zero-centering constraint. Therefore, we initialize all A2Q+ networks using the A2Q initialization in the scope of this work. In practice, we observe this still significantly reduces initial weight quantization error for A2Q+ networks. In Appendix B.3, we provide a deeper investigation for both A2Q and A2Q+ networks.

4. Experimental Results

Models & Datasets. Throughout our experiments, we focus on two computer vision tasks: image classification and single-image super resolution. In Section 4.1, we evaluate MobileNetV1 (Howard et al., 2017) and ResNet18 (He et al., 2016) trained on the CIFAR-10 dataset (Krizhevsky et al., 2009) for image classification, and ESPCN (Shi et al., 2016) and U-Net (Ronneberger et al., 2015) trained on the BSD300 dataset (Martin et al., 2001) for super resolution. In Section 4.2, we evaluate larger image classification benchmarks, namely ResNet18, ResNet34, and ResNet50 trained on the ImageNet-1K dataset (Deng et al., 2009).

Quantization Design Space. Following the experiments of Colbert et al. 2023, we constrain our quantization design space to uniform-precision models such that every hidden layer has the same weight, activation, and accumulator bit width, respectively denoted as M, N, and P. Our experiments consider 3- to 8-bit integers for both weights and activations, extending the quantization design space of Colbert et al. 2023 by $4\times$. For each of the 64 weight and activation combinations, we calculate the most conservative accumulator bit width for each model using Eq. 20, as derived by Colbert et al. 2023. Here, $\lceil \cdot \rceil$ denotes ceiling rounding and $K^* = \arg\max_{K_l} \{K_l\}_{l=1}^{L}$, where K_l is the dot product size of layer l in a network with L layers. We calculate P^* for each unique (M, N) combination and evaluate up to a 10-bit reduction in accumulator bit width, creating a total of 640 unique configurations per model. We repeat each experiment 3 times using different random seeds.

$$P^* = \lceil \alpha + \phi(\alpha) + 1 \rceil \tag{20}$$

$$\alpha = \log_2(K^*) + N + M - 1 - \mathbb{1}_{\text{signed}}(\boldsymbol{x}) \tag{21}$$

$$\phi(\alpha) = \log_2(1 + 2^{-\alpha}) \tag{22}$$

We implement A2Q+ in PyTorch (Paszke et al., 2019) using v0.10 of the Brevitas quantization library (Pappalardo, 2021) and leverage their implementations of A2Q and baseline QAT methods for benchmarking. We include all training details and hyperparameters in Appendix B.1.

4.1. Optimizing for Accumulator Constraints

Following the benchmarking strategy in Colbert et al. 2023, we first optimize QNNs for accumulator-constrained processors. Here, the goal is to maximize model accuracy[1] given a target accumulator bit width P. This scenario has implications for accelerating inference on general-purpose platforms (Xie et al., 2021; Li et al., 2022) and reducing the computational overhead of encrypted computations (Lou

[1]We loosely use the term *model accuracy* to also describe peak signal-to-noise ratio (PSNR) for convenience of discussion.

Table 1. We provide the test top-1 accuracy and quantization configuration for some of the Pareto-optimal image classification models that form a section of the frontiers visualized in Fig. 2. We emphasize the Pareto-dominant points in **bold**.

P	MobileNetV1 (Float: 92.43%)						ResNet18 (Float: 95.00%)					
	Base		A2Q		A2Q+		Base		A2Q		A2Q+	
	Top-1	(M, N)	Top-1	(M, N)	Top-1	(M, N)	Top-1	(M, N)	Top-1	(M, N)	Top-1	(M, N)
11	-	-	71.8%	(5,4)	**78.5%**	**(4,5)**	-	-	81.0%	(3,3)	**89.9%**	**(3,3)**
12	-	-	76.9%	(4,5)	**83.5%**	**(4,6)**	-	-	89.1%	(3,3)	**92.8%**	**(3,3)**
13	-	-	83.4%	(6,5)	**89.7%**	**(3,6)**	-	-	92.5%	(3,3)	**93.8%**	**(4,3)**
14	-	-	89.3%	(3,6)	**90.8%**	**(3,7)**	-	-	93.9%	(4,3)	**94.1%**	**(4,4)**
15	-	-	90.9%	(4,7)	**91.9%**	**(4,7)**	-	-	94.2%	(4,4)	**94.4%**	**(5,5)**
16	-	-	91.9%	(4,7)	**92.1%**	**(5,8)**	-	-	94.4%	(4,4)	**94.5%**	**(3,5)**
17	62.2%	(3,3)	92.2%	(4,8)	**92.3%**	**(6,8)**	-	-	94.5%	(5,5)	**94.6%**	**(6,6)**
18	83.0%	(3,4)	92.2%	(4,8)	**92.4%**	**(6,8)**	-	-	**94.7%**	**(4,5)**	94.6%	(6,7)
19	90.3%	(3,5)	92.3%	(6,8)	**92.4%**	**(6,8)**	94.0%	(3,3)	**94.8%**	**(4,7)**	94.7%	(6,8)
20	91.1%	(3,6)	92.3%	(6,8)	**92.4%**	**(8,8)**	94.3%	(3,4)	**94.8%**	**(4,7)**	94.7%	(8,8)

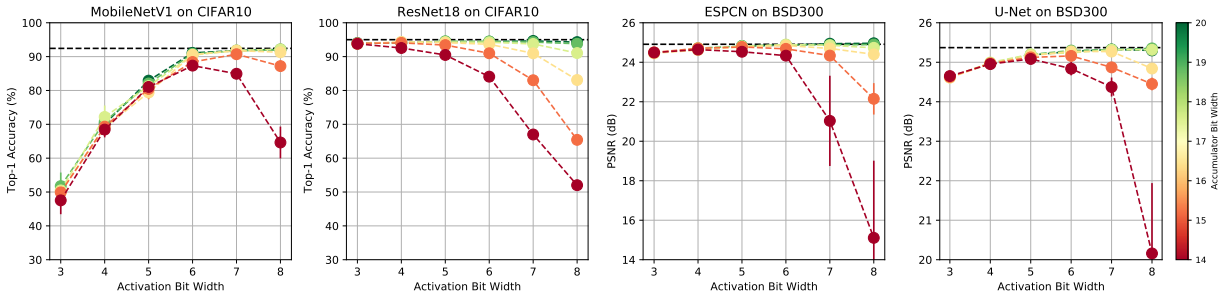
Table 2. We provide the test peak signal-to-noise ratio (PSNR) and quantization configuration for some of the Pareto-optimal super resolution models that form a section of the frontiers visualized in Fig. 2. We emphasize the Pareto-dominant points in **bold**.

P	ESPCN (Float: 24.91)						U-Net (Float: 25.37)					
	Base		A2Q		A2Q+		Base		A2Q		A2Q+	
	PSNR	(M, N)	PSNR	(M, N)	PSNR	(M, N)	PSNR	(M, N)	PSNR	(M, N)	PSNR	(M, N)
9	-	-	17.0	(4,3)	**21.9**	**(3,3)**	-	-	18.2	(5,3)	**23.9**	**(5,3)**
10	-	-	21.1	(4,3)	**24.1**	**(4,3)**	-	-	23.4	(4,3)	**24.4**	**(3,3)**
11	-	-	24.1	(6,3)	**24.4**	**(4,3)**	-	-	24.5	(6,3)	**24.8**	**(5,4)**
12	-	-	24.4	(7,3)	**24.7**	**(4,4)**	-	-	24.7	(4,4)	**25.0**	**(3,5)**
13	-	-	**24.8**	**(7,4)**	**24.8**	**(5,5)**	-	-	25.0	(8,4)	**25.1**	**(7,5)**
14	-	-	24.8	(7,4)	**25.0**	**(6,5)**	-	-	**25.2**	**(8,5)**	**25.2**	**(8,5)**
15	-	-	24.9	(4,5)	**25.0**	**(6,5)**	-	-	**25.3**	**(8,6)**	**25.3**	**(6,6)**
16	24.5	(3,3)	25.0	(6,6)	**25.1**	**(6,7)**	24.6	(3,3)	**25.4**	**(8,6)**	**25.4**	**(6,6)**
17	24.7	(3,4)	25.0	(6,6)	**25.2**	**(8,7)**	25.0	(3,4)	**25.5**	**(4,8)**	**25.5**	**(6,8)**
18	24.8	(3,5)	25.0	(6,7)	**25.2**	**(8,7)**	25.2	(3,5)	**25.5**	**(4,8)**	**25.5**	**(6,8)**

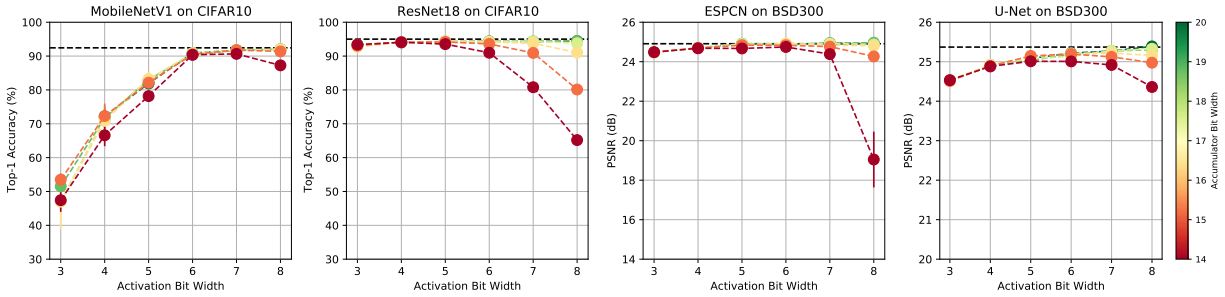
& Jiang, 2019; Stoian et al., 2023). As an alternative to A2Q, one could also heuristically manipulating weight bit width M and activation bit width N according to Eq. 20. To the best of our knowledge, this is the only other method to train a uniform-precision QNN for a given P without overflow. Therefore, we use exhaustive bit width manipulation as a baseline when comparing A2Q+ against A2Q.

In Fig. 2, we visualize this comparison using Pareto frontiers and provide the 32-bit floating-point model accuracy for reference. For each model and each QAT algorithm, the Pareto frontier provides the maximum observed model accuracy for a given target accumulator bit width P. In addition, we provide a detailed breakdown of each Pareto frontier in Tables 1 and 2, where we also report the weight and activation bit widths of the Pareto-dominant model. In these experiments, all super resolution benchmarks are trained from scratch and all image classification benchmarks are initialized from pre-trained floating-point checkpoints using our Euclidean projection-based weight initialization (EP-init). We handle depthwise separable convolutions using the technique discussed in Appendix B.2, which only impacts MobileNetV1. It is important to note that this is not a direct comparison against Colbert et al. 2023 because we apply EP-init to both A2Q and A2Q+ models to strictly compare weight quantizers. However, we provide an ablation study in Appendix B.3 that shows EP-init improves both A2Q and A2Q+ by up to $+50\%$ in extremely low-precision accumulation regimes.

Intuitively, heuristic bit width manipulations can only reduce the accumulator bit width so far because P is ultimately limited by dot product size K. Alternatively, using A2Q to train QNNs directly for low-precision accumulation allows one to push the accumulator bit width lower than previously attainable without compromising overflow; yet, a trade-off still remains. We observe that A2Q+ significantly improves this trade-off, especially in the extremely low-precision accumulation regime. Thus, by alleviating the pressure on model weights, A2Q+ recovers model accuracy lost to the overly restrictive accumulator constraints imposed by A2Q.



(a) A2Q



(b) A2Q+

Figure 3. We evaluate the trade-off between activation bit width N and model accuracy under fixed accumulator constraints. We visualize the average and standard deviation in model accuracy measured over 3 experiments as N is increased from 3 to 8 bits when targeting accumulator widths that range from 14 to 20 bits. The weights of all hidden layers are fixed to 4-bits.

Finally, we observe that the Pareto-optimal activation bit width N monotonically decreases along the frontier as the target accumulator bit width P is reduced. We hypothesize this is in part a consequence of the alleviated pressure on $\|\boldsymbol{q}\|_1$ discussed in Section 3. To investigate this relationship, we evaluate model accuracy as we increase N with fixed P. To focus on N and P, we fix weights to 4 bits. Figure 3 shows the average accuracy for each model as we increase N from 3 to 8 bits when targeting 14- to 20-bit accumulation. As previously established, reducing P invariably limits model accuracy. However, additionally increasing N continues to tighten the constraint on $\|\boldsymbol{q}\|_1$, further limiting model accuracy and introducing a non-trivial trade-off observed across neural architectures. When compared to A2Q, A2Q+ significantly alleviates this trade-off by alleviating the constraints on $\|\boldsymbol{q}\|_1$ for fixed N and P, increasing the Pareto-optimal activation bit width and model accuracy.

4.2. Low-Precision Accumulation for ImageNet Models

The ℓ_1-norm of an unconstrained weight vector inherently grows as its dimensionality K increases. This suggests that, with a fixed activation bit width N and target accumulator bit width P, A2Q and A2Q+ scale well to deeper architectures as the accumulator constraint tightens with the width of a neural architecture rather than the depth.

We investigate this hypothesis by evaluating larger ResNet models trained on ImageNet from pre-trained floating-point checkpoints. Rather than exploring the full quantization design space, we focus on 4-bit weights and activations while evaluating A2Q and A2Q+ under various accumulator constraints. We also evaluate the impact of our Euclidean projection-based weight initialization strategy (EP-init) on standard A2Q and use the standard methods discussed in Appendix B.1 to provide a reference QAT baseline. We use the pre-trained checkpoints provided by PyTorch (Paszke et al., 2019) and report our results in Table 3.

We observe that both A2Q and A2Q+ can maintain baseline accuracy when targeting 16-bit accumulators; however, it is important to note that this is a non-trivial result. Only about 50% of the output channels in the PyTorch ResNet18 and ResNet34 checkpoints inherently satisfy a 16-bit accumulator constraint, with less than 10% satisfying 12-bit constraints (see Appendix B.3). While A2Q is able to recover when $P = 16$, we observe that EP-init significantly improves model accuracy as P is reduced, with a noteable $+11.7\%$ increase in test top-1 accuracy on ResNet50 when targeting 12-bit accumulation. Furthermore, we observe that A2Q+ can consistently maintain over 96% of the test top-1 accuracy relative to the 32-bit floating-point baselines when targeting 14-bit accumulators.

Interestingly, we observe that the accuracy gap between accumulator-constrained models and their original floating-point counterparts decreases as model size increases. Building from our hypothesis, we conjecture this is in part because the models are growing in depth but not width, which increases model capacity without tightening our constraints.

Finally, we observe that both A2Q and A2Q+ inherently expose opportunities to exploit unstructured weight sparsity. As shown in Colbert et al. 2023, decreasing P increases sparsity when N is fixed. Furthermore, since A2Q is more restrictive than A2Q+ for fixed P and N, we see that A2Q can result in significantly higher sparsity levels. We observe this gap in sparsity decreases with P while the accuracy gap increases, with A2Q+ resulting in $+17\%$ top-1 accuracy with only -6.3% sparsity when compared to A2Q for 12-bit accumulator constraints on ResNet50.

Table 3. We evaluate A2Q+ for W4A4 ImageNet models and compare against baseline QAT methods and standard A2Q, both with and without Euclidean projection-based initialization (EP-init.)

Network	Method	P	Top-1	Sparsity
ResNet18 (Float: 69.76%)	Base	32	70.2%	20.8%
	A2Q	16	69.2%	73.7%
		14	60.3%	91.7%
		12	35.5%	94.7%
	A2Q (w/ EP-init)	16	69.3%	73.7%
		14	62.5%	91.2%
		12	42.7%	94.6%
	A2Q+	**16**	**69.8%**	**50.4%**
		14	**67.1%**	**85.0%**
		12	**56.4%**	**93.4%**
ResNet34 (Float: 73.31%)	Base	32	73.4%	23.9%
	A2Q	16	73.1%	75.2%
		14	65.8%	94.5%
		12	43.4%	96.9%
	A2Q (w/ EP-init)	16	73.1%	74.9%
		14	67.6%	94.2%
		12	51.4%	96.8%
	A2Q+	**16**	**73.3%**	**51.4%**
		14	**71.4%**	**85.0%**
		12	**62.1%**	**95.9%**
ResNet50 (Float: 76.13%)	Base	32	75.9%	25.8%
	A2Q	16	76.0%	56.1%
		14	73.8%	77.2%
		12	55.0%	90.7%
	A2Q (w/ EP-init)	16	76.0%	56.1%
		14	74.5%	77.1%
		12	66.7%	88.6%
	A2Q+	**16**	**76.0%**	**44.0%**
		14	**75.7%**	**67.7%**
		12	**72.0%**	**84.4%**

5. Conclusions and Future Work

As weights and activations are increasingly represented with fewer bits, we anticipate the accumulator to play a critical role in the quantization design space. However, while reducing the precision of the accumulator offers significant hardware efficiency improvements, it also invariably limits model accuracy by means of either numerical overflow or learning constraints (de Bruin et al., 2020; Ni et al., 2021; Xie et al., 2021; Colbert et al., 2023). Our results show that A2Q+ significantly improves this trade-off, outperforming prior methods that guarantee overflow avoidance.

A2Q+ uses zero-centering to alleviate the ℓ_1-norm constraints of A2Q, improving model accuracy without compromising overflow avoidance. It is important to note that prior work has also studied benefits of zero-centering in other contexts. Huang et al. 2017 show that normalizing weights to have zero mean and unit ℓ_2-norm can stabilize pre-activation distributions and yield better-conditioned optimization problems. Qiao et al. 2019 show that normalizing weights to instead have zero mean and unit variance can smooth the loss landscape and improve convergence. Li et al. 2019 propose a non-uniform quantization scheme that also normalizes weights to have zero mean and unit variance and report increased training stability. However, this collection of favorable properties may not directly translate to A2Q+, which is a uniform quantization scheme that normalizes each output channel of the weights to have zero mean and unit ℓ_1-norm, but we do observe that A2Q+ inherits an unfavorable property of zero-centering that seems to have been overlooked in these prior works: implicit dimensionality reduction. We observe this to only negatively impact depthwise separable convolutions (see Appendix B.2).

A2Q+ uses Euclidean projections to minimize weight quantization error at initialization. As a consequence of accumulator constraints, naïve initialization forces models to recover from superfluous quantization error as P is reduced. However, while our experiments show that minimizing the weight quantization error at initialization yields significant improvements in the resulting model accuracy, we do not observe increased post-training quantization performance. Similar to Colbert et al. 2023, we find this is due to the reliance on round-to-zero and the severity of accumulator constraints, which we highlight for future work.

Prior works have reported that weight normalization and its variants have negligible training overhead, which we observe in our experiments with A2Q+. Furthermore, because we apply our zero-centering constraint directly to the floating-point weights, it is part of the quantization mapping itself. After the model is trained, our quantization mapping is intended to be applied once offline before deployment and is therefore transparent to any hardware implementation. Thus, there is zero inference overhead as well.

Finally, A2Q+ introduces unstructured weight sparsity as the accumulator bit width is reduced. Although studies have exploited unstructured sparsity to improve inference performance on both programmable logic (Nurvitadhi et al., 2017; Colbert et al., 2021a) and general-purpose platforms (Elsen et al., 2020; Gale et al., 2020), many off-the-shelf accelerators require structured patterns to see performance benefits (Mao et al., 2017; Mishra et al., 2021). We highlight controlling weight sparsity patterns for future work.

Acknowledgements

We would like to thank Gabor Sines, Michaela Blott, Benoit Jacob, Giuseppe Franco, Mehdi Saeedi, Nicholas Malaya, Jake Daly, Max Kiehn, and Syed Naim from AMD for their feedback and support. We also thank Rayan Saab and Jinjie Zhang from UC San Diego and Alec Flowers from EPFL for insightful discussions, and the anonymous reviewers for their constructive feedback that enhanced this paper.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning by reducing the cost of querying a deployed model. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Aggarwal, S., Pappalardo, A., Damsgaard, H. J., Franco, G., Preußer, T. B., Blott, M., and Mitra, T. Post-training quantization with low-precision minifloats and integers on FPGAs. *arXiv preprint arXiv:2311.12359*, 2023.

Azamat, A., Park, J., and Lee, J. Squeezing accumulators in binary neural networks for extremely resource-constrained applications. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–7, 2022.

Baier, L., Jöhren, F., and Seebacher, S. Challenges in the deployment and operation of machine learning in practice. In *ECIS*, volume 1, 2019.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Blott, M., Preusser, T. B., Fraser, N. J., Gambardella, G., O'brien, K., Umuroglu, Y., Leeser, M., and Vissers, K. FINN-R: An end-to-end deep-learning framework for fast exploration of quantized neural networks. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 11(3):1–23, 2018.

Blumenfeld, Y., Hubara, I., and Soudry, D. Towards cheaper inference in deep networks with lower bit-width accumulators. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ NeurIPS 2023)*, 2023.

Colbert, I., Daly, J., Kreutz-Delgado, K., and Das, S. A competitive edge: Can FPGAs beat GPUs at DCNN inference acceleration in resource-limited edge computing applications? *arXiv preprint arXiv:2102.00294*, 2021a.

Colbert, I., Kreutz-Delgado, K., and Das, S. An energy-efficient edge computing paradigm for convolution-based image upsampling. *IEEE Access*, 9:147967–147984, 2021b.

Colbert, I., Pappalardo, A., and Petri-Koenig, J. A2Q: Accumulator-aware quantization with guaranteed overflow avoidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16989–16998, 2023.

de Bruin, B., Zivkovic, Z., and Corporaal, H. Quantization of deep neural networks for accumulator-constrained processors. *Microprocessors and Microsystems*, 72:102872, 2020.

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

Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pp. 272–279, 2008.

Elsen, E., Dukhan, M., Gale, T., and Simonyan, K. Fast sparse convnets. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14629–14638, 2020.

Gale, T., Zaharia, M., Young, C., and Elsen, E. Sparse GPU kernels for deep learning. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–14. IEEE, 2020.

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., and Keutzer, K. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

Hardy, G. H., Littlewood, J. E., and Pólya, G. *Inequalities*. Cambridge university press, 1952.

Hawks, B., Duarte, J., Fraser, N. J., Pappalardo, A., Tran, N., and Umuroglu, Y. Ps and Qs: Quantization-aware pruning for efficient low latency neural network inference. *Frontiers in Artificial Intelligence*, 4:676564, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Horowitz, M. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pp. 10–14. IEEE, 2014.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Huang, L., Liu, X., Liu, Y., Lang, B., and Tao, D. Centered weight normalization in accelerating training of deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2803–2811, 2017.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. 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.

Jain, S., Gural, A., Wu, M., and Dick, C. Trained quantization thresholds for accurate and efficient fixed-point inference of deep neural networks. *Proceedings of Machine Learning and Systems*, 2:112–128, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Li, H., Liu, J., Jia, L., Liang, Y., Wang, Y., and Tan, M. Downscaling and overflow-aware model compression for efficient vision processors. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 145–150. IEEE, 2022.

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

Loroch, D. M., Pfreundt, F.-J., Wehn, N., and Keuper, J. Tensorquant: A simulation toolbox for deep neural network quantization. In *Proceedings of the Machine Learning on HPC Environments*, pp. 1–8. 2017.

Lou, Q. and Jiang, L. She: A fast and accurate deep neural network for encrypted data. *Advances in Neural Information Processing Systems*, 32, 2019.

Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., and Dally, W. J. Exploring the granularity of sparsity in convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 13–20, 2017.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pp. 416–423. IEEE, 2001.

Mishra, A., Latorre, J. A., Pool, J., Stosic, D., Stosic, D., Venkatesh, G., Yu, C., and Micikevicius, P. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*, 2021.

Nagel, M., Baalen, M. v., Blankevoort, T., and Welling, M. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1325–1334, 2019.

Ni, R., Chu, H.-m., Castañeda Fernández, O., Chiang, P.-y., Studer, C., and Goldstein, T. Wrapnet: Neural net inference with ultra-low-precision arithmetic. In *International Conference on Learning Representations ICLR 2021*. OpenReview, 2021.

Nurvitadhi, E., Venkatesh, G., Sim, J., Marr, D., Huang, R., Ong Gee Hock, J., Liew, Y. T., Srivatsan, K., Moss, D., Subhaschandra, S., et al. Can FPGAs beat GPUs in accelerating next-generation deep neural networks? In *Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays*, pp. 5–14, 2017.

Odena, A., Dumoulin, V., and Olah, C. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

Pappalardo, A. Xilinx/brevitas, 2021. URL https://doi.org/10.5281/zenodo.3333552.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

Qiao, S., Wang, H., Liu, C., Shen, W., and Yuille, A. Micro-batch training with batch-channel normalization and weight standardization. *arXiv preprint arXiv:1903.10520*, 2019.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1874–1883, 2016.

Sifre, L. and Mallat, S. Rigid-motion scattering for texture classification. *arXiv preprint arXiv:1403.1687*, 2014.

Stoian, A., Frery, J., Bredehoft, R., Montero, L., Kherfallah, C., and Chevallier-Mames, B. Deep neural networks for encrypted inference with TFHE. *arXiv preprint arXiv:2302.10906*, 2023.

Umuroglu, Y. and Jahre, M. Streamlined deployment for quantized neural networks. *arXiv preprint arXiv:1709.04060*, 2017.

Umuroglu, Y., Fraser, N. J., Gambardella, G., Blott, M., Leong, P., Jahre, M., and Vissers, K. FINN: A framework for fast, scalable binarized neural network inference. In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '17, pp. 65–74. ACM, 2017.

Wu, H., Judd, P., Zhang, X., Isaev, M., and Micikevicius, P. Integer quantization for deep learning inference: Principles and empirical evaluation. *arXiv preprint arXiv:2004.09602*, 2020.

Wu, X., Li, C., Aminabadi, R. Y., Yao, Z., and He, Y. Understanding int4 quantization for language models: latency speedup, composability, and failure cases. In *International Conference on Machine Learning*, pp. 37524–37539. PMLR, 2023.

Xie, H., Song, Y., Cai, L., and Li, M. Overflow aware quantization: Accelerating neural network inference by low-bit multiply-accumulate operations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 868–875, 2021.

Yang, G., Pennington, J., Rao, V., Sohl-Dickstein, J., and Schoenholz, S. S. A mean field theory of batch normalization. *arXiv preprint arXiv:1902.08129*, 2019.

Yao, Z., Dong, Z., Zheng, Z., Gholami, A., Yu, J., Tan, E., Wang, L., Huang, Q., Wang, Y., Mahoney, M., et al. HAWQ-V3: Dyadic neural network quantization. In *International Conference on Machine Learning*, pp. 11875–11886. PMLR, 2021.

Zhang, X., Colbert, I., and Das, S. Learning low-precision structured subnetworks using joint layerwise channel pruning and uniform quantization. *Applied Sciences*, 12 (15):7829, 2022.

A. Proofs

A.1. Proof of Proposition 3.1

Let weights $\boldsymbol{q}$ be a K-dimensional vector of M-bit integers, and let $\mathbb{Z}_N^K$ denote the set of all K-dimensional vectors of N-bit integers. To prove Prop. 3.1, restated below for completeness, we examine the vectors that maximize and minimize the dot product of $\boldsymbol{q}$ by any $\boldsymbol{x} \in \mathbb{Z}_N^K$ and directly derive our result by exhaustively evaluating each case. Without loss of generality, we assume a two's complement representation for signed integers in our work as is common practice (Wu et al., 2020; Gholami et al., 2021).

Proposition 3.1. *Let $\boldsymbol{x}$ be a K-dimensional vector of N-bit integers such that the value of the i-th element x_i lies within the closed interval $[c, d]$ and $d - c = 2^N - 1$. Let $\boldsymbol{q}$ be a K-dimensional vector of signed integers centered at zero such that $\sum_i q_i = 0$. To guarantee overflow avoidance when accumulating the result of $\boldsymbol{x}^T\boldsymbol{q}$ into a signed P-bit register, it is sufficient that the ℓ_1-norm of $\boldsymbol{q}$ satisfies:*

$$\|\boldsymbol{q}\|_1 \leq \frac{2^P - 2}{2^N - 1} \tag{7}$$

Proof. Let α denote the sum of all positive elements of $\boldsymbol{q}$ and let β denote the sum of all negative elements of $\boldsymbol{q}$. It follows that $\alpha + \beta = 0$ (property of the zero-centered vector) and $\alpha - \beta = \|\boldsymbol{q}\|_1$ (property of the ℓ_1-norm). This yields the following relationships: $\alpha = -\beta = \frac{1}{2}\|\boldsymbol{q}\|_1$.

Let the closed interval $[e, f]$ denote the output range of the dot product of $\boldsymbol{q}$ by any $\boldsymbol{x} \in \mathbb{Z}_N^K$, where $f \geq e$. To safely use a signed P-bit accumulator without overflow, all of the following inequalities need to be satisfied:

$$f \leq 2^{P-1} - 1 \tag{23}$$
$$-e \leq 2^{P-1} \tag{24}$$
$$f - e \leq 2^P - 1 \tag{25}$$

We start with the first inequality, Eq. 23. Since the value of each input element x_i is bounded to the closed interval $[c, d]$, the maximizing vector $\boldsymbol{\mu} = \arg\max_{\boldsymbol{x}} \boldsymbol{x}^T\boldsymbol{q}$ is defined as:

$$\mu_i = \begin{cases} d, & \text{where } q_i \geq 0 \\ c, & \text{where } q_i < 0 \end{cases} \tag{26}$$

Exploiting the identities of α, β, c, d, and f, we can derive the following upper bound on the ℓ_1-norm of $\boldsymbol{q}$:

$$\boldsymbol{\mu}^T\boldsymbol{q} \leq 2^{P-1} - 1 \tag{27}$$
$$d\alpha + c\beta \leq 2^{P-1} - 1 \tag{28}$$
$$\alpha(d - c) \leq 2^{P-1} - 1 \tag{29}$$
$$\|\boldsymbol{q}\|_1 \leq \frac{2^P - 2}{2^N - 1} \tag{30}$$

Note that this aligns with Prop. 3.1. Next, we prove that satisfying this bound will also satisfy Eqs. 24 and 25.

We continue onto Eq. 24. Similar to Eq. 26, the minimizing vector $\boldsymbol{\nu} = \arg\min_{\boldsymbol{x}} \boldsymbol{x}^T\boldsymbol{q}$ is defined as:

$$\nu_i = \begin{cases} c, & \text{where } q_i \geq 0 \\ d, & \text{where } q_i < 0 \end{cases} \tag{31}$$

Again exploiting our defined identities, we can derive the following upper bound on the ℓ_1-norm of $\boldsymbol{q}$:

$$-\boldsymbol{\nu}^T\boldsymbol{q} \leq 2^{P-1} \tag{32}$$
$$-c\alpha - d\beta \leq 2^{P-1} \tag{33}$$
$$\alpha(d - c) \leq 2^{P-1} \tag{34}$$
$$\|\boldsymbol{q}\|_1 \leq \frac{2^P}{2^N - 1} \tag{35}$$

Note that by satisfying Eq. 30, we also satisfy Eq. 35.

Finally, we evaluate the last inequality, Eq. 25. From Eqs. 26 and 31, it follows that $\mu_i - \nu_i = (d - c)\text{sign}(q_i)$. With this new identity, we can derive the following upper bound on the ℓ_1-norm of $\boldsymbol{q}$:

$$(\boldsymbol{\mu} - \boldsymbol{\nu})^T\boldsymbol{q} \leq 2^P - 1 \tag{36}$$
$$(d - c)\text{sign}(\boldsymbol{q})^T\boldsymbol{q} \leq 2^P - 1 \tag{37}$$
$$\|\boldsymbol{q}\|_1 \leq \frac{2^P - 1}{2^N - 1} \tag{38}$$

Thus, by satisfying Eq. 30, we also satisfy both Eqs. 35 and 38, enabling the use of a signed P-bit accumulator. □

A.2. Proof of Proposition 3.2

To prove Prop. 3.2, we first present the following lemma:

Lemma A.1. *Let $\boldsymbol{x}$, $\boldsymbol{w}$, and $\boldsymbol{q}$ each be K-dimensional vectors. If $\text{sign}(x_i) = \text{sign}(w_i) = \text{sign}(q_i)$ for all non-zero x_i and $|q_i| \leq |w_i|$ for all i, then $\boldsymbol{x}^T\boldsymbol{q} \leq \boldsymbol{x}^T\boldsymbol{w}$.*

Proof. Given that $\text{sign}(x_i) = \text{sign}(w_i) = \text{sign}(q_i)$ for all non-zero x_i, it follows that $\boldsymbol{x}^T\boldsymbol{q} = \sum_i |x_i||q_i|$ and $\boldsymbol{x}^T\boldsymbol{w} = \sum_i |x_i||w_i|$. Using these identities, we can directly derive the following inequality:

$$\boldsymbol{x}^T\boldsymbol{q} \leq \boldsymbol{x}^T\boldsymbol{w} \tag{39}$$
$$\textstyle\sum_i |x_i||q_i| \leq \sum_i |x_i||w_i| \tag{40}$$
$$\textstyle\sum_i |x_i|\left(|q_i| - |w_i|\right) \leq 0 \tag{41}$$

Given that $|q_i| \leq |w_i|$ for all i, this leads us to the desired result that the inequality holds, *i.e.*, $\boldsymbol{x}^T\boldsymbol{q} \leq \boldsymbol{x}^T\boldsymbol{w}$. □

Consider again inputs $\boldsymbol{x}$ and integer-quantized weights $\boldsymbol{q}$. Recall that simulated quantization derives $\boldsymbol{q}$ from floating-point counterpart $\boldsymbol{w}$ using a transformation function referred

to as a quantizer. To prove Prop. 3.2, we leverage the necessary accumulator constraint conditions formally articulated in Appendix A.1: Eqs. 7, 23, 24, and 25. We again directly derive our result by exhaustively evaluating each case.

Proposition 3.2. *Let $\boldsymbol{x}$ be a vector of N-bit integers such that the i-th element x_i lies within the closed interval $[c, d]$ and $d - c = 2^N - 1$. Let $\boldsymbol{w}$ be a zero-centered vector such that $\sum_i w_i = 0$. Let $Q(\boldsymbol{w})$ be a symmetric quantizer parameterized in the form of Eq. 2, where $Q(\boldsymbol{w}) = s \cdot \boldsymbol{q}$ for strictly positive scaling factor s and integer-quantized weight vector $\boldsymbol{q}$. Given that $\text{sign}(s \cdot q_i) = \text{sign}(w_i)$ and $|s \cdot q_i| \leq |w_i|$ for all i, then $\boldsymbol{x}^T\boldsymbol{q}$ can be safely accumulated into a signed P-bit register without overflow if $\boldsymbol{w}/s$ satisfies all necessary conditions for such a constraint.*

Proof. As shown in Section 3.1, $\boldsymbol{q}$ must satisfy Eqs. 7, 23, 24, and 25 to avoid overflow when accumulating the result of $\boldsymbol{x}^T\boldsymbol{q}$ into a P-bit register. To show that $\boldsymbol{q}$ satisfies these four necessary conditions when $\boldsymbol{w}/s$ does as well, we directly prove each case, starting with Eq. 7. Given that $|s \cdot q_i| \leq |w_i|$ for all i and s is a strictly positive scalar, it follows that $|q_i| \leq |w_i/s|$ and thus $\|\boldsymbol{q}\|_1 \leq \|\boldsymbol{w}/s\|_1$. Therefore, when $\boldsymbol{w}/s$ satisfies Eq. 7, then $\boldsymbol{q}$ does as well.

To evaluate Eq. 23, let $\boldsymbol{\mu}$ be the vector that maximizes $\boldsymbol{x}^T\boldsymbol{w}$ as defined in Eq. 26. Given that s is strictly positive and $\text{sign}(s \cdot q_i) = \text{sign}(w_i)$, it follows that $\text{sign}(q_i) = \text{sign}(w_i)$ and thus $\boldsymbol{\mu}$ also maximizes $\boldsymbol{x}^T\boldsymbol{q}$. Furthermore, given that μ_i is an N-bit integer, the closed interval $[c, d]$ is defined as $[-2^{N-1}, 2^{N-1} - 1]$ when μ_i is signed and $[0, 2^N - 1]$ when unsigned. It follows that $\text{sign}(\mu_i) = \text{sign}(w_i) = \text{sign}(q_i)$ for all non-zero μ_i and $|q_i| \leq |w_i/s|$ for all i, and thus $\boldsymbol{\mu}^T\boldsymbol{q} \leq \boldsymbol{\mu}^T\boldsymbol{w}/s$ by Lemma A.1. Therefore, when $\boldsymbol{w}/s$ satisfies Eq. 23, then so does $\boldsymbol{q}$.

Similarly, let $\boldsymbol{\nu}$ be the vector that minimizes $\boldsymbol{x}^T\boldsymbol{w}$ as defined in Eq. 31. Given that $\text{sign}(q_i) = \text{sign}(w_i)$, then $\boldsymbol{\nu}$ also minimizes $\boldsymbol{x}^T\boldsymbol{q}$. It again follows that $\text{sign}(-\nu_i) - \text{sign}(w_i) = \text{sign}(q_i)$ for all non-zero ν_i and $|q_i| \leq |w_i/s|$ for all i, and thus $-\boldsymbol{\nu}^T\boldsymbol{q} \leq -\boldsymbol{\nu}^T\boldsymbol{w}/s$ by Lemma A.1. Therefore, $\boldsymbol{q}$ satisfies Eq. 24 when $\boldsymbol{w}/s$ does as well.

Following the same logic for Eq. 25, $\text{sign}(\mu_i - \nu_i) = \text{sign}(w_i) = \text{sign}(q_i)$ for all i where $\mu_i \neq \nu_i$, and thus $(\boldsymbol{\mu} - \boldsymbol{\nu})^T\boldsymbol{q} \leq (\boldsymbol{\mu} - \boldsymbol{\nu})^T\boldsymbol{w}/s$, again by Lemma 25. Therefore, when $\boldsymbol{w}/s$ satisfies Eq. 25, then so does $\boldsymbol{q}$, leading to the desired result for all four necessary conditions. □

B. Experiment Details & Ablations

B.1. Hyperparameters & Quantization Schemes

Below, we provide further details on training hyperparameters, neural network architectures, and quantization schemes for our image classification and single-image super resolution benchmarks. As we are building from the work of Colbert et al. 2023, we adopt a quantization scheme that is amenable to compilation through FINN (Umuroglu et al., 2017), where batch normalization layers, floating-point biases, and even scaling factors are absorbed into thresholding units via mathematical manipulation during graph compilation (Umuroglu & Jahre, 2017). Thus, we are not constrained to rely on power-of-2 scaling factors, quantized biases, or batch-norm folding as is common for integer-only inference (Jacob et al., 2018; Wu et al., 2020; Gholami et al., 2021). For all models, we fix the first and last layers to 8-bit weights and activations for all configurations, as is common practice (Wu et al., 2020; Gholami et al., 2021).

Following the quantization scheme of Colbert et al. 2023, we apply A2Q and A2Q+ to only the weights of a QNN and adopt the regularization penalty defined in Eq. 42 to avoid g getting stuck when $g > T_+$.

$$R = \max\{g - T_+, 0\} \tag{42}$$

This penalty is imposed on every hidden layer and combined into one regularizer: $\mathcal{L}_{\text{reg}} = \sum_l \sum_i R_{l,i}$, where $R_{l,i}$ denotes the regularization penalty for the i-th output channel in the l-th layer of the network. We scale this regularization penalty $\mathcal{L}_{\text{reg}}$ by a constant scalar $\lambda = 1e-3$ such that $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda\mathcal{L}_{\text{reg}}$, where $\mathcal{L}_{\text{task}}$ is the task-specific loss.

Baseline QAT. Our baseline QAT method is synthesized from common best practices. Similar to A2Q and A2Q+, we symmetrically constrain the weight quantization scheme around the origin such that $z = 0$ while allowing activations to be asymmetric (Gholami et al., 2021; Zhang et al., 2022). Eliminating these zero points on the weights reduces the computational overhead of cross-terms during integer-only inference (Jacob et al., 2018; Jain et al., 2020). We use unique floating-point scaling factors for each output channel (or neuron) to adjust for varied dynamic ranges (Nagel et al., 2019). However, extending this strategy to activations can be computationally expensive (Jain et al., 2020). As such, we use per-tensor scaling factors for activations and per-channel scaling factors on the weights, as is standard practice (Jain et al., 2020; Wu et al., 2020; Zhang et al., 2022). Similar to A2Q and A2Q+, all scaling factors are learned in the log domain such that $s = 2^d$, where d is a log-scale learnable parameter. The scaled weights (or activations) are rounded to the nearest integer and clipped to the limits of the representation range (see Section 2.2). Noticeably, the rounding function introduces extremely sparse gradients; therefore, we use the straight-through estimator (Bengio et al., 2013) during training to allow gradients to permeate the rounding function such that $\nabla_x \lfloor x \rceil = 1$ everywhere and ∇_x denotes the gradient with respect to x.

ImageNet models. When training ResNet (He et al., 2016) models on ImageNet (Deng et al., 2009), we leverage the unmodified implementations from PyTorch (Paszke et al.,

2019) as well as their pre-trained floating-point checkpoints. We use batch sizes of 64 images with an initial learning rate of 1e-4 that is reduced by a factor of 0.1 at epochs 30 and 50. We fine-tune all models for 60 epochs using the standard stochastic gradient descent (SGD) optimizer with a weight decay of 1e-5. Before fine-tuning, we apply the graph equalization and bias correction techniques proposed by Nagel et al. 2019 using a calibration set of 3000 images randomly sampled from the training dataset. When applying our Euclidean projection-based weight initialization strategy discussed in Section 3.2, we do so after graph equalization, but before bias correction. Finally, although common practice is to keep residuals as 32-bit additions (Yao et al., 2021), we quantize our residual connections to 8 bits to reduce the cost of such high-precision additions.

CIFAR10 models. When training MobileNetV1 (Howard et al., 2017) and ResNet18 (He et al., 2016) to classify images on the CIFAR10 dataset (Krizhevsky et al., 2009), we follow the modified network architectures used by Colbert et al. 2023. These modifications reduce the degree of downsampling throughout these networks to yield intermediate representations that are more amenable to the smaller image sizes of CIFAR10. For MobileNetV1, we use batch sizes of 64 images with an initial learning rate of 1e-3 that is reduced by a factor of 0.9 every epoch. For ResNet18, we use batch sizes of 256 with an initial learning rate of 1e-3 that is reduced by a factor of 0.1 every 30 epochs. We use a weight decay of 1e-5 for both models. We initialize all quantized models from pre-trained floating-point checkpoints and fine-tune for 100 epochs using the standard SGD optimizer. We again apply the graph equalization and bias correction techniques before fine-tuning, but using a calibration set of 1000 images. We again use our Euclidean projection-based weight initialization strategy after graph equalization, but before bias correction. Finally, we further quantize our residual additions to the same bit width specified for our hidden activations, *i.e.*, N.

BSD300 models. When training ESPCN (Shi et al., 2016) and U-Net (Ronneberger et al., 2015) to upscale images by a factor of $3\times$ using the BSD300 dataset (Martin et al., 2001), we again follow the modified architectures used by Colbert et al. 2023. These modifications rely on the nearest neighbor resize convolution to upsample intermediate representations to improve model accuracy during training (Odena et al., 2016), without impacting inference efficiency (Colbert et al., 2021b). For both models, we use batch sizes of 8 images with an initial learning rate of 1e-3 that is reduced by a factor of 0.999 every epoch and again use a weight decay of 1e-5. We randomly initialize all models according to He et al. 2015 and train them from scratch for 300 epochs using the Adam optimizer (Kingma & Ba, 2014). Similar to the CIFAR10 models, we quantize our residual additions in U-Net to the hidden activation bit width N.

B.2. A2Q+ for Depthwise Separable Convolutions

A2Q+ relies on zero-centering the weights for the purpose of alleviating the overly restrictive ℓ_1-norm constraints of A2Q. As discussed in Section 5, several studies have investigated the impact of zero-centering within the context of weight normalization (Huang et al., 2017; Qiao et al., 2019; Li et al., 2019). While these works highlight the favorable properties of zero-centered weight normalization, such as stabilized pre-activation distributions and improved convergence, they seem to overlook an unfavorable property: implicit dimensionality reduction.

Given K-dimensional weight vector $\boldsymbol{w}$, the zero-centering operation can be interpreted as a projection onto a $K-1$ hyperplane (Yang et al., 2019). This implies that such a constraint reduces the degrees of freedom of the zero-centered weight vector, and such a reduction has a more significant impact with smaller K. In the context of our work, we find that this introduces issues when handling layers with smaller dot product sizes, as is the case with the depthwise separable convolutions (Sifre & Mallat, 2014) commonly used in MobileNets (Howard et al., 2017).

Depthwise separable convolutions factorize the standard convolution into two chained operations: (1) a depthwise convolution that applies a single filter to each input channel; followed by (2) a pointwise convolution that applies a 1×1 kernel that combines the resulting output channels (Sifre & Mallat, 2014; Howard et al., 2017). The size of pointwise convolution dot products is equivalent to the number of the input channels in the layer, which tend to be large. However, size of depthwise convolution dot products is equivalent to the size of the kernels, which tend to be orders of magnitude smaller (Howard et al., 2017). Prior studies on zero-centered weight normalization focus on benchmarks without these convolutions, *e.g.*, VGGs and ResNets. In the scope of our work, we find that zero-centering the weights of depthwise convolutions negatively impacts model accuracy.

In Fig. 4, we evaluate the impact of zero-centering on depthwise separable convolutions. We visualize the maximum test top-1 accuracy observed over 3 experiments when training MobileNetV1 to classify CIFAR10 images assuming 4-bit weights and activations (W4A4). Using the hyperparameters detailed in Section B.1, we find that using A2Q for depthwise convolutions and A2Q+ for pointwise convolutions, we are able to recover model accuracy lost to implicit dimensionality reduction. In addition, we are able to improve the trade-off between model accuracy and accumulator bit width. Thus, we use the mixed depthwise separable convolution wherever possible in this work.

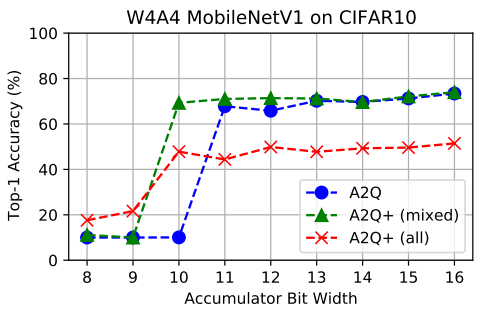


Figure 4. We evaluate the impact of zero-centering on depthwise convolutions as we reduce the target accumulator bit width. We visualize the maximum observed test top-1 accuracy when training a W4A4 MobileNetV1 model on CIFAR10. We show that using A2Q for all depthwise convolutions and A2Q+ for all other hidden layers (**green triangles**) outperforms uniformly applying A2Q (**blue circles**) or A2Q+ (**red crosses**) to all hidden layers.

B.3. Impact of Euclidean Projection Initialization

In Section 3.2, we introduce a Euclidean projection-based weight initialization strategy designed to minimize the quantization error when initializing A2Q and A2Q+ models from pre-trained floating-point checkpoints. We refer to this strategy as EP-init. Our results in Section 4.2 show that EP-init significantly improves A2Q as a reference baseline for ImageNet models. This section provides a deeper empirical analysis of the initial weight quantization error. We additionally discuss our ablation study that isolates the impact of EP-init across quantizers and target accumulator bit widths.

B.3.1. Initial Weight Quantization Error

A common practice when initializing QNNs from pre-trained floating-point checkpoints is to define scaling factor s to be the ratio given by Eq. 43 (Gholami et al., 2021; Zhang et al., 2022; Aggarwal et al., 2023). Here, $\max(|\boldsymbol{w}_{\text{float}}|)$ is the maximum observed floating-point weight magnitude defined per-output channel and M is the target weight bit width defined per-tensor.

$$s = \frac{\max(|\boldsymbol{w}_{\text{float}}|)}{2^{M-1} - 1} \tag{43}$$

Using A2Q and A2Q+ to fine-tune QNNs from pre-trained floating-point checkpoints requires initializing two new learnable parameters: g and $\boldsymbol{v}$. One could trivially initialize $\boldsymbol{v}$ to be the pre-trained floating-point weight vector $\boldsymbol{w}_{\text{float}}$ and g to be its ℓ_1-norm such that $\boldsymbol{v} = \boldsymbol{w}_{\text{float}}$ and $g = \|\boldsymbol{w}_{\text{float}}\|_1$ to ensure $\boldsymbol{w} = \boldsymbol{w}_{\text{float}}$. While this works well when targeting high-precision accumulators (*e.g.*, 32 bits), we observe that A2Q-quantized networks are forced to quickly recover from extremely high losses when targeting low-precision accumulation scenarios (*e.g.*, 16 bits or fewer).

Figure 5 visualizes the test cross entropy loss when training various W4A4 ResNets to classify ImageNet images while targeting 14-bit accumulators. A2Q-quantized networks do not fully recover when naïvely initialized. Upon deeper investigation, we identify that this is in large part a consequence of A2Q clipping g according to T in Eq. 3.

We first analyze the pre-trained floating-point ResNet checkpoints to demonstrate the breadth of this problem. We evaluate $\|\boldsymbol{w}_{\text{float}}/s\|_1$ for each output channel in each hidden layer of each ImageNet model using the scaling factor definition provided in Eq. 43. We visualize the results as an empirical cumulative distribution function (CDF) in Fig. 6. This CDF shows the percentage of channels in each W4A4 ImageNet model that inherently satisfies 14-, 16-, and 18-bit accumulator constraints assuming A2Q is the weight quantizer. While all per-channel weight vectors satisfy a 18-bit accumulator constraint upon initialization, we observe that only 52% of ResNet18 channels inherently satisfy a 16-bit accumulator constraint and a mere 23% inherently satisfy a 14-bit accumulator constraint. For ResNet34, we observe 47% satisfy the 16-bit constraint and 14% satisfy the 14-bit constraint. Interestingly, we observe that 92% of ResNet50 channels inherently satisfy the 16-bit constraint and 46% satisfy the 14-bit constraint. We hypothesize this is because our accumulator constraints tighten with the width rather than the depth of a neural network. This allows model capacity to increase without tightening of constraints. It is important to note that these observations are dependent on the exact weight values of the pre-trained floating-point checkpoint. We use the standard pre-trained floating-point checkpoints provided by PyTorch (Paszke et al., 2019) within the scope of this work and leave an exhaustive analysis of other checkpoints for future work.

As a consequence of the ℓ_1-norm constraints, naïvely initializing g such that $g = \|\boldsymbol{w}_{\text{float}}\|_1$ significantly increases the initial weight quantization error when $\|\boldsymbol{w}_{\text{float}}\|_1 > T$. Consider again the same ResNet models and let the weight quantization error at initialization be defined as follows:

$$\frac{1}{2}\|Q(\boldsymbol{w}) - \boldsymbol{w}_{\text{float}}\|_2^2 \tag{44}$$

To demonstrate how this weight quantization error increases as the target accumulator bit width is reduced, we independently evaluate Eq. 44 for each output channel and plot the average in Fig. 7. To account for the varied sizes of each layer in the network, we normalize the quantization error of each output channel by the squared ℓ_2-norm of the floating-point weights, formally defined as $\frac{1}{2}\|\boldsymbol{w}_{\text{float}}\|_2^2$. Noticeably, when initializing g and $\boldsymbol{v}$, the average weight quantization error increases exponentially with the reduction in accumulator bit width regardless of the strategy. In fact, when targeting 10-bit accumulation, $Q(\boldsymbol{w})$ is initialized with nearly 100% sparsity across all output channels. However, we are

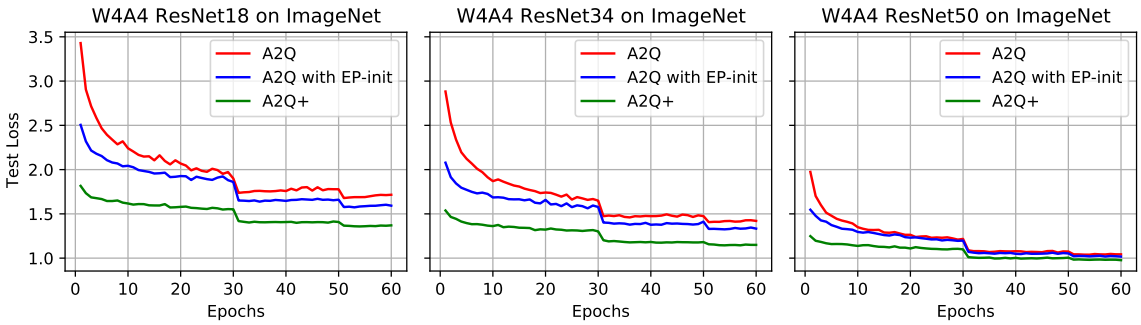


Figure 5. We visualize the test cross entropy loss when training ResNet18, ResNet34, and ResNet50 to classify ImageNet images using 4-bit weights and activations (W4A4) and targeting 14-bit accumulation using A2Q. We observe that our Euclidean projection initialization (EP-init) helps improve convergence. Note that respective test top-1 accuracies are detailed in Table 3.

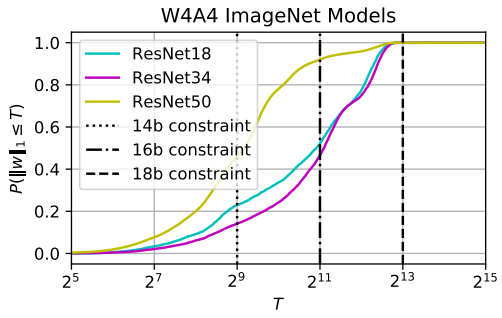


Figure 6. We provide an empirical CDF to visualize the percentage of output channels in various A2Q-quantized W4A4 ResNets that inherently satisfies 14-, 16-, and 18-bit ℓ_1-norm constraints when first initialized from a pre-trained ImageNet checkpoint.

able to effectively minimize initial weight quantization error when using EP-init. We additionally observe that combining this strategy with our new bound further reduces the initial weight quantization error as the ℓ_1-norm constraints are relaxed. We show this reduced weight quantization error yields improved model accuracy in Section 4.2.

B.3.2. Ablation Study on CIFAR10

When constructing our Pareto frontiers in Section 4.1, we applied EP-init to all A2Q and A2Q+ models to strictly compare the quantizers without the influence of initialization. To isolate the influence of initialization, we detail an ablation study that further investigates the impact of EP-init on model accuracy. Our analysis aims to further connect initial weight quantization error to model accuracy. Thus, we focus on ResNet18 trained on the CIFAR10 dataset. Building from the ImageNet analysis, we again focus on 4-bit weights and activations (W4A4).

We first analyze $\|\boldsymbol{w}_{\text{float}}/s\|_1$ for each output channel in each layer of the model using the scaling factor initialization defined in Eq. 43. We again calculate the percentage of channels that inherently satisfy various ℓ_1-norm constraints and visualize the analysis as an empirical CDF in Fig. 8a. We observe that all channels natively satisfy an 18-bit accumulator constraint with only 53% satisfying a 16-bit constraint and only 19% satisfying a 14-bit constraint.

Next, we evaluate the initial weight quantization error as the target accumulator bit width is reduced. We again normalize the quantization error of each output channel by its squared ℓ_2-norm. We visualize the results in Fig. 8b, where we plot the average weight quantization error for each target accumulator bit width for both A2Q and A2Q+ models with and without EP-init. Similar to our ImageNet results, we observe that combing our new bound with EP-init yields the lowest initial weight quantization error across target accumulator bit widths.

Finally, we evaluate the how initial weight quantization error translates to model accuracy as we reduce the target accumulator bit width for both A2Q and A2Q+ models. In Fig. 8c, we visualize the maximum test top-1 accuracy observed over 3 experiments. Intuitively, the strategy for initialization becomes more important as the target accumulator bit width is reduced. The impact of EP-init increases as the expected initial weight quantization error increases, with the highest impact in the extremely low-precision accumulation regime. In fact, we observe that EP-init yields up to a $+50\%$ increase in test top-1 accuracy for both A2Q+ and A2Q when targeting 9- and 10-bit accumulation, respectively.

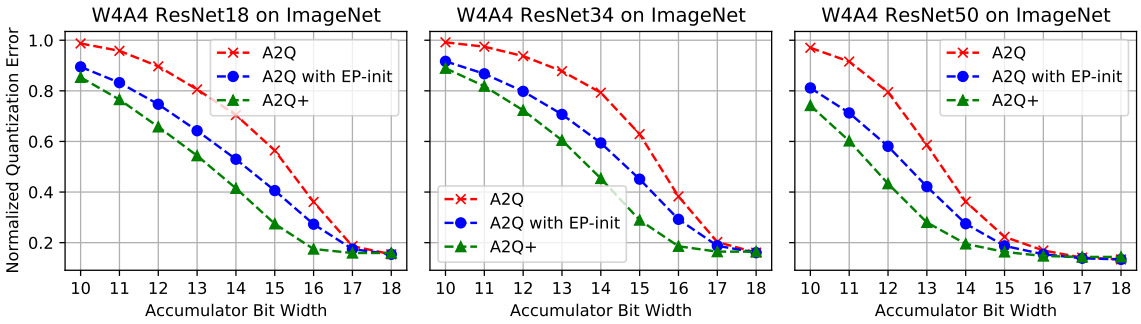
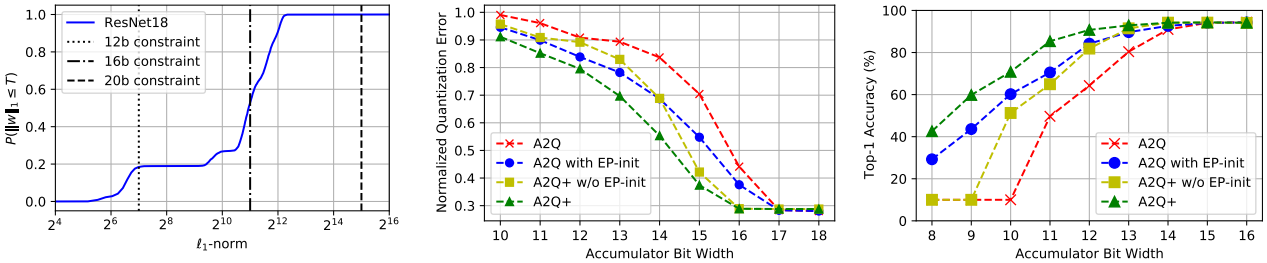


Figure 7. We evaluate the normalized weight quantization error averaged over each output channel when initializing W4A4 ResNet variants from a pre-trained floating-point models trained on ImageNet. As the accumulator bit width is reduced, we observe that our Euclidean projection initialization (EP-init) yields less error than naïve initialization for A2Q. We additionally show that A2Q+ yields the lowest initial weight quantization error by combining EP-init with our new bound.



(a) (b) (c)

Figure 8. We evaluate the impact of Euclidean projection-based weight initialization (EP-init) as we reduce the target accumulator bit width for W4A4 ResNet18 trained on CIFAR10: (a) we visualize an empirical CDF to visualize the percentage of output channels that inherently satisfies various ℓ_1-norm constraints; (b) we visualize the initial weight quantization error for both A2Q and A2Q+ with and without EP-init; and (c) we visualize the maximum observed test top-1 accuracy for both A2Q and A2Q+ both with and without EP-init.